Multi Scale Dictionary Learning for Sparse Representation of Images

Boaz Ophir
Multi Scale Dictionary Learning
for
Sparse Representation of Images

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Boaz Ophir

Submitted to the Senate
of the Technion — Israel Institute of Technology
Adar Bet 5776 Haifa March 2016
This research was carried out under the supervision of Prof. Michael Elad, in the Faculty of Computer Science.

Acknowledgements

I would like to thank my advisor, Prof. Michael Elad, for his support and guidance throughout the course of my PhD studies. Mostly I would like to thank him for his unwavering belief in me and my research even when I did not.

I would like to thank my collaborators Dr. Michael Zibulevsky and Prof. Michael Lustig for their contributions, and a special warm thanks to Jeremias Sulam, who brought a new energy to this project without which I may not have reached this point.

Thanks go also to my colleagues Ron Rubinstein, Raja Giryes and Sharon Alpert for our many conversations and sharing of ideas.

Finally my gratitude and love go to my family, my parents and especially my wife Naama, for bearing with me throughout this journey, giving me endless love and support.

The Technion’s funding of this research is hereby acknowledged.
# Contents

## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
</table>

## Abstract

1 Introduction
- 1.1 Motivation
- 1.2 Background
- 1.3 K-SVD on Wavelets
- 1.4 Online Sparse Dictionary Learning
- 1.5 Thesis Structure

2 Background
- 2.1 Sparse Representations
  - 2.1.1 Sparse Coding
  - 2.1.2 Learning Signal/Image Dictionaries
- 2.2 Multi-scale Transforms
  - 2.2.1 Wavelets
- 2.3 Previous Works
  - 2.3.1 Adapting Wavelets for Sparsity
  - 2.3.2 Semi Multi-Scale Learnt Dictionaries
  - 2.3.3 Sparse K-SVD
  - 2.3.4 Structured/Constrained Dictionaries
  - 2.3.5 Conclusion

3 K-SVD on Wavelets
- 3.1 Introduction
- 3.2 K-SVD in the Wavelet Domain
  - 3.2.1 The Core Approach
  - 3.2.2 Options and Flexibilities in the Proposed Scheme
  - 3.2.3 Sparse Coding
- 3.3 Experiments
  - 3.3.1 M-term Approximation
  - 3.3.2 Compressed Sensing
## List of Figures

2.1 Schematic Sparse Coding. .............................................. 12  
2.2 Schematic Wavelet analysis and synthesis ............................. 16  
2.3 Sparsity inducing prior. .............................................. 17  
2.4 Quadtree structure. .................................................. 18  
2.5 Example sparse coding. .............................................. 18  
2.6 Double-Sparse Coding. .............................................. 19  

3.1 Bands from the Wavelet transform of an image. ...................... 23  
3.2 K-SVD applied to the Wavelet domain ................................ 24  
3.3 Multi-scale atom visualization. ..................................... 25  
3.4 Multi-band atoms. .................................................... 27  
3.5 Training Images - fingerprint data set. .............................. 31  
3.6 Training Images - coastal scenery data set. ......................... 31  
3.7 M-Term approximation of fingerprint image. ....................... 32  
3.8 M-Term approximation of coastal scenery image. .................... 32  
3.9 M-Term approximation. .............................................. 32  
3.10 Compressed Sensing experiment. ................................... 34  

4.1 M-Term denoising. .................................................... 38  
4.2 Joint sparse coding. .................................................. 40  
4.3 Denoising example. ................................................... 41  
4.4 Residual error after denoising. ..................................... 42  
4.5 Denoising results - coastal scenery dataset. ....................... 43  
4.6 Denoising results - standard images. ................................ 44  

5.1 Border extensions for different methods. ............................ 48  
5.2 Standard and separable Wavelet decompositions. .................... 49  
5.3 Cascaded implementations of standard Wavelet decompositions. .... 51  
5.4 Cascaded implementations of separable Wavelet decompositions. .... 51  
5.5 Average reconstruction error for 1D synthetic data. ............... 52  
5.6 Average reconstruction error for 1D image data. ................... 53  
5.7 Point-wise reconstruction error. .................................... 53  
5.8 Average reconstruction error for 2D image data. ................... 54
6.1 Experiment 1: Run time comparison. ........................................... 62
6.2 Experiment 2: Convergence comparison. ............................... 63
6.3 Experiment 2: Dictionary atoms. ............................................. 64
6.4 Experiment 3: Denoising as a function of patch size. ............... 66
6.5 Experiment 4: Sample images. ................................................. 68
6.6 Experiment 4: Sample atoms. .................................................. 68
6.7 Experiment 4: Compression results. ....................................... 68
6.8 Experiment 5: Subset of the dictionary for patches of size 32 × 32. 70
6.9 Experiment 5: Atoms with recurring patterns. ......................... 71
6.10 Experiment 5: M-term approximation. ................................... 72
6.11 Experiment 6: M-term approximation. ................................... 73
6.12 Experiment 6: Subset of the dictionary for patches of size 64 × 64. 74
Abstract

The main topic of our research is creating a multi-scale dictionary learning paradigm for sparse and redundant signal and image representations. The appeal of a multi-scale dictionary is obvious - in many cases data naturally comes at different scales. To date, popular dictionary based approaches are limited to a single scale, and small signals/patch sizes. Multi-scale approaches, on the other hand, are typically analytic, with little to no adaptation to the data. A multi-scale dictionary should be able to combine the advantages of generic multi-scale representations (such as Wavelets), with the power of learnt dictionaries, in capturing the intrinsic characteristics of a family of signals. Using such a dictionary would allow representing the data in a more efficient, i.e. sparse, manner, allowing applications to take a more global look at the signal. In our work we aim to achieve this goal without incurring the prohibitive costs of an explicit dictionary with large atoms.

We present two approaches to tackle this problem:

The first approach is based on learning the dictionary, in parts, in the analysis domain of an orthogonal multi-scale operator (namely, orthogonal Wavelets). While our analysis domain atoms are small, when applying the inverse Wavelet transform as part of the synthesis process, the “effective” atoms we get can be very large. Using this approach we get promising results, both in achieving sparser representations of images, and as a sparsifying dictionary for compressed sensing. We show how by combining this approach with standard single-scale denoising we can achieve state-of-the-art image denoising results.

In the second approach, we plug in a multi-scale dictionary as the base/fixed part of a double-sparsity model. The multi-scale dictionary we use is Cropped Wavelets, a Wavelet based dictionary specially adapted to representing finite support signals/images without significant border affects. Using this model we can extend patch-based image processing tasks to handle large image patches (i.e., 64 × 64 pixels). In order to train a dictionary for such large signals, using this model, we introduce an online dictionary learning algorithm that uses stochastic gradient techniques to replace the standard batch approach. This algorithm allows training a dictionary on millions of examples.
We demonstrate the potential of such dictionaries for tasks such as image restoration and compression, as well as training general dictionaries for natural images.
Chapter 1

Introduction

1.1 Motivation

The motivation for this work is twofold. On the one hand, the fundamental idea behind dictionary learning for sparse representation is adapting the model to best fit our data. When dealing with images, a basic quality of the signals is their multi-scale nature. The obvious conclusion is that in order to best handle image data, dictionaries must in some way capture this quality.

The other motivation stems from the fact that local patch based approaches are apparently reaching their limit, at least in some applications such as denoising. This has been shown both theoretically [LN11] and in practice where, in the last few years, only marginal improvements have been made. A more global approach may break through this barrier.

Simply increasing atom size is not a viable option. The computational (and memory) costs this implies are prohibitively high not to mention that the degrees of freedom large atoms provide would require vast amounts of training data to avoid over-fitting.

All this leads to a need for a structured multi-scale dictionary learning paradigm for sparse and redundant signal and image representations. A multi-scale dictionary should be able to combine the advantages of generic multi-scale representations (such as Wavelets), with the power of learnt dictionaries, in capturing the intrinsic characteristics of a family of signals. Using such a dictionary would allow representing the data in a more efficient, i.e. sparse, manner, allowing applications to take a more global look at the signal. In our work we aim to achieve this goal without incurring the prohibitive costs of an explicit dictionary with large atoms.

1.2 Background

The field of sparse representations is comprised, as we see it, of two main flavors, differing in the basic approach to constructing the signal dictionary. These are:

- Analytic dictionaries
Trained dictionaries

Analytic dictionaries are fixed dictionaries, arising from a fixed signal model, and are described by analytic formulas or algorithms. These dictionaries (or transforms) are typically applied using fast algorithms, and have known mathematical properties, e.g. coefficient decay rates. In fact, many of these dictionaries are optimal in terms of sparse representation, each for a certain class of signals (for instance, piecewise smooth signals). Another favorable quality of many such dictionaries is their inherent multi-scale quality. The limitations of these dictionaries stem from the fact that they treat all signals the same way, and can not be adapted to any specific class of signals. Examples of such dictionaries include the Cosine transform, Gabor transform, Wavelets and many others.

Trained dictionaries, as the name states, arise from a set of example signals. By learning the dictionary from actual data, these dictionaries can adapt to the target signals, allowing for a sparser representation. This in turn translates into superior performance in many applications. The down side is that these dictionaries are expressed as explicit matrices. This means that their application is inefficient as it involves large matrix multiplications. These matrices are in general unstructured, meaning they are also expensive to store and transmit. The many degrees of freedom also mean a large number of training examples is required in order to avoid over fitting. Lastly, these dictionaries are single scale, meaning they are adapted to a specific signal size and inter-scale relations are not expressed.

The goal of this work is to bring these two worlds together, creating a multi-scale dictionary learning framework that hopefully combines the benefits of both. We present in this dissertation two different approaches to multi-scale dictionary learning both combining an analytic multi-scale transform with a learnt dictionary.

1.3 K-SVD on Wavelets

The Wavelet transform is known to give a sparse representation of the original signal to some degree. What we would like to do is squeeze out some of the redundancy left by the Wavelet decomposition, specifically the spatial correlation between Wavelet coefficients in the same band, or between bands, thus producing sparser image representations than plain Wavelet decompositions. Capturing this structure is the essence of the learning process.

Incorporating Wavelets in a straightforward way in the synthesis model of the dictionary learning problem yields the following optimization problem:

$$\arg\min_{D,X} \|Y - WS DX\|_{F}^2 \quad s.t. \quad \|x_i\|_0 \leq T.$$  \hspace{1cm} (1.1)

Here $D$ denotes the learnt dictionary, $X$ the (sparse) representation vectors, and $Y$ are the training set images. The matrix $WS$ denotes the Wavelet synthesis operator.
(inverse Wavelet), or equivalently the Wavelet atom dictionary. This would give us an “equivalent” dictionary $D_{eq} = W_SD$ combining multi-scale Wavelet synthesis dictionary $W_S$ with a learnt dictionary $D$. This model suggests that the data can be expressed by a sparse combination of atoms, which are themselves combinations of atoms from a fixed multi-scale base dictionary, e.g Wavelets. While this dictionary is both multi-scale and adapted to the data, this problem is however intractable, in general, for reasonably sized data, without additional constraints or assumptions on the unknown dictionary $D$.

If each data sample $x_i$ is an image, then atoms in $D$ are also image sized, and thus too large for us to tackle directly.

A more workable formulation sparsely codes small groups of Wavelet coefficient (as opposed to the whole signal), and then concatenates the results prior to applying the inverse Wavelet. While the dictionary is now composed of reasonably sized atoms, training it is still prohibitively expensive as we are measuring the affect of any update to the coefficients on the whole image.

This problem can be solved by switching to the analysis domain of the Wavelet transform by applying the forward Wavelet operator to the images $Y$. The analysis and synthesis formulations are equivalent when the transform used in unitary (i.e. orthogonal Wavelets). Now we can train a dictionary using a dictionary learning algorithm working directly on small blocks of coefficients in the analysis domain of a multi-scale decomposition operator, specifically the Wavelet transform.

A natural way to view the Wavelet analysis domain is not as a single vector of coefficients, but rather as a collection of coefficient “images” or bands. The different Wavelet coefficient images contain data at different scales and orientations (horizontal, vertical and diagonal). As such it makes sense that separate dictionaries be used to represent these images. We achieve this by training our dictionary in parts, training separate sub-dictionaries $D_b$ for each Wavelet band (or group of bands).

This approach makes sense from another perspective as well – the coefficient images themselves have local 2-D structure – adjacent Wavelet coefficients tend to be correlated. Capturing this structure is the essence of the learning process.

In contrast to patch-based approaches in the image domain that emphasize only the local correlation between pixels, in this approach even a small patch in the deeper decomposition levels affects a large area in the image domain. The “effective” atoms (i.e. in the image domain) in our dictionary are interpolated versions of the atoms in the learnt dictionary $D$, interpolated by the Wavelet synthesis process. This allows our approach to have a more global outlook, as well as a local one.

While simple, this approach allows for many options and flexibilities to choose from, depending on the application. Some of these include:

- The choice of the Wavelet and the depth of the Wavelet decomposition.
- The size of the sub-dictionaries (number of atoms and their size). These may have different sizes at different scales and orientations.
• Training sub-dictionaries for groups of bands (rather than treating them separately). Grouping can be done according to scale or orientation.

• Dictionaries of multi-band atoms jointly coding coefficients in several bands.

• Replacing the orthogonal Wavelets with a different multi-scale transform, be it bi-orthogonal Wavelets, Curvelets, Contourlets etc. While using a non-orthogonal transform may complicate matters, this may be warranted in some scenarios.

Every sparse coding algorithm requires a multiplication by the dictionary and its adjoint, as part of the numerical process of computing the representation. A major problem with explicit multi-scale dictionaries, limiting their usefulness, is the prohibitively high cost of applying them for sparse coding. Atoms with large support simply require too many operations to be effective. This is where the proposed approach shows its advantage.

The two most prevalent approaches to sparse coding are greedy methods (such as OMP) and relaxation based methods. Our approach yields itself in a simple manner to both.

Greedy sparse coding algorithms can be applied in the analysis domain of a multi-scale transform, at the same cost as when using a single-scale dictionary. The stopping criterion can be calculated locally per patch, independently of the other patches, or using a more global look. In the global approach a sparsity (or error) constraint is applied to the whole image representation. Operations remain, however, local (and thus efficient), with patches essentially “competing” for additional coefficients, until the global criteria is met. All operations and comparisons can be done purely in the analysis domain, without need to apply the synthesis operator. This is possible since the multi-scale transform we use is orthogonal and the patches in the same band are non-overlapping. We note that while use of non-overlapping patches in the image domain would inevitably result in block artifacts, this is not the case in the multi-scale framework.

As for the relaxation based approach, the obtained optimization task can be solved numerically by algorithms such as iterative shrinkage. There are two operations that need to be computed efficiently in order for this process to be feasible – the multiplication by the dictionary $D_{eq}$ and multiplication by it’s transpose $D_{eq}^T$. Both these operations are simple to implement in the proposed framework. In our experiments we use both the greedy and the relaxation methods, and demonstrate their effectiveness.

We utilize this multi-scale dictionary learning structure in a series of experiments. We use it as a sparsifying dictionary in a compressed sensing scenario, outperforming stand alone Wavelets and a combination of Wavelets and DCT. We show how K-SVD on Wavelets compares with single scale K-SVD and regular Wavelets in m-term representation of images, achieving high quality reconstruction with a fraction of the number of non-zero coefficients. We show how the K-SVD on Wavelets can be used in an image denoising scenario and how, by combining it with single-scale K-SVD denoising
we can achieve near state-of-the-art results (in terms of PSNR) but with much less visible artifacts.

The K-SVD on Wavelets algorithm was reported in [OLE11] and the joint denoising algorithm in [SOE14].

1.4 Online Sparse Dictionary Learning

In this work we chose a different approach to multi-scale dictionary learning - learning a dictionary to sparsely code large image blocks. This required two components:

1. A model that balances the expressiveness required to represent large blocks with (multi-scale) structure that would allow effective training.

2. A learning paradigm that can handle the very large data-sets needed to train large blocks.

The model we chose to work with is the double-sparsity model [RZE10]. In this model the dictionary is composed of a multiplication of a fixed part (a.k.a. the base dictionary) by a sparse learned part. Each effective atom is thus a sparse combination of base dictionary atoms. This sparsity keeps the degrees of freedom of the model in check.

In [RZE10] this model is trained using a variant of the K-SVD algorithm (coined Sparse K-SVD) to sparsely code 2-D and 3-D image patches. The base dictionary used was the (over-complete) Discrete Cosine Transform (DCT). Unfortunately, neither this base dictionary nor the training paradigm are suitable for large image blocks. The DCT has been shown to be a very good model for small image patches, but this is not true for large ones. The K-SVD like training procedure is unsuitable for very large data-sets. When the data-set is very large taking a batch approach is infeasible. We simply cannot sweep through all training samples in each iteration. In addition, the K-SVD atom update step is impossible in this scenario.

We offer a solution to these problems - a novel multi-scale base dictionary and an online training paradigm.

**Cropped Wavelets:** The authors of [RZE10] suggested the use of Wavelets as a base dictionary, however standard Wavelets are not a good choice for representing image blocks. The reason for this is the Wavelets limitation in representing image boundaries. In standard Wavelets, finite length signals are extended (periodically/symmetrically/zero padded) when applying the Wavelet transform. This inevitably creates a discontinuity (of some degree) at the signal boundary. Coding this discontinuity (and the extension) “costs” us coefficients. When applying the transform to a whole image this cost may be negligible, but it becomes prohibitive when dealing with patches (even large ones).

We offer a novel Wavelet-like dictionary we call Cropped Wavelets. The basic idea is to look at Wavelet atoms taken from a larger domain, cropped to the smaller interval.
When comparing to a Wavelet dictionary on the original interval, we see that atoms near the center of the original interval remain unchanged, but we have added to the dictionary all atoms that overlap the original boundaries. These new atoms are cropped to the original interval and normalized. This construction gives us a redundant Wavelet-like dictionary with atoms that specialize in representing the signal near the boundaries. Experiments on synthetic and natural signals show that Cropped Wavelets no longer suffer from the boundary issue and give a sparser representation of the signals.

Another deviation we take from standard Wavelets is in the separability of the transform when applied to multi-dimensional signals. The standard Wavelet construction is separable only per decomposition level. The transform as a whole is not separable. A rarely used alternative is a fully separable construction – applying all decomposition levels in turn to each dimension. This construction is superior not only in speed, but surprisingly also gives better (sparser) results for natural 2-D images. The combined Separable Cropped Wavelet dictionary gives superior results both in terms of speed and in sparse representation of image blocks.

**Online Sparse Dictionary Learning (OSDL):** As noted earlier, a K-SVD like algorithm cannot handle vast data-sets. Instead we turn to online approaches. We adopt a stochastic gradient descent (SGD) approach to the sparse dictionary learning.

Taking a block coordinate descent approach we alternate between sparse coding and updating the dictionary. While the sparse coding remains unchanged (OMP) we use a gradient descent approach to training the adaptable sparse part of the dictionary. Taking a gradient descent step on its own would give us non-sparse atoms Therefore this step is followed by a hard-thresholding stage operating on the columns of the sparse matrix.

This procedure can now be plugged into a stochastic gradient descent where gradient steps are taken each iteration with gradient estimates based on a randomly chosen example. These estimates tend to be very noisy hindering the optimization process. A common approach to alleviate this problem is to estimate the gradient on groups of examples (batches). Using relatively small batches also allows us to reduce computational costs since the sparse coding of these signals can be done using efficient batch-OMP [RZE08].

Several other improvements to the SGD process can also be implemented to improve performance such as an adaptive gradient step size and adding a momentum term. Standard dictionary learning techniques such as pruning unused or similar atoms also aid the process.

In a series of experiments we show how Separable Cropped Wavelets outperform standard Wavelets in image representation, and its beneficial use as a multi-scale base dictionary in the double-sparse model. We show that the OSDL algorithm outperforms Sparse K-SVD in terms of both speed and dictionary performance. More importantly, OSDL scales up nicely to scenarios where Sparse K-SVD is no longer applicable. We show preliminary results for how the OSDL approach can be used in applications such as
image compression and denoising as well as for training a global model for representing natural images.

The Cropped Wavelets and Online Sparse Dictionary Learning algorithm were reported in [SOZE16].

1.5 Thesis Structure

The structure of this dissertation is as follows. In Chapter 2 we give some background on sparse representations, dictionary learning and analytic dictionaries as well as a brief survey of previous works on multi-scale dictionary learning. In Chapter 3 we present the K-SVD on Wavelets approach followed by the fused multi-and-single scale image denoising in Chapter 4. In Chapter 5 we present Cropped Wavelets and in Chapter 6 the OSDL algorithm.
Chapter 2

Background

2.1 Sparse Representations

Sparse representations of signals over redundant dictionaries is an evolving field with state of the art results in many signal and image processing tasks. The basic assumption of this model is that natural signals can be expressed as a sparse linear combination of atom signals. Thus two main problems need to be addressed:

- How to represent a signal in the sparsest way, for a given dictionary.
- How to train a dictionary as to allow sparse coding for a specific class of signals.

2.1.1 Sparse Coding

The problem of sparse coding can be formally described by:

\[
\min_x \|x\|_0^0 \quad \text{subject to} \quad y = Dx. \tag{2.1}
\]

where \(y\) is the input signal to be represented, \(x\) is the representation vector and \(D\) is the dictionary of signal atoms.

This problem in itself is generally an NP-hard combinatorial problem. Approximate solutions can be found by a wide variety of algorithms. These algorithms come in two main flavors - greedy algorithms and convex relaxation based algorithms.

An approximation of the sparse coding problem, using an error constraint can be given by:

\[
\min_x \|x\|_0^0 \quad \text{subject to} \quad \|y - Dx\|_2^2 \leq \epsilon, \tag{2.2}
\]

Alternatively, we can impose a sparsity constraint by:

\[
\min_x \|y - Dx\|_2^2 \quad \text{subject to} \quad \|x\|_0^0 \leq K \tag{2.3}
\]

Greedy, a.k.a. “pursuit”, algorithms such as matching pursuit (MP) [MZ93] and
orthogonal matching pursuit (OMP) [PRK93, Tro04], add atoms to the signal representation one by one. The OMP algorithm is presented in Algorithm 2.1.

Algorithm 2.1 The Orthogonal Matching Pursuit algorithm.

**Parameters:** $y$ (signal in $\mathbb{R}^n$), $\tilde{D} \in \mathbb{R}^{n \times m}$ (dictionary), $K$ (target sparsity) or $\epsilon$ (target error)

**Initialization:** $\text{supp} = \emptyset$, $\text{res} = y$

**Loop:** Repeat until target is reached (either $K$ iterations or until norm of residual is lower than $\epsilon$

1. $s_{\text{new}} = \arg\max_s \| d_s^T \text{res} \|
2. $\text{supp} = \{ \text{supp}, s_{\text{new}} \}$
3. $x = D_{\text{supp}}^+ y$
4. $\text{res} = y - D_{\text{supp}} x$

Another popular alternative is to substitute this problem with a simpler, i.e. convex, one by replacing the $l^0$ norm with an $l^p$ norm with $p = 1$ or $p \leq 1$ as is done in the Basis Pursuit [CDS01] and FOCUSS [GR97] algorithms.

### 2.1.2 Learning Signal/Image Dictionaries

Learnt image dictionaries have been of much interest in recent years. Two prominent examples are the MOD algorithm [EAHH99] and the K-SVD algorithm [AEB06a,
where $\mathbf{Y} = [\mathbf{y}_1 \mathbf{y}_2 \ldots \mathbf{y}_N] \in \mathbb{R}^{n \times N}$ denote the set of training examples, $\mathbf{D}$ is the dictionary, and $\mathbf{X} \in \mathbb{R}^{m \times N}$ the sparse representation matrix ($\mathbf{x}_i$ are the columns of $\mathbf{X}$).

For a given (and necessarily low) signal dimension, such a dictionary can be trained using for example the K-SVD algorithm (Figure 2.2). The problem with this algorithm (and others similar to it) is that the atoms (columns of the dictionary $\mathbf{D}$) are of the same size as the signal they represent. While in theory this gives the learning algorithm maximal freedom to shape the atoms to describe different scales of the data, in practice, due to limited computational resources, this severely limits the size of the signals represented.

**Algorithm 2.2** The Single-scale K-SVD algorithm. The description assumes a fixed number of non-zeroes $T$ in every representation, which could be replaced with a fixed representation error.

**Parameters:** $m$ (number of atoms), $n$ (size of signals)

**Initialization:** Set the dictionary matrix $\hat{\mathbf{D}} \in \mathbb{R}^{n \times m}$ (using examples, or a pre-chosen matrix)

**Loop:** Repeat until convergence (or according to a stopping rule)

- **Sparse Coding:** Fix $\hat{\mathbf{D}}$ and use OMP to compute the representation vector $\mathbf{x}_i$ for each example $\mathbf{y}_i$ ($i = 1, 2, \ldots, N$) by solving:

  $$\min_{\mathbf{x}_i} \| \mathbf{y}_i - \mathbf{D}\mathbf{x}_i \|_2^2 \quad \text{subject to} \quad \| \mathbf{x}_i \|_0 \leq T$$

- **Dictionary Update:** update each atom $\mathbf{d}_k$ ($k = 1, 2, \ldots, m$) in turn by:
  - Select $\omega_k$, the group of examples that use atom $\mathbf{d}_k$.
  - For each example $j$ in $\omega_k$, compute its residual $\mathbf{e}_{j,k}$ without the contribution of $\mathbf{d}_k$.
  - Create residual matrix $\mathbf{E}_k$ as the matrix whose columns are $\mathbf{e}_{j,k}$.
  - Update the atom $\mathbf{d}_k$ and weights $x_{j,k}^k$ by minimizing:

    $$\left( \mathbf{d}_k, x_{j,k}^k \right) = \arg\min_{\mathbf{x}, \mathbf{d}} \| \mathbf{E}_k - \mathbf{d}\mathbf{x}^T \|_F^2 \quad \text{subject to} \quad \| \mathbf{d} \|_2 = 1$$

This one-rank approximation is performed by truncated SVD of $\mathbf{E}_k$.

Any attempt to increase the signal dimension $n$ beyond few hundreds immediately implies an intolerable amount of computations for the training phase, and an unrealistic size of training set to use. Even if the training has been done somehow, using such a dictionary in applications is prohibitive, as any multiplication by this dictionary (as an explicit matrix) leads to high-complexity algorithms. As said above, this is the prime reason for the popular patch-based processing that is commonly practiced with learned
dictionaries in recent years.

Using these algorithms for larger images, in practice, means breaking the input image into blocks and treating each block independently. When these blocks have large overlap the end result is a very redundant representation of the original image. While this representation may be very useful for purposes of denoising [EA06], it is counter intuitive to our goal of representing the complete signal with a sparse combination of atoms. On the other hand, small, or no, overlap of the blocks leads to boundary issues when reconstructing the image.

Dictionary learning algorithms that attempt to learn small, shift invariant, atoms from large signals also exist [Aha06, JLVG06, Jos07]. One such algorithm is the MoTIF algorithm [JLVG06, Jos07]. In this algorithm small atoms are learnt from larger training signals, however the main driving force is not sparsity of the representation. Instead, the algorithm forces the learnt atoms to be as different from one another as possible by penalizing correlation between them. This approach works well when the atoms are indeed very dissimilar, however this is not the case in most natural signals/images.

2.2 Multi-scale Transforms

Multi-scale analysis for images took a center stage in image processing since the 1980’s, starting with the Gaussian and Laplacian pyramids (first suggested by Burt and Adelson [BA83]), and the Gabor Transform. In the 1990’s Wavelets [Dau92] became the premier multi-scale analysis tool in signal processing. Being however better suited for single dimensional signals, their usefulness in image processing was limited. To overcome the Wavelet shortcomings in handling two dimensional signals, more advanced “Wavelet like” decompositions were developed starting in the late 1990’s and into the 2000’s. Several different families of multi-scale decompositions are available for us to consider:

1. Laplacian Pyramid [BA83, DV03],
2. Steerable Pyramid [SF95],
3. Standard Wavelet decompositions [Dau92],
4. Wavelet Packet decompositions [CMW92],
5. 2nd Generation Wavelet decompositions - Contourlets [DV05, Do01], Curvelets [CD99, CDDY06], Ridgelets [Can98], Shearlets [LLKW05], Bandlets [LM05] and more.

While all these decompositions create multi-scale representations of an image, they all suffer from the curse of generality. Designed to handle any and all images, these decompositions can not, and do not, handle any subset of images optimally.

Of the above decompositions, the Curvelet, Contourlet, Shearlets and Bandlets decompositions feature some sort of optimality for two dimensional signals. This
optimality is usually measured by the decay of the M-term approximation error, the distance $||y - y_M||^2$ between the signal $y$ and its approximation $y_M$, using the $M$ strongest representation coefficients. A low M-term approximation error would suggest that these representations are well suited for many image processing applications. However, even in these cases, optimality is shown only for specific classes of images, such as piece-wise smooth images, which do not necessarily reflect true image content.

That said, many useful applications have been found for these decompositions. A key feature that makes these transforms appealing is their tendency to sparsify specific image content. We shall build on this important property and target it directly by merging a learning procedure on top of an existing multi-scale transform. In that respect, our work follows the intuition of the Bandlets, as advocated by Stephane Mallat: Rather than seek the direct transform to get the ultimate sparsification of our signals, start by using an existing transform that does reasonably well in that respect, and then add another layer of processing that squeezes more over the already simplified signals.

2.2.1 Wavelets

Wavelets are perhaps the most widely used multi-scale transform in signal and image processing [Dau92, Mal08]. This is partially due to the fact that have been around for many years, with a deep theoretical analysis for both the continuous and discrete cases. A more important reason, in our eyes, is the relative simplicity of applying Wavelets to various signal processing tasks, with good (or at least adequate) results. This fact leads us to use Wavelets in the majority of our work, attempting to augment the Wavelets and with sparse representations.

Wavelets are typically defined by either a scaling filter or a Wavelet function. Orthogonal wavelets are typically defined by the scaling filter – a low-pass finite impulse response (FIR) filter of finite length. Analysis with orthogonal Wavelets require also a high pass filter which is calculated as the quadrature mirror filter of the low pass, and reconstruction filters that are time reversed versions of the decomposition filters. For bi-orthogonal Wavelets, separate decomposition and reconstruction filters are defined. Daubechies and Symlet Wavelets are examples of such Wavelets. Another option is to define the Wavelet by a pair of functions in the time domain, the Wavelet function $\Psi(t)$ (a.k.a. the mother Wavelet) and a scaling function $\Phi(t)$ (a.k.a. the father Wavelet). The combination of the functions creates a set of band-pass filters partitioning the frequency domain. For Wavelets with compact support, $\Phi(t)$ is finite in length making the two definitions equivalent. Meyer Wavelets are an example of Wavelets defined by scaling functions.

The standard way Wavelets analysis is applied involves a low/high frequency partition of the signal, followed by decimation (in each dimension for multi-dimension signals). This operation is then repeated on the low frequency band, a.k.a. approximation band, for further decomposition levels. Synthesis is applied in a mirrored process with
interpolation, filtering and summation. The process is described in Fig.2.2.

![Schematic 2 level Wavelets analysis (left) and synthesis (right).](image)

Figure 2.2: Schematic 2 level Wavelets analysis (left) and synthesis (right).

While other options are available - skipping the decimation stage (Undecimated Wavelets) or decomposing the high pass band as well (Wavelet Packets), for the most part we chose to use the standard analysis/synthesis procedure. While all Wavelet schemes give a multi-scale flavor, these options add a strong redundancy to the multi-scale transform, a trait we largely preferred to incorporate through the sparse representation parts of our algorithms.

### 2.3 Previous Works

The idea of learning multi-scale dictionaries is not new. We present here several works on the topic, related to our work, and discuss their limitations.

#### 2.3.1 Adapting Wavelets for Sparsity

In [OSL00, SO02, Sal04] the Wavelet pyramid analysis structure is maintained (thus achieving a multi-scale decomposition). The Wavelet parameters are trained so that the Wavelet coefficients conform with a sparsity inducing prior distribution. The results are Wavelet-like filters that give a slightly sparser representation for the training images.

While this approach tries to adapt the Wavelets to the data, the number of free parameters is fairly small (the coefficients of the filters) since the strict Wavelet decomposition structure is maintained. This means that the ability to adapt to complex data is limited.

#### 2.3.2 Semi Multi-Scale Learnt Dictionaries

In [MSE07, MSE08] the first steps are taken towards more general multi-scale learnt dictionaries. Using a Quadtree structure (Fig.2.4), different sized blocks are used to make up one joint dictionary. This dictionary is learnt and used in a similar manner as in the K-SVD algorithm. This structure imposed on the dictionary would seemingly put
this approach in a disadvantage to regular single-scale K-SVD (with the same number of atoms all at the largest size), as the totally unstructured approach has more degrees of freedom. What we see in practice is that constraining the dictionary (in a way that is relevant to the data) aids the highly non-convex learning process to reach a better result. In the case of images, this multi-scale structure proves to be more effective than a totally unstructured one.

Although different sized atoms are used, this method does not help us reach our goal of truly large atoms. The computational constraints limit the maximal atom size used, just as in the single-scale approach. The largest atom size is thus the same as that possible in the standard approach. However, we can see that even when dealing with small image patches, a multi-scale approach proves beneficial.

### 2.3.3 Sparse K-SVD

The works reported in [RZE10, YD09] could be interpreted as yet another way to train a multi-scale dictionary. These works suggests to form the effective learnt dictionary as a multiplication of a fixed and fast transform $B$ (we will refer to it as the core or base dictionary) by a sparse matrix $A$. Interestingly, a similar dictionary construction is used in [NZ02], but for a different purpose, of approximating a desired frame by combinations
over another base-dictionary. The dictionary learning process can be written as:

$$\arg\min_{A,X} \|Y - BAX\|_F^2 \text{ subject to } \begin{cases} \|x_i\|_0 \leq T, & \forall i \\ \|a_j\|_0 \leq P, \quad \|Ba_j\|_2 = 1 & \forall j. \end{cases} \quad (2.7)$$

The meaning of this structure is that every atom in the effective dictionary is a linear combination of few and arbitrary atoms from the base dictionary. In [RZE10] the learning procedure used is a variant of the K-SVD (termed Sparse-K-SVD), where the dictionary update stage amounts also to a series of sparse coding problems.

While the work in [RZE10] used this construction for working with somewhat larger patches using the redundant DCT as the base dictionary, one could envision applying the same algorithm with a Wavelet (or any other multi-scale transform) base dictionary, thereby leading to a highly structured and learnt multi-scale dictionary. We note that the work in [RZE10] did not address such an option, and in particular did not consider the numerical complexities that such large matrices give rise to. More on this in Chapter 6.
2.3.4 Structured/Constrained Dictionaries

Over the years different types of constrains have been proposed for dictionaries/atoms, imposing a certain structure on them. All these works limit the degrees of freedom of the dictionary or impose a desired property.

One such property that may be desired is shift invariance. Dictionary learning algorithms that attempt to learn small, shift invariant, atoms from large signals include [Aha06, JLVG06, Jos07]. In the MoTIF algorithm [JLVG06, Jos07] small atoms are learnt from larger training signals, however the main driving force is not sparsity of the representation. Instead, the algorithm forces the learnt atoms to be as different from one another as possible by penalizing correlation between them. This approach works well when the atoms are indeed very dissimilar, however this is not the case in most natural signals/images, where we expect many atoms to have a relatively high correlation with each other. Most MoTIF atoms will thus be “difference” atoms that are only relevant when used in conjunction with a “base” atom during coding.

A different approach is taken in [AE08, BMBP11]. In these works a dictionary is trained which is an image in itself. This translates into a lower degree of freedom in the dictionary and (close to) shift invariant atoms.

Explicit (unconstrained) dictionaries are computationally expensive to apply. In this context, some efforts have been made to design fast dictionaries that can be both applied and learned efficiently. This requirement implies constraining the degrees of freedom of the explicit matrix in a way that would speed up application of the dictionary (and also its transpose). One such possibility is the search for adaptable separable dictionaries, as in [HSK13]. The works in [RZE10, YD09] (presented above) where the dictionary is a multiplication of two structured matrices, also fall in this category. The work in
[LMG15] takes this idea a step further by composing a dictionary from the multiplication of a sequence of sparse matrices. In the work reported in [CMTD14] the dictionary is modeled as a collection of convolutions with sparse kernels, lowering the complexity of the problem and enabling the approximation of popular analytically-defined atoms.

2.3.5 Conclusion

All the above-mentioned attempts (and others) to tie multi-scale representations to dictionary learning are innovative, but unfortunately they all lead to marginal improvements over existing non-multi-scale methods. This may be explained by the heavily constrained structures these methods force, the limitations of the learning paradigm and other factors. In our work we build on these approaches, attempting to achieve a truly multi-scale learning paradigm and dictionary.
Chapter 3

K-SVD on Wavelets

3.1 Introduction

A fundamental question when using the sparse representation model is the choice of
dictionary to be used [RBE10]. Most approaches to this problem can be divided into
one of two categories: the analytic approach and the learning based approach. In
the analytic approach a mathematical model of the data is formulated, leading to an
implicit dictionary described by it’s algorithm rather than by an explicit matrix. These
dictionaries include the Fourier transform, the DCT, Hadamard, Wavelets, Curvelets
and Countourlets among others. The dictionaries generated by these approaches are
highly structured and have fast implementation. A common theme among many of
these methods is a multi-scale approach to signal representation.

In contrast, the second approach infers the dictionary from a set of training examples.
The dictionaries learnt are typically represented as explicit matrices. Dictionary learning
algorithms range from the well-known and simple PCA, through the seminal work by
Olshausen and Field [OF97], the MOD [EAHH99] and K-SVD [AEB06a] follow-up
methods, and all the way to generalizations of the PCA (e.g. GPCA [VMS05]). All
these methods target the same goal – finding a direct sparsifying transform [RBE10].
This approach yields dictionaries more finely fitted to the data, thus producing better
performance in many applications. However, this comes at a price of unstructured
dictionaries, which are more costly to apply. Complexity constraints limit such learnt
dictionaries and specifically the atom size that can be learnt. This constraint is the
reason why low-dimensional (a typical dimension is of the order of 100) patch-based
processing is so often practiced when using such dictionaries.

In this chapter we present an attempt to merge the two approaches described above
to create a truly multi-scale learnt dictionary, hopefully gaining the advantages of both
methods. In a nut-shell, we propose training the dictionary, in parts, over the analysis
range of an analytic multi-scale transform, applied to the training set.

This approach bares some similarity to the work in [RZE10], as we too create atoms
by combining base Wavelet atoms. However, our construction forces a close spatial
proximity between the combined base-atoms, leading to a more constrained structure that directly targets the spatial redundancy that Wavelet coefficients tend to exhibit when handling images.

3.2 K-SVD in the Wavelet Domain

3.2.1 The Core Approach

The Wavelet transform gives a sparse representation of the original signal to some degree. What we would like to do is squeeze out some of the redundancy left by the Wavelet decomposition, specifically the spatial correlation between Wavelet coefficients in the same band, or between bands, thus producing sparser image representations than plain Wavelet decompositions.

Following the work reported in [RZE10], we begin our derivation by looking at the learning problem expressed by the following modification to Equation (2.4):

$$\arg\min_{D,X} \|Y - W_DX\|_F^2 \text{ subject to } \|x_i\|_0^0 \leq T \quad \forall i.$$  

(3.1)

Here $D$ denotes the learnt dictionary, $X$ the (sparse) representation vectors, and $Y$ are the training set images. The matrix $W_S$ denotes the Wavelet synthesis operator (inverse Wavelet), or equivalently the Wavelet atom dictionary. This model suggests that the data can be expressed by a sparse combination of atoms, which are themselves combinations of atoms from a fixed multi-scale base dictionary, e.g. Wavelet. This problem is however intractable, in general, for reasonably sized data, without additional constraints or assumptions on the unknown $D$. In [RZE10], the assumption chosen is that $D$ has very sparse columns. This implies that the overall dictionary atoms are linear combination of few (and arbitrary) Wavelet atoms.

Assuming that $W_S$ is square and unitary (i.e. orthogonal Wavelet with periodic extension), we can equivalently write:

$$\arg\min_{D,X} \|W_AY - DX\|_F^2 \text{ subject to } \|x_i\|_0^0 \leq T \quad \forall i,$$  

(3.2)

where $W_A$ denotes the Wavelet analysis operator. This formulation suggests that we can train our dictionary not in the image domain but in the analysis domain of the multi-scale decomposition operator, specifically the Wavelet transform.

A natural way to view the Wavelet analysis domain is not as a single vector of coefficients, but rather as a collection of coefficient “images” or bands. The different Wavelet coefficient images contain data at different scales and orientations (horizontal, vertical and diagonal). As such it makes sense that separate dictionaries be used to represent these images. We achieve this by training our dictionary in parts, training separate
sub-dictionaries $D_b$ for each Wavelet band (or group of bands):

$$\forall b \arg\min_{D_b, X_b} \| (W_A Y)_b - D_b X_b \|_F^2 \ \text{subject to} \ \| x_{i,b} \|_0 \leq T \ \forall i,$$

(3.3)

where subscript $b$ denotes the different Wavelet coefficient bands.

While the learning process can be applied as is on small images, this is computationally impossible with larger images (the coefficient band images for the first decomposition level are for instance one forth the size of the original image). We solve this problem by returning to the patch-based approach, prevalent in many image processing methods.

This approach makes sense from another perspective as well – the coefficient images themselves have local structure - adjacent Wavelet coefficients tend to be correlated. Figure 3.1 demonstrates this. Capturing this structure is the essence of the learning process.

![Figure 3.1: Several bands taken from the Wavelet transform of an arbitrary image. As can be seen, there is a large amount of correlation between the Wavelet coefficients, suggesting that more can be done to sparsely represent the image content.](image)

In contrast to patch-based approaches in the image domain that emphasize only the local correlation between pixels, in this approach even a small patch in the deeper
Parameters: The following parameters should be chosen:

- Wavelet type to use,
- \( S \) – number of decomposition levels (scales),
- \( K \) – number of atoms per dictionary, and
- \( n \) – size of the dictionaries’ atoms.

Initialization: Set the dictionary matrices for all bands, \( \hat{D}_b \in \mathbb{R}^{n \times K} \), \( b = 1, 2, \ldots, 3S + 1 \).

Wavelet Decomposition: Decompose each of the training-set images using the chosen 2D-Wavelet transform, each into \( 3S + 1 \) bands.

For each band:

- **Extract Patches:** Extract maximally overlapping patches of size \( \sqrt{n} \times \sqrt{n} \) from the same band of all training set decompositions.
- **K-SVD:** Apply K-SVD separately for each decomposition band to train the sub-dictionary \( \hat{D}_b \). This process should be repeated \( 3S + 1 \) times, once per each band.

Algorithm Output: The set \( \hat{D}_b \in \mathbb{R}^{n \times K} \), \( b = 1, 2, \ldots, 3S + 1 \), combined with the Wavelet transform used, define the effective multiscale dictionary learned.

Figure 3.2: The proposed multi-scaled dictionary learning – K-SVD applied to each band in the Wavelet domain

decomposition levels affects a large area in the image domain. This allows our approach to have a more global, as well as local, outlook. The complete learning algorithm is described in Figure 3.2.

In effect, the effective dictionary we created is \( W_s \hat{D} \), replacing the standard Wavelet dictionary \( W_s \). The “effective” atoms are thus interpolated versions of the atoms in the learnt dictionary \( \hat{D} \), interpolated by the Wavelet synthesis process. This dictionary enjoys the multi-scale capabilities of the Wavelet transform while adding to it information specific to the training domain. Note that in the training we use maximally overlapping patches. This creates a “richness” in the training data that generates a level of shift-invariance in the resulting dictionary. Some effective atoms from different scales and bands, obtained by training on a corpus of fingerprint images, are presented in Figure 3.3. In order to visualize a single effective atom, all the coefficients (for all patches in all bands), except one, are set to zero. Multiplying such a coefficient set by the learnt dictionary and passing the result through a Wavelet synthesis process gives us a visualization of a single “effective” atom in the image domain. We can thus verify that our atoms are indeed multi-scale, localized and adapted to the training data.
Figure 3.3: Visualization of some effective atoms from different levels/bands trained on fingerprint images using a 3 level Haar Wavelet transform. A separate sub-dictionary was trained for each band. Top row - atoms from the approximation band. 2nd row - atoms from the coarsest level horizontal and vertical bands and so on.

While the idea presented here seems quite simple – applying dictionary learning in the transform domain – it leads to an elegant way of creating a truly multi-scale dictionary, while still retaining reasonable computational cost for both learning the dictionary and using it in practice. One of the most appealing aspects of this structure is the ease with which sparse coding is performed. Before we turn to describe this, we first discuss several flexibilities and options that could be incorporated into the above scheme.

### 3.2.2 Options and Flexibilities in the Proposed Scheme

In the above algorithm, the degrees of freedom are the choice of the Wavelet filters, the depth of the Wavelet decomposition and the size of the sub-dictionaries (number of atoms and their size). An obvious extension would be to allow the sub-dictionaries to have different sizes at different scales and orientations. This may be warranted for specific data types where the variability needed to be expressed by the dictionary is high in some bands (requiring a higher level of redundancy in the dictionary to allow for sparse coding) while it is low in others.

Other options arise from the fact that in the above algorithm, each Wavelet decomposition level and each directional band is treated separately and has it’s own sub-dictionary. This, of course, in not mandatory. Two additional options may have
merit:

1. Grouping by decomposition level - a single sub-dictionary is trained for all bands at the same Wavelet decomposition level. The rational is that each such sub-dictionary will express all the data at the same scale.

2. Grouping by directional band - a *single* sub-dictionary could be trained for all bands of the same *orientation* in all different scales. The rational for this approach is that the directional features are self similar at different scales, and this should be exploited in our construction. In a way, this directly extends what analytic multi-scale transform are naturally doing.

An advantage of both these approaches is that they increase the amount of data available for training of each sub-dictionary. In some scenarios, there is very little data to train on, especially at the lower decomposition levels. While these options affect which data is used to train each sub-dictionary, and which sub-dictionary is associated with each band, the atoms themselves still represent two dimensional patches in the Wavelet coefficient domain.

A different direction we can take is departing from the two dimensional patch approach, and create multi-band atoms (Fig. 3.4). These atoms are created by concatenating two dimensional patches from different decomposition bands in such a way that all the 2D patches are mapped to the same location in the image. We offer two options for this construction:

1. Grouping by decomposition level - group same-sized patches from all three bands at each decomposition level. Thus, the sub-dictionary could be trained on 3D patches of size $\sqrt{n} \times \sqrt{n} \times 3$, containing three $\sqrt{n} \times \sqrt{n}$ matched patches merged together. The rational is that a correlation exists, not only spatially within each band, but also between “brother”-bands at the same scale, and this should be utilized.

2. Grouping by directionality - group patches from all bands with the same direction. For orthogonal Wavelets we have three directions - horizontal, vertical and diagonal, but for redundant Wavelets (or other transforms such as Contourlets there may be more). Each sub-dictionary could be trained on pyramidal patches, containing matched patches of sizes $\sqrt{n} \times \sqrt{n}$ (for the course level), $2\sqrt{n} \times 2\sqrt{n}$, $4\sqrt{n} \times 4\sqrt{n}$ … The rational for this approach is that sharp image features (edges), are composed of many frequency components. The multi-scale transform actually partitions the same edge into multiple bands. The pyramidal atom merges the partitions back together.

Another option available to us is to replace the standard orthogonal Wavelet transform (with periodic extension) by non-unitary transforms such as bi-orthogonal Wavelet
and more advanced transforms such as Contourlets or Curvelets. While these decompositions generally give a sparser representation of the data, especially for images, there is still redundancy in the representation that can be reduced by using our scheme. However, losing the unitary property means that Equation (3.1) and Equation (3.2) are no longer equivalent, and this will affect the way the sparse-coding should be performed. In this work we will restrict our study to unitary Wavelet transforms, leaving the more general transforms for a future work.

3.2.3 Sparse Coding

Assuming that we have been able to train a multi-scale dictionary somehow, one of the main issues we face is using it in applications. Every sparse coding algorithm requires a multiplication by the dictionary and its adjoint, as part of the numerical process of computing the representation. A major problem with explicit multi-scale dictionaries, limiting their usefulness, is the prohibitively high cost of applying them for sparse coding. Atoms with large support simply require too many operations to be effective. This is where the proposed multi-scale dictionary approach shows it’s advantage.

The sparse representation of an image $y$ with respect to a dictionary $D$ is the solution $x$ of the problem

$$
(P_{0,\epsilon}) \min_x \|x\|_0^0 \text{ subject to } \|y - Dx\|_2 \leq \epsilon,
$$

(3.4)

where we aim at getting the sparsest representation that explains $y$ as $Dx$ with an error that is at most $\epsilon$. In our approach we replace this problem by

$$
(P_{0,\epsilon})_{synth} \min_x \|x\|_0^0 \text{ subject to } \|y - W_S D x\|_2 \leq \epsilon.
$$

(3.5)
in the synthesis domain, or equivalently,

\[(P_{0,w}^W)_{\text{analysis}} \min_x \|x\|_0^0 \text{ subject to } \|W_A y - Dx\|_2 \leq \epsilon, \quad (3.6)\]

in the analysis domain.

All these problems are NP hard in general, and approximate solutions can be found using various methods. The two most prevalent approaches in sparse coding are greedy methods (such as OMP) and relaxation based methods. The proposed dictionary that combines Wavelets and K-SVD can easily be used in both.

In the greedy methods, signals are coded by adding coefficients one at a time until a stopping criterion is met. This is typically done in the signal domain. We propose instead to first apply the multi-scale decomposition \(W_A\) to the image. Then, patches (atom sized) in the analysis domain can be coded using the appropriate sub-dictionary, by using the sparse coding greedy algorithm of choice. These operations are done using the small atoms in the sub-dictionary, at the same cost as when using a single-scale dictionary.

The stopping criterion can be calculated locally per patch, independently of the other patches, or using a more global look. In such a local approach, the \(k\)-th patch from the band \(b\), denoted by \([W_A y]_{b, k}\), will be allocated a fixed number of non-zero coefficients \(\ell\) (or a per-patch noise threshold), based on

\[(P_{0,w}^W)^k_{b} \min_x \|W_A y_{b, k} - Dx_{b, k}\|_2^2 \text{ subject to } \|x_{b, k}\|_0^0 \leq \ell, \quad (3.7)\]

and this should be repeated for all patches \(k\) from all bands \(b\).

A global counterpart approach can also be proposed, where patches are coded in conjunction with the coding of the other patches. In this case, we seek the sparse representation vector \(x\) that solves

\[(P_{0,w}^W)^{\text{global}} \min_x \sum_b \sum_k \|W_A y_{b, k} - Dx_{b, k}\|_2^2 \text{ subject to } \sum_b \sum_k \|x_{b, k}\|_0^0 \leq \ell, \quad (3.8)\]

Patches will essentially "compete" for additional coefficients, until a global criteria is met (either a fixed number of non-zero coefficients coding the whole image or a global noise threshold).

The process can be viewed as if we have competing local pursuits running on all the patches, together acting as a sort of global pursuit. At each step we will compare the gain (in terms of the residual energy) obtained by "activating" an additional coefficient for each patch, thus allowing the local pursuit to add another coefficient to the patch’s representation. In other words, we let each local pursuit show us what can be gained by letting it code with one more non-zero coefficient, and then choose the best one. We are thus considering a set of size \(|\sum_b \sum_k 1|\) of possibilities (i.e., the number of overall patches in the complete Wavelet domain). The patch that gives maximum benefit
will then have an additional coefficient allocated to it. Since the multi-scale transform is energy preserving all these operations and comparisons can be done purely in the analysis domain, without need to apply the synthesis operator.

Once an atom has been chosen to be added to a certain patch, a least-squares step is required to update the patch’s representation (in OMP), followed by updating the residual. Since the patches in the same band are non-overlapping and the Wavelet transform is orthogonal (so one band’s representation does not affect the others), these steps are all local. In addition, when moving to the next iteration of the algorithm for choosing the next atom, one need not reevaluate all the $|\sum_b \sum_k 1|$ inner-products, but only update the last patch’s gain (how that patch will reduce the residual by having another coefficient allocated to it).

While the algorithm described above is an extension to pursuit algorithms that add coefficients one at a time, such as OMP, our scheme can also be similarly adapted to pursuit methods such as CoSaMP [NT08] and Subspace Pursuit [DM09] that activate coefficients in groups.

As for the relaxation based approach, the obtained optimization task can be solved numerically by algorithms such as iterative shrinkage. There are two operations that need to be computed efficiently in order for this process to be feasible – the multiplication by the dictionary $D_{eq}$ and multiplication by it’s transpose $D_{eq}^T$. Multiplication by the complete equivalent dictionary $D_{eq}$ is simply done by multiplying each explicit subdictionary by the appropriate (sub)-coefficient vector, aggregating the results for all patches $k$ and bands $b$, and applying the synthesis multi-scale operator (i.e. the inverse Wavelet transform)

$$\hat{y} = D_{eq}x = W_s \left( \bigcup_b \bigcup_k D_b x_k^b \right).$$  \hspace{1cm} (3.9)

Multiplication by the transpose $D_{eq}^T$ is also a simple procedure. For unitary transforms this operation is the inverse transform. For orthogonal Wavelets this is simply the forward (analysis) Wavelet transform. We note that for non-orthogonal Wavelets the transpose of the synthesis operation can still be computed efficiently by using the Wavelet analysis operator. In this case the synthesis and analysis filters are no longer equal, and thus the analysis operator is applied with the synthesis filters instead of the standard analysis ones. Multiplication by $D_{eq}^T$ is thus computed by applying the (forward) Wavelet transform to the signal followed by breaking each Wavelet coefficient image into atom sized blocks and multiplying each block by the appropriate (explicit) sub-dictionary transposed.

In the experiments that follow we shall use both the greedy and the relaxation methods, and demonstrate their effectiveness.
3.3 Experiments

In the following experiments we aim to demonstrate the advantages that our multi-scale learnt dictionary has, compared to standard multi-scale representations and compared to single-scale patch processing in the image domain. We show that by replacing standard Wavelet dictionaries with learnt ones, based on the same multi-scale structure described above, we get an improved representation. We also show that compared to the single-scale patch processing in the image domain, we obtain a more compact representation.

3.3.1 M-term Approximation

To demonstrate the potential of our scheme to various image processing applications we start by looking at the M-term approximation error, as a measure of how well our dictionary describes the particular features of a set of images.

An input image representation, using a trained dictionary, can be created by the following steps:

- Apply the Wavelet transform to the image.
- Each coefficient band (at each level) should be broken into non-overlapping blocks.
- Using the global approach and local OMP coding, a sparse representation of the transformed image is found with $L$ non-zeros.
- Each band’s representation is thus an $M \times N$ sparse matrix, $M$ being the number of atoms in the appropriate sub-dictionary and $N$ being the number of non-overlapping blocks in the band.
- The total image representation is the collection of the representations of all the bands.

From this representation the image can be then reconstructed by:

- Multiplying each block’s representation vector by the appropriate sub-dictionary.
- Reconstructing the Wavelet coefficient images for each band and level by tiling the non-overlapping blocks.
- Applying the inverse Wavelet transform.

We trained dictionaries on two sets of images with different characteristics, and using different Wavelets commonly used in various image processing tasks. We trained one dictionary on 50 fingerprint images (Figure 3.5 presents three of them), and using a 2D separable 16-tap Symlet Wavelet transform (Matlab 'sym8') three layers deep. A second dictionary was trained on 20 coastal scenery images [Nat] (Figure 3.6 presents three of them), using Daubachies 8-tap Wavelets (Matlab 'db4') also three layers deep. These
transforms produce ten coefficient bands from each input image: horizontal, vertical, and diagonal coefficient images for each level, plus an approximation image at the last level. A separate sub-dictionary was trained for each band, 10 sub-dictionaries in total. Each sub-dictionary is a $64 \times 64$ matrix, i.e., 64 atoms, each of size $8 \times 8$. Using a fixed sparsity constraint ($\sim 5$) or a fixed error constraint ($\varepsilon \sim 1$) during the training worked equally well. The training samples used were maximally overlapped blocks extracted from the training set Wavelet coefficient images. Since the amount of training data in some cases exceeded the memory capacity available to us, low energy samples were removed from the training set. This approach was chosen (as opposed to a regular sub-sampling scheme) as the relative sparsity of the Wavelet coefficient images produces many near-zero training patches.

![Figure 3.5: Training Images - fingerprint data set.](image)

![Figure 3.6: Training Images - coastal scenery data set.](image)

We compare our reconstruction to a Wavelet clipping scheme, where small Wavelet coefficients are nullified, and to single-scale K-SVD, where non-overlapping image patches are coded using a dictionary trained on the same input images.

For both data sets, our reconstruction gives, for the same number of active coefficients, significantly higher image quality at low bit-rates (see Figure 3.7 for an example from the fingerprint data set, and 3.8 for an example from the coastal scenery data set). In Figure 3.9 we show PSNR as a function of the number of active (non-zero) coefficients for all three schemes. The results shown are the average PSNR over 15 test images for the fingerprints and 10 images for the coastal scenery images.

At the low end of the graph, we can see that our reconstruction continues to give good quality images even with a very low number of coefficients. At these levels, both
Figure 3.7: M-Term approximation of fingerprint image using 3 level sym8 Wavelet. From left to right - input image, Wavelet reconstruction (PSNR=20.68dB), Single-scale K-SVD reconstruction (PSNR=20.44dB), K-SVD on Wavelet reconstruction (PSNR=26.20dB). All reconstructions have 3300 active coefficients for an $448 \times 448$ image.

Figure 3.8: M-Term approximation of coastal scenery image using 3 level db4 Wavelet. From left to right - input image, Wavelet reconstruction (PSNR=28.36dB), Single-scale K-SVD reconstruction (PSNR=27.82dB), K-SVD on Wavelet reconstruction (PSNR=30.43dB). All reconstructions have 32000 active coefficients for an $1152 \times 1728$ image (only a $400 \times 400$ segment of the images is shown).

Figure 3.9: M-Term approximation - PSNR as function of number of active coefficients. Left - fingerprint images (averaged over 15 test images). Right - coastal scenery images (averaged over 10 test images). For fingerprint images a 3-level 'sym8' Wavelet is used, and for coastal scenery images a 3-level 'db4' Wavelet. All dictionaries are $64 \times 64$. 

32
Wavelet clipping and single-scale K-SVD distort the image beyond recognition. The effect is much more prominent in the scenery images which exhibit true multi-scale, compared to the fingerprint images which are mostly single scale. At the high end of the graph the Wavelet clipping gives better quality. Both these results stem from the fact that we train the dictionary specifically to represent the signal well using few active coefficients. Thus for a very sparse representation our scheme performs better. When we test for a much denser representation, our scheme seems inferior. However, our construction is still valid in this case, except that the dictionary needs to be trained for a denser representation to begin with.

We note that these comparisons are fair one since both our representation, the Wavelet representation and the single-scale K-SVD representation are all non-redundant. In this experiment our scheme does not add any redundancy to the signal representation. The Wavelet decomposition, the square learnt dictionaries and the proposed combination of both elements are all unitary transforms. Our scheme has the advantage of having more flexibility and more degrees of freedom since we train separate sub-dictionaries for each band, but overall the transform remains non-redundant. Needless to say, these tests were all done on images outside the training set.

3.3.2 Compressed Sensing

In the compressed sensing scenario [Don06] a signal $f$ is sampled by a linear measurement process that computes inner products between the signal and a collection of random vectors (whose number is significantly smaller than the signal dimension),

$$y = \Phi f.$$  \hspace{1cm} (3.10)

Assuming the signal $f$ can be expressed as a sparse combination $x$ of atoms from a sparsifying dictionary $D$,

$$y = \Phi Dx,$$  \hspace{1cm} (3.11)

the signal can be reconstructed by minimum $l_0$ or $l_1$ norm reconstruction. In our test we used $l_1$ norm minimization,

$$\hat{x} = \text{argmin}_{x} \frac{1}{2} \|y - \Phi Dx\|_2^2 + \lambda \|x\|_1.$$  \hspace{1cm} (3.12)

We compare three options for the sparsifying basis $D$,

- A standard Wavelet base,
- Wavelet + Overcomplete DCT (ODCT) Dictionary,
- Wavelet + Trained Dictionary.

We used the separable surrogate functionals (SSF) method [DDDM04, ZE10] to minimize (3.12).
Our test and training images were again fingerprint images (a different database). We used 16-tap orthogonal Symlet Wavelet (Matlab ‘sym8’) as the Wavelet base in all tests. The input image was 4096 pixels, and 1024 measurements were taken. The SSF algorithm was allowed to run for 10000 iterations. The results are shown in Figure 3.10. The “Wavelt+ODCT” shows a clear advantage over plain Wavelet, and is in turn outperformed by the “Wavelet+Trained Dictionary”.

![Figure 3.10: Compressed Sensing test - PSNR vs. λ.](image-url)
Chapter 4

Application to Image Denoising

4.1 Introduction

The problem of recovering an underlying image, or any other data, from measurements contaminated with noise is one of the most studied problems in signal processing. In the traditional set-up, a clean signal $z \in \mathbb{R}^n$ is contaminated by additive noise $\eta$ giving the noisy measurements $y = z + \eta$. The objective is then to recover the original signal by removing the noise from the corrupted data $y$. In this work, and as it is mostly assumed, we will consider the case of Additive white Gaussian iid noise, i.e. $\eta \sim N(0, \sigma^2)$.

Sparsity-based models have had a growing importance in signal processing in general, and have led to efficient algorithms in image denoising, in particular [BDE09]. This class of methods assume that a natural signal can be expressed as a linear combination of only a few atoms from a redundant dictionary $D \in \mathbb{R}^{n \times m}, n < m$. Looking for such a sparse representation accounts for solving the following problem:

$$\min_x ||x||_0 \text{ subject to } ||y - Dx||_2^2 \leq \epsilon^2,$$  \hspace{1cm} (4.1)

where $x \in \mathbb{R}^m$ is the sparse representation vector for $y$ within an accuracy of $\epsilon$, and $||x||_0$ counts the number of non-zeros in $x$. Obtaining such a sparse representation is NP-hard in general, but several greedy algorithms and other relaxations methods are at our disposal to tackle this problem under certain conditions [BDE09]. Methods such as the OMP [PRK93], MP [MZ93], FOCUSS [GR97] and others enable us to approximate the solution to the sparse coding problem.

A central issue in this approach is the choice of the dictionary. Transforms that are analytically defined might serve the purpose, providing also an important advantage in terms of their implementation by avoiding an explicit matrix representation. On the other side, learning the dictionary itself from the real data has proven to be more effective, at the expense of managing explicit matrices and more complex algorithms.
This task can be written as:

\[
\min_{D, X} \|Y - DX\|_F \quad \text{subject to } \|x_i\|_0 \leq T, \, \forall i,
\]

where \(Y \in \mathbb{R}^{n \times N}\) is a matrix containing \(N\) signal examples, and \(X \in \mathbb{R}^{m \times N}\) are the corresponding sparse vectors, both ordered column wise. Several authors have proposed iterative methods to undertake this problem [EAHH99, MBPS09]. Among them, the K-SVD algorithm [AEB06a] has been widely used for different applications in image processing. In denoising in particular, one could seek for a sparse representation over a dictionary trained to perform optimally for some group or kind of images. Interestingly, the same denoising task can be dealt with while training a dictionary on the noisy image itself achieving better performance [EA06].

Adopting a broader view, a common feature in most state of the art denoising methods is a patch-based concept: when dealing with high dimensional data, the motif is to work on overlapping patches of size \(\sqrt{n} \times \sqrt{n}\), and then tile and average the results. Some of these include the NLM [BCM05], BM3D [DFKE06], LSSC [MBP+09], and the K-SVD is no exception to this common approach. While this strategy provides a practical solution under this framework, several problems arise from working in a single and small scale scheme. As we will show later, no matter what the core algorithm is, this affects the visual quality of the denoised image. The resulting artifacts are more noticeable in large smooth areas, typical of commonplace images such as scenery and landscapes. What is there to gain by these algorithms from a more global approach? Could a multi-scale framework provide an improvement to these methods?

We propose to merge the K-SVD denoising algorithm [EA06] with a Wavelet analysis, in a similar way to the approach taken in Chapter 3. This will lead to an effective sparse decomposition of the image content using different scale atoms in a natural way. As a result, the potential of the K-SVD denoising algorithm is exploited beyond the single scale limitations, reaching state of the art results. In this chapter we describe this idea in details, tie it and contrast it to existing work, and demonstrate the effectiveness of the proposed scheme.

### 4.2 M-term Approximation of Noisy Images

We can use the M-term approximation, described in section 3.3.1, as a form of rudimentary denoising. We can seek (by exhaustive search) the best threshold to denoise the images by performing hard thresholding. In the following experiment we add noise (\(\sigma = 20\)) to fingerprint and scenery images, and plot the denoising achieved by the M-term approximation in our scheme, as a function of the number of non-zero coefficients. We compare our result to a simplified version on K-SVD denoising in the image domain. In this scheme a dictionary is trained on image patches (from the same training data set). The noisy image is broken into non-overlapping patches and global
coding is applied in a similar manner to our construction.

We used the discrete Meyer (Matlab 'dmey') Wavelet transform with three levels of decomposition for the fingerprint images and six levels deep for the scenery images. To make the comparison fair, all the dictionaries used are $64 \times 256$ (with only one dictionary trained for all Wavelet bands together). Our results can be further improved by training separate sub-dictionaries for the different bands, as in the M-term experiment in the previous chapter, thus adding more degrees of freedom to our representation without changing the overall redundancy.

The results are shown in Fig. 4.1. We make several key observations:

- The peak denoising of the multi-scale approach is achieved at a fraction $(\frac{1}{3} - \frac{1}{2})$ of the number of non-zero coefficients compared to the single-scale approach. The effect is much more dramatic for the coastal scenery images which exhibit true multi-scale, but is also seen for the fingerprint images.

- The multi-scale scheme reaches about the same PSNR level ($\pm 0.3 \text{dB}$) as the single-scale approach. This result is somewhat disappointing as our initial hope was that less coefficients would lead to less residual noise.

- A point-wise comparison of the single and multi-scale results suggests that the different methods denoise more effectively in different areas of the image. Large smooth areas are naturally denoised more effectively by the multi-scale approach which. On the other hand sharp edges are better handled by the single scale method. One explanation for this is the fact that sharp edges contain energy in many frequencies. The multi-scale decomposition splits this energy across bands, making it harder for the multi-scale approach to recover each frequency component from the noise. This observation leads us to the final algorithm presented in section 4.4.

We note that this configuration (one dictionary trained to represent all bands) is biased against our scheme. The dictionary training procedure, as is, will tend to focus more on the approximation band, and will therefore not represent the directional bands particularly well. Adding a higher redundancy to our dictionary (by training per band sub-dictionaries) improves the result, while the coding cost of the coefficients remains the same. This extension is only viable however when dictionary training is done on a large corpus of images. Only then can we collect enough patches, from each of the bands, to effectively train a separate sub-dictionary for each band.

The graphs also shows the denoising results for Wavelet coefficient hard-thresholding, for comparison. These results are significantly inferior, but the comparison is not really fair as the Wavelet dictionary on its own is non-redundant.

This process, coding non-overlapping patches, is a simplified or naive denoising scheme. A more complete multi-scale denoising method is described in the following section, however the key observations we made, and the intuition gained, remain valid.
Figure 4.1: M-Term denoising - PSNR as function of number of active coefficients. Additive Gaussian white noise ($\sigma = 20$) is cleaned by sparse coding of non-overlapping patches in the Wavelet and image domains. Left - fingerprint images (averaged over 15 test images). Right - coastal scenery images (averaged over 10 test images). For fingerprint images a 3-level 'dmey' Wavelet is used, and for coastal scenery images a 6-level 'dmey' Wavelet. All dictionaries are $64 \times 256$.

4.3 Basic Configuration Multi-scale K-SVD denoising

Our 1st step is to extend the image denoising scheme in [EA06] to a multi-scale scenario. Consider a noisy image $Y$, its wavelet transform as a collection of band images $Y_b^W = (W_A Y)_b$, and its estimated denoised version $\hat{Z}_b^W$, $b = 1, ..., L = 3S + 1$, with $S$ decomposition levels. Generalizing the work in [EA06], we propose a global maximum a posteriori (MAP) estimator for denoising the image in the Wavelet domain as:

$$\forall b, \{x_{ij,b}, D_b, \hat{Z}_b^W\} = \arg\min_{x_{ij,b}, D_b, \hat{Z}_b^W} \lambda \|Y_b^W - \hat{Z}_b^W\|_2^2 + \sum_{ij} \mu_{ij,b}\|x_{ij,b}\|_0 + \sum_{ij} \|D_b x_{ij,b} - R_{ij,b} \hat{Z}_b^W\|_2^2 \quad (4.3)$$

where $x_{ij,b}$ is the sparse vector for the $(i,j)$–patch in the decomposition band $b$, $R_{ij,b}$ is a matrix that extracts that patch from the sub-image $Z_b^W$, and $\lambda$ is a Lagrange multiplier. This optimization problem can be solved iteratively by first considering a fixed set of sub-dictionaries $D_b$ and obtaining the vectors $x_{ij,b}$ by any pursuit method. Then the sub-dictionaries are updated using a K-SVD step. These steps are repeated for a fixed number of iterations. Finally, we update $\hat{Z}_b^W$ by:

$$\hat{Z}_b^W = \left(\lambda I + \sum_{ij} R_{ij,b}^T R_{ij,b}\right)^{-1} \left(\lambda Y_b^W + \sum_{ij} R_{ij,b} D_b x_{ij,b}\right). \quad (4.4)$$

In essence we decomposing an image into sub-bands, and treat each sub-band independently as a noisy image. By using orthogonal Wavelets (with periodic extension)
the noise characteristics are retained. Additive Gaussian white noise remains as such, as does the noise level in each band (up to a constant scaling factor). Wavelet coefficient patches are denoised independently and then merged by overlapping to create denoised Wavelet coefficient sub-bands. After the different sub band images have been denoised in the Wavelet domain, the multi-scale denoised image is obtained by applying the inverse Wavelet transform.

By working on patches of the same size in all decomposition levels, we consider different-scale effective patches in the image domain. This gives our algorithm a more global outlook than that of the regular K-SVD denoising algorithm, and involves essentially the same computational complexity, plus the forward and backward Wavelet transform. The complexity analysis detailed in the previous chapter is still valid here.

### 4.4 Fusing Single and Multi-Scale Denoising Results

As we observed when comparing single and multi-scale denoising, the two approaches give similar denoising in terms of PSNR, but achieve this in a different manner. The single scale approach excels in areas with sharp edges, while the multi-scale approach does better in smooth areas of the image. This lead us to attempt to merge these 2 results in a point-wise manner, i.e. choosing per pixel whether to take its single scale denoised version or its multi-scale one (or a weighted averaging of them). The choice being made according to the statistics of the (noisy) pixels around each point. This proved tricky to do and gave results that were not significantly better than simple (equal weight) averaging of the two methods.

While the residual noise made by the two methods was of a different nature, the underlying image was denoised fairly well in both cases. This means that the signal components recovered by both methods was most likely the clean signal, while residual signal components that were retained by only one method, or the other, were most likely noise. Following this motivation, we propose to merge the outcome of the original (single-scale) K-SVD denoised image \( \hat{Z}_{ss} \) with the output of the multi-scale K-SVD algorithm proposed above, \( \hat{Z}_{ms} \). Both of these have some remaining noise and different artefacts, but correspond to the same underlying image. We aim to recover the information common to both of them by a weighted joint sparse coding, as motivated by [YQR13] and shown in Fig. 4.2. We concatenate corresponding patches of both images with a weighting factor \( \beta \) as \( \tilde{y} = [ y_{ms}^T \sqrt{1+\beta}, y_{ss}^T \sqrt{1-\beta} ]^T \in \mathbb{R}^{2n} \). We may then use the dictionary given by \( A = [ D^T \sqrt{1+\beta}, D^T \sqrt{1-\beta} ]^T \in \mathbb{R}^{2n \times m} \) to obtain the sparse vector \( \alpha \in \mathbb{R}^m \) by the OMP algorithm. Finally, the denoised patch will be given by \( \hat{z} = D\alpha / \sqrt{2} \).

As we will see later, the multi scale K-SVD algorithm outperforms the single scale K-SVD especially in the presence of high noise due to the increasing patch-like artefacts, which the multi scale approach is more robust to. This indicates that \( \beta \) should be close to 1 in such cases, and close to 0 when the noise level is lower. One could just propose
Figure 4.2: Joint sparse coding stage of the Fused K-SVD denoising algorithm.

\[ \beta_1 = \sqrt{1 + \beta}, \quad \beta_2 = \sqrt{1 - \beta}. \]

a function \( \beta = f(\sigma_n) \) accordingly, or choose an adaptive method that optimizes this parameter for each patch. For the sake of simplicity we consider here a linear function of the initial noise level, from \( \beta = 0 \) for \( \sigma = 0 \) to \( \beta = 0.9 \) for \( \sigma = 50 \). Certainly other choices are possible, and the implications of this choice will be commented later on.

4.4.1 Results

In this section we present the results of a denoising experiment on landscape images from the online NOAA library [Nat]. We chose these images as they contain large scenery areas that are poorly treated by typical patch-based denoising methods. One of this images is depicted in the top left corner of Fig.4.3. Fifteen images from this dataset, size 870×1360, were contaminated by additive white Gaussian noise with zero mean and variable standard deviation \( \sigma \). For the multi-scale decomposition we used a discrete Meyer wavelet, with 2 decomposition levels. By choosing a unitary transform, the stopping criteria for the sparse coding stage in the denoising algorithm is simply \( \epsilon = c \cdot \sigma \), where \( c = 1.15 \) following [EA06].

We evaluate our denoising results with two image quality measures: the popular Peak Signal to Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) [WBSS04]. While simple and practical, the PSNR relies only on the absolute difference pixel by pixel, and does not provide a good signal fidelity measure [WB09]. As such, its ability to compare images from a human perception point of view is poor. The SSIM is somehow a more complete image quality measure, which builds upon the idea that human perception is highly adaptive to structural information from images and visual scenes [WBSS04]. We include in the results those obtained by the BM3D algorithm [DFKE06], computed with the code made available by the authors, and with their recommended parameters. We also compare our performance against the regular single-scale K-SVD. Note that all
Figure 4.3: One image from the dataset in [Nat], and its denoising results (noise level \( \sigma_\eta = 35 \)). Top left: original image. Top right: K-SVD (PSNR = 31.01, SSIM = 0.857). Bottom left: BM3D (PSNR = 33.16, SSIM = 0.923). Bottom right: Fused K-SVD algorithm (PSNR = 33.16, SSIM = 0.940). Note the artefacts in the single scale patch based methods.

three methods use 8 \( \times \) 8 patches.

We may also benefit from choosing an appropriate initial dictionary [EA06]. To this end, we trained a single-scale and a multi-scale dictionary on 20 natural images (outside the above set of test images), for the single-scale and multi-scale versions of the K-SVD algorithm, respectively. The same single scale initial dictionary was later used to merge the final outcome of the Fused K-SVD algorithm, as described in the previous section. In this case we use OMP with an error threshold of \( \epsilon = 0.1 \cdot \sigma_\eta \), where this factor has been chosen empirically, accounting not only for the remaining noise but also for the difference in the artifacts of the two images.

In Fig. 4.5 we present the averages over all testing images for the different algorithms, relative to that of K-SVD. The multi-scale K-SVD outperforms the single-scale K-SVD in almost the whole range of noise variance, and the Fused K-SVD and BM3D present the best results, with the latest being slightly higher in terms of PSNR. Note that the last weighted joint sparse coding stage enables an extra boost, and the full fused algorithm improves the results by 0.2 - 0.3 dB compared to the plain multi-scale K-SVD. Turning to the SSIM results, the artefacts on the smooth areas in the regular K-SVD
denoised images are strongly penalized by this measure. Multi-scale K-SVD and Fused K-SVD seem to be the best, with our methods slightly outperforming BM3D. Fusion gives no gain with respect to this measure. In Fig.4.3 we depict the results of the K-SVD, BM3D and Fused K-SVD on the example image, and in Fig.4.4 the residual errors of BMSD and Fused K-SVD.

The reason for this difference in both measures should not be surprising. While BM3D makes little mistakes in terms of absolute value, these errors are more noticeable when there are large smooth areas, which causes the annoying texture artifacts that can be seen in the images in Fig. 4.3. It is in these areas where our method shows its greatest benefits. The coding of the deeper decomposition levels implies choosing big atoms yielding nicely coded smooth patches. These atoms are treated considering a more global approach than just looking at an $8 \times 8$ patch in the image domain. This makes this method more robust to higher noise levels, where the texture artifacts become stronger. However, this advantage comes at the cost of losing some details in the sharp edges of the image. The absolute error at these points are slightly higher that those made by BM3D, as noted by the PSNR results.

To finish this section, we have a word about the standard images such as Lena, Barbara, etc. On these images the Fused K-SVD algorithm performs between 0.3-0.4dB (PSNR) and 0.002-0.01 (SSIM) lower than BM3D (Fig.4.6). Note that these images are small ($512 \times 512$). Thus there is a fairly limited amount of data to train on. Furthermore, these images are highly textured, hardly presenting any smooth areas of considerable size. Both these factors handicap the multi-scale approach. We would go so far as to claim that in this day and age, where every mobile device can produce mega-pixel images, these traditional images are no longer relevant for accessing image processing algorithms. That said, even in these images, the fuzed approach shows a notable improvement over the regular single-scale K-SVD in both measures (up to 0.55 dB in PSNR and 0.035 in SSIM).
Figure 4.5: Denoising results by the K-SVD [EA06], BM3D [DFKE06], multi-scale K-SVD and Fused K-SVD algorithms, averaged over 15 testing images from the NOAA database [Nat]. Left: PSNR gain with respect to K-SVD. Right: SSIM gain with respect to K-SVD.

4.5 Conclusion

We have presented a multi-scale extension of the K-SVD denoising algorithm by proposing a global MAP estimator for the denoised image in the wavelet domain. We solve this minimization problem iteratively in terms of the K-SVD algorithm in each band, applying a multi-scale patch denoising of the image. We then boost the results by merging the single scale and multi-scale K-SVD outcome images by a weighted sparse coding step. The results obtained by this method shows the potential benefits of working within a multi-scale framework. We are able to combine bigger effective atoms that give rise to clear smooth areas, in which most current methods fail.
Figure 4.6: Denoising results by the K-SVD [EA06], BM3D [DFKE06], multi-scale K-SVD and Fused K-SVD algorithms, averaged over 10 standard testing images (Lena, Barbara etc.). Left: PSNR gain with respect to K-SVD. Right: SSIM gain with respect to K-SVD.
Chapter 5

Cropped Wavelets

5.1 Introduction

In the previous approach we applied the multi-scale Wavelet transform globally (on the whole image) and subsequently applied the sparse coding on small patches. A different approach is to apply a joint multi-scale + sparse transform on large image patches. An issue that arises acutely when applying Wavelets to an image patch (or small image), which is largely overlooked, is the issue of border handling. In this chapter we look at this issue more closely and offer an alternative to the standard approaches, that alleviates this problem, creating a Wavelet transform that can be applied to image patches.

5.1.1 Wavelets on a Finite Interval

Wavelet theory and design is rigorously studied (and mathematically sound) on the infinite and continuous domain, as is the transition to the discrete domain. However, there are several common (and less common) approaches on how to handle finite length signals, none of which is completely satisfactory.

- Periodic extension: the signal is extended in a periodic manner, and the Wavelet transform is applied to this infinite length periodic signal. In general, the transition from one side of the interval to the opposite one is not necessarily smooth or even continuous. Thus, from an analysis standpoint, an artificial edge is created, generating in turn many more (non negligible) coefficients. From a synthesis standpoint, atoms that intersect the boundary are replaced by atoms that wrap around in the finite interval. This approach is popular since it allows (with the appropriate choice of Wavelet) for an orthogonal transform.

- Symmetric extension: the signal is extended in a symmetric manner (reflected at the borders), and the Wavelet transform is applied to this infinite length reflected-periodic signal. The infinite length signal is thus continuous on interval borders. When used with symmetric Wavelets, this allows for a non-redundant transform.
The most popular use of this approach is in image compression where it is used with symmetric bi-orthogonal Wavelets.

- **Zero Padding**: the signal is padded with zeros to create an infinite length signal. Typically, standard Wavelet analysis, followed by processing and then synthesis, is applied to the signal.

- **Analytic construction of Orthogonal Wavelets on a finite interval.** Specialized Wavelets can be constructed so that all Wavelet filters are contained on the finite interval while still retaining orthogonality, for instance [CDV93]. However, the orthogonality constraint creates highly irregular Wavelet filters near the signal borders. As a result, these Wavelets are not very practical as the irregular Wavelet filters at the borders offset, as it were, the gain achieved by them being contained in the interval. Though interesting theoretically, these constructions are complicated and are seldom used.

What we found is that actually the simplest option of Zero padding, with a few crucial modifications and a clever coefficient selection scheme that takes into account the redundant nature of the transform, gives the best results. This allows for a multi-scale Wavelet transform to be applied to image patches.

### 5.2 A Closer look at Zero Padding

Let $f \in \mathbb{R}^n$ be a finite signal of length $n$, and denote $P \in \mathbb{R}^{m \times n}$ a zero-padding operator onto signal size $m > n$ (where $m$ is "large enough"). $\bar{f} = Pf$ is the zero-padded version of $f$.

**Forward Transform**: Wavelet analysis is applied to the zero-padded version of the signal. The Wavelet coefficients $g$ are obtained by:

$$g = W_A \bar{f} \quad (5.1)$$

**Inverse Transform**: Wavelet synthesis is applied to the coefficients, followed by a cropping of the signal to the original interval (denoted $P^T$).

$$f = P^T W_S g \quad (5.2)$$

What we are doing in practice (just as in periodic extension) is creating an artificial edge at the signal border. A different way of looking at this is that we are applying a unitary transform (orthogonal Wavelet with periodic extension) to the zero padded signal. The resulting coefficients try to encode not just the original signal but also the transition to the padding, resulting in “superfluous” coefficients. The forward transform
can be viewed as a minimization problem:

\[ g = \operatorname{argmin}_g \| W_S g - \bar{f} \|_2^2 \]  

(5.3)

and since \( W_A W_S = I \) we get \( g = W_A \bar{f} \). This cost function penalizes errors not only in the original interval, but in the padded one.

### 5.3 Cropped Wavelets

What we really want to do is minimize the errors only in the original interval. We can do this by introducing the cropping operator \( P^T \) to the previous equation, resulting in:

\[ g = \operatorname{argmin}_g \| P^T W_S g - f \|_2^2. \]  

(5.4)

Solving this quadratic equation gives:

\[ g = (P^T W_S)^T f = W_A Pf = W_A \bar{f}. \]  

(5.5)

The same as before!

What was previously overlooked is that while the Wavelet transform on the padded interval is unitary, if we look only at the original interval we have a redundant transform. This means that the system of equations we solved is an under-determined one. The Least Squares solution to such a problem will inevitably be a dense one, which translates to superfluous Wavelet coefficients. The solution to this problem is obvious to the reader at this point – Sparse Coding. We can add a sparsity promoting constraint to our minimization problem and rewrite it as:

\[ g = \operatorname{argmin}_g \| g \|_0 \text{ subject to } \| P^T W_S g - f \|_2^2 \leq \epsilon. \]  

(5.6)

The solution to this problem can now be approximated by OMP, almost. One more tweak needs to be done for this to work.

We are trying to sparse code by solving the synthesis problem using the dictionary \( P^T W_S \). This dictionary is composed of Wavelet synthesis atoms of size \( m \) cropped to size \( n \). The cropping operation means that some of the atoms no longer have unit norm. We therefore need to re-normalize this dictionary. We refer to the resulting operator \( W_c \) as Cropped Wavelets:

\[ W_c = P^T W_S / \| P^T W_S \|_2 \]  

(5.7)

where the division is performed on a column by column basis (i.e., each atom is normalized separately).
5.3.1 A different perspective

A different way to look at this problem is as follows. We want to minimize on both signal extension \( \bar{f} \) and the representation \( g \). Mathematically this can be stated as follows:

\[
\{g, \bar{f}\} = \arg\min_{g, \bar{f}} \|W_S g - \bar{f}\|_2^2 \text{ subject to } P^T \bar{f} = f.
\] (5.8)

This is obviously an under-determined problem as there are infinite possible extensions and subsequent coefficients. When we add to this the requirement that the coefficients be as sparse as possible, we get back (5.6).

What we gain from this way of looking at things is an understanding why Cropped Wavelets will generally outperform periodic and symmetric signal extensions. In those two approaches the signal extension is fixed to a specific extension (per signal), then the sparsest coefficients are chosen to fit this extension. In contrary, in our approach, the signal extension can be chosen as to give the sparsest possible coefficients. This will typically yield a sparser solution. An example illustrating the different extensions is given in Fig. 5.1. The extension chosen by the Cropped Wavelets is, as expected, smooth. The choice of which Cropped Wavelet atoms to use on the borders can be however less intuitive, giving surprising results, as seen on the extension to the right side.

![Figure 5.1: Border extensions for different methods. Left – periodic, Center – symmetric, Right – cropped. The original signal is marked with the thicker line.](image)

Note however that we cannot make guarantees that our method will give a sparser solution in every case. The functions minimized are different for the various extensions, meaning that in some singular cases the optimum solution, for periodic extension for example, will be lower (i.e., sparser). We can come up with an illustrative example for this by considering the “wrap around” property of the atoms in the periodic case. Atoms near the border wrap around, or continue, to the other side. Now consider a signal composed of one such atom (sparsity is 1). Cropped Wavelets do not wrap around. Their construction would actually split the periodic atom into two separate atoms, each
one with support adjacent to one border. The sparsest coding with Cropped Wavelets would therefore use 2 atoms, one for the right side of the signal, the other for the left. In this artificial case the optimal coding with periodic extension outperforms the sparsest coding with Cropped Wavelets. In real world signals, this type of occurrence is rare, and, as we will show in the following section, Cropped Wavelets outperform the other methods, on average, by a wide margin.

5.3.2 A note on separability

The standard way to extend one level of the 1-D Wavelet transform to two (or more) dimensions is by applying the 1-D transform to each dimension separately in a cascade (followed by down-sampling). 2-D atoms are thus the separable product of two 1-D atoms. In the next decomposition scale this procedure is repeated on the approximation band of the previous level. What this means is that while each atom (and each level) is in itself separable, the complete multi-scale transform is not.

Figure 5.2: Example of standard (left) and separable (right) Wavelet decompositions of house image.

An alternative extension to multiple dimensions is a completely separable construction where a complete 1-D multi-scale decomposition is done in turn in each dimension (Fig. 5.2). The different cascaded implementations are shown in Fig. 5.3 and Fig. 5.4.

The separable transform can be conveniently represented in matrix form. Considering all the basis elements of the 1-D Wavelet transform (in all scales) arranged column-wise in the matrix $\Phi_1$, the complete 2-D separable transform can be represented as $\Phi_2 = \Phi_1 \otimes \Phi_1$, where $\otimes$ denotes the Kronecker product. Given an image $A$, its transform can be computed by matrix multiplication from the right and left $\hat{A} = \Phi_1 A \Phi_1^T$.

While the standard 2-D extension is typically implemented via a cascade of filter
convolutions and down-sampling operations, when dealing with (large) image patches it is substantially faster to apply the separable transform via matrix multiplications.

Even though computational efficiency was our initial motivation for using separable Wavelets, a surprising result came out in our experiments with natural image data. The separable transform actually performed better than the standard 2-D approach when dealing with natural images. The separable transform needed less coefficients (on average) to represent the same image data. Our intuition on the matter is that the elongated atoms the separable transform uses are more suitable for natural images.

This modification is not only applicable to the traditional Wavelet transform, but also to the cropped Wavelets dictionary introduced above.
Figure 5.3: Cascaded implementations of standard Wavelet decompositions.

Figure 5.4: Cascaded implementations of separable Wavelet decompositions.
5.4 Comparison

In the following experiments we compare traditional Wavelets with our Cropped Wavelets on various signal types. In all the cases, for the traditional methods we use non-linear approximation as the standard method of signal reconstruction. In this scheme the signal 1st passes through a Wavelet Analysis phase. The resulting coefficients are then sorted according to magnitude, and only the largest are kept (others are nullified). Finally, the signal is reconstructed by Wavelet synthesis. We measure the reconstruction error as a function of the number of coefficients that are retained.

For the Cropped Wavelets we (approximately) solve the sparse synthesis problem using the Cropped Wavelet atoms as our dictionary by Orthogonal Matching Pursuit. As before we measure the reconstruction error as a function of the number of coefficients used.

We first compare traditional Wavelets with our Cropped Wavelets on synthetic 1D signals. The signals, of length 32, are created by concatenating two randomly created polynomials on the unit interval. The resulting signals are thus piecewise smooth, a class of signals for which Wavelets are optimally suited for. All signals are normalized to have unit norm. Reconstruction error as a function of the number of atoms, averaged over 1,000 signals is shown in Fig. 5.5. While symmetric extension and zero-padding perform poorly, periodic extension actually performs well due to the orthogonality of the transform. Nevertheless Cropped Wavelet outperform all the other methods. The Wavelets used in this experiment are Daubechies of length 4, though similar results were observed using other Wavelets.

![Figure 5.5: Average reconstruction error as a function of number of active coefficients for 1D synthetic data.](image)

A second similar experiment is run on natural 1D signals. We extract 1,000 random 32 pixel line segments from 10 standard images (Lena, Barbara etc.). Results are shown in Fig. 5.6. Results are similar to the synthetic case. These signals are actually simpler
than the artificial ones (a noisy piece-wise constant model is more appropriate here), a fact that could explain why the Cropped Wavelets perform only marginally better than the orthogonal ones.

Figure 5.6: Average reconstruction error as a function of number of active coefficients for 1D image data.

While the average representation error of orthogonal Wavelets with periodic extension is only slightly higher than Cropped Wavelets, the location of the errors is the key to understanding the benefits of Cropped Wavelets. In Fig. 5.7 we compare the average point-wise error for these two methods, for a fixed number of coefficients. In general, discontinuities in the signal generate error spikes. While the discontinuity at the center point is an inherent part of the signal, the discontinuities on the edges are artificially produced by the periodic extension.

Figure 5.7: Average point-wise reconstruction error using 5 coefficients for 1D synthetic data.

In the following experiment we compare traditional Wavelets as well as Cropped and Seperable-Cropped Wavelets on 2D signals. We extract 1,000 random 32 × 32 image
patches from 10 standard images (Lena, Barbara etc.). Results are shown in Fig. 5.8. The separable Wavelet decomposition clearly outperforms the standard one for natural images. Both these decompositions used the same orthogonal Wavelets with periodic extension. Our separable Cropped Wavelet outperforms both other methods by a wide margin, much more so than in the 1D scenario.

![Figure 5.8: Average reconstruction error as a function of number of active coefficients for 2D image data.](image)

### 5.5 Conclusion

In this chapter we presented a new Wavelet scheme that outperforms standard methods while reducing border effects. By combining a separable and redundant Wavelet dictionary with a pursuit method targeting sparse signal representation we can provide a multi-scale Wavelet decomposition for small-medium sized images that solves the border problems of traditional methods in a computationally efficient manner.
Chapter 6

Online Sparse Dictionary Learning

6.1 Introduction

In this chapter we explore a different method to get large atoms and go beyond the $8 \times 8$ patches typically used in image processing. Unlike Chapter 3 where the small Wavelet domain atoms translated indirectly to large image patches, here we take a more direct approach. This requires two things:

1. A model that gives constrained/structured yet expressive atoms.
2. An algorithm for training on big data.

The model we chose to work with is the Double-Sparsity model introduced in [RZE10], and the training algorithm is an online algorithm, based on Stochastic Gradient Decent. These components and how they come together are described in the following sections.

6.2 Extending The Double Sparsity Model

Learning dictionaries for large signals requires adding some constraint to the dictionary, otherwise signal diversity and the number of training examples needed make the problem intractable. Often, these constraints are given in terms of a certain structure. One such approach is the double-sparsity model [RZE10]. In this model the dictionary is assumed to be a multiplication of a fixed and fast transform $B$ (we will refer to it as the base-dictionary) by a sparse adaptable matrix $A$. Every atom in the effective dictionary $D$ is therefore a linear combination of few and arbitrary atoms from the base-dictionary. Formally, this means that the training procedure requires solving the following problem:

$$\min_{A,X} \| Y - BAX \|_F^2 \quad \text{s.t.} \quad \begin{cases} \| x_i \|_0 \leq p \quad \forall i \\ \| a_j \|_0 = k \quad \forall j. \end{cases}$$ (6.1)
Note that the number of columns in $B$ and $A$ might differ, allowing flexibility in the redundancy of the effective dictionary.

The authors in [RZE10] used an over-complete Discrete Cosine Transform (DCT) as the base-dictionary in their experiments. This base dictionary has been shown to work very well for small image patches (as in the popular JPEG compression algorithm). This dictionary does not however provide a good (i.e. sparse) representation for significantly larger patches. Large image patches require both localized atoms as well as large structure atoms, or in other words a multi-scale dictionary. This is a quality that the periodic atoms in the DCT dictionary simply do not have.

Using Wavelets as a base dictionary was proposed in [RZE10] (among other options) but this option was never implemented. This is due to two main issues.

- The first issue is the significant border-effects Wavelets have in small-to-medium sized patches, making standard Wavelets a mediocre choice at best for a base dictionary.

- The second issue was a technical implementation issue. Their efficient implementation of all dictionary operations required the base dictionary to be completely separable, which the traditional Wavelet transform is not.

Both these issues are addressed by the use of the Separable Cropped Wavelets, introduced in the previous chapter.

Thus, by using Cropped Wavelets as the base dictionary we get a model that is both expressive enough to model large image patches in a good way, while remaining structured and with a manageable amount of degrees of freedom to allow for efficient training.

### 6.3 Online Learning

In the Double-Sparsity model the dictionary update is now constrained by the number of non-zeros in the columns of $A$. In [RZE10] a variant of the K-SVD algorithm (termed Sparse-K-SVD) was proposed for updating the dictionary. As in the work in [AEB06a], this is a batch method that updates every atom sequentially. In the context of the double-sparsity structure, this task is converted into a sparse-coding problem, and approximated by the OMP algorithm.

When working with higher dimensional data, however, the required amount of training examples and the corresponding computational load increase. In this big-data (or medium-data) scenario, it is often unfeasible to perform several sweeps over the entire data set. In some cases, the dimensionality and the amount of data might restrict the learning process to only a couple of iterations. In an extreme online learning set-up, each data sample is seen only once as new data flows in, in a completely online scheme. In this regime of work it may be impossible to even store all training samples in memory.
during the training process. These reasons lead naturally to the formulation of an online training method for the double-sparsity model, detailed below.

In this section, we propose an Online Sparse Dictionary Learning (OSDL) algorithm based on the popular Stochastic Gradient Descent algorithm, and show how it can be applied efficiently to this specific problem. We should note that online dictionary learning, and specifically SGD, have been already applied to the dictionary-learning task [SE10, AE08, MBPS09]. The special setting analyzed here is different for two main reasons:

• The dimensions involved are different, as we will be handling up to 4,000 entries in a typical example.

• The double-sparsity structure imposes a tailored approach.

### 6.3.1 A Stochastic Gradient Descent Approach

As a reminder, we are interested in the solution of the problem given by:

\[
\min_{A,X} ||Y - BA X||^2_F \quad \text{s.t.} \quad \begin{cases} ||x_i||_0 \leq p & \forall i \\ ||a_j||_0 = k & \forall j \end{cases}. \tag{6.2}
\]

A popular practice in dictionary learning, which has been shown to be quite effective, is to employ a block coordinate minimization over this non-convex problem. This often reduces to alternating between a sparse coding stage, throughout which the dictionary is held constant, and a dictionary update stage in which the sparse coefficients (or their support) are kept fixed. We shall focus on the second stage, as the first remains unchanged, essentially applying sparse coding to a group of examples. In the dictionary update stage, the problem to consider is the following:

\[
\min_A f(A) \quad \text{s.t.} \quad ||a_j||_0 = k \quad \forall j, \tag{6.3}
\]

where the cost function is given by

\[
f(A) = \sum_{i=1}^{N} f_i(y_i, x_i, A) = \sum_{i=1}^{N} ||y_i - BA x_i||^2_F. \tag{6.4}
\]

The gradient of the cost function \(\nabla f(A)\) is given by

\[
\nabla f(A) = B^T \sum_{i=1}^{N} (y_i - BA x_i)x_i^T. \tag{6.5}
\]

The purpose of computing the gradient of our objective function is to perform a gradient step. However, this update would clearly not give a sparse solution since \(\nabla f(A)\) is not expected to be sparse. Therefore, this update is then projected onto the set of
matrices with \( k \)-sparse columns. This way, we perform a projected gradient step to minimize the problem in (6.3), i.e.,

\[
A_{t+1} = \mathcal{P}_k [A_t - \eta_t \nabla f (A_t)],
\]

(6.6)

where \( \eta_t \) is a step size and \( \mathcal{P}_k \) is a hard-thresholding step that keeps the \( k \) largest entries in each column.

As noted in other works [AE08, MBPS09], it is not compulsory to accumulate all the examples to perform an update in the gradient direction. Instead, we turn to a Stochastic (projected) Gradient Descent approach. In this scheme, instead of computing the expected value of the gradient by the sample mean over all examples, we estimate this gradient over a randomly chosen example \( y_t \), and perform the update accordingly:

\[
A_{t+1} = \mathcal{P}_k [A_t - \eta_t \nabla f (A_t, y_t, x_t)],
\]

(6.7)

where

\[
\nabla f (A_t, y_t, x_t) = B^T(y_t - BA_t x_t) x_t^T.
\]

(6.8)

With these elements, we outline the basic idea of the method in Algorithm 6.1.

**Algorithm 6.1** Stochastic Projected Gradient Descent for Sparse Dictionary Learning.

**Data:** Training samples \( \{y_i\} \), base-dictionary \( B \), initial sparse matrix \( A_0 \)

**For** \( t = 1, \ldots, T \)

1. Draw \( y_t \) at random
2. \( x_t \leftarrow \text{Sparse Code} (y_t, B, A_t) \)
3. \( \nabla f^t = B^T(y_t - BA_t x_t) x_t^T \)
4. \( A_{t+1} = \mathcal{P}_k [A_t - \eta^t \nabla f^t] \)

**EndFor**

**Result:** Sparse Dictionary \( A \)

### 6.3.2 OSDL In Practice

We now turn to describe the Online Sparse Dictionary Learning algorithm in more detail. The following practical modifications and tweaks to the basic algorithm improve both the overall convergence and alleviate the computational cost:

**Mini-Batches:** Instead of considering single sample \( y_t \) per iteration, a common practice in Stochastic Gradient Descent (SGD) based algorithms is to consider mini-batches \( \{y_i\} \) of \( N \) examples arranged in the matrix \( Y_t \). Adopting this strategy has an additional benefit in our case, because we can now take advantage of the efficient batch version of the OMP algorithm in the sparse coding step. Moreover, further acceleration can be obtained by pre-computing the projection \( D^TY_t \) [RZE08]. Though this would be prohibitive for handling the whole data set due to the size of the matrix \( Y \), it is still
feasible for mini-batches. This can also be computed efficiently due to the separability of the base dictionary $B$ and the sparsity of matrix $A$.

**Base Dictionary Separability:** Recalling that the operator $B$ is separable, so a multiplication by $B$ or $B^T$ need not be computed explicitly but instead obtained in terms of the one-dimensional operators $B_1$. Image patches are usually vectorized and arranged as columns of the matrix $Y_t$. Whenever one needs to apply $B$ or its transposed, we rearrange each vectorized patch $x$ back as a 2-D matrix, $x_m$, and apply $B_1$ (by matrix multiplication) from the left and right, $B_1x_mB_1^T$, finally reverting the outcome back to a column vector.

**Partial Dictionary Update:** Due to the limited amount of examples in each mini-batch $Y_t$, it is unlikely that these examples use all the atoms in the dictionary. Rather, only a fraction of the dictionary is used, and hence needs to be updated. We therefore restrict the update to those columns of $A$ that are used by the current samples. Let $S = supp(X_t)$ denote the set of chosen atoms at the $t^{th}$ iteration. Then, we can discard the unused atoms and restrict the update of the gradient as:

$$\nabla f(A_t^S) = B^T(Y_t - BA_t^S X_t^S)X_t^{ST}.$$  \hspace{1cm} (6.9)

Naturally, the dictionary is then updated only in the atoms indicated by $S$, as

$$A_{t+1}^S = P_k [A_t^S - \eta_t \nabla f(A_t^S)].$$  \hspace{1cm} (6.10)

**Step Size:** An important point in all stochastic learning methods is the choice of the step size $\eta_t$. In this respect, note that if our problem was only to minimize the quadratic term in Eq. (6.4), we could obtain the optimal step size as the solution to the following minimization problem:

$$\eta_t = \arg \min \eta f(A_t - \eta \nabla f(A_t)),$$  \hspace{1cm} (6.11)

with a closed-form solution given by

$$\eta^* = \frac{||\nabla f(A)||_F^2}{||B \nabla f(A) X||_F^2}.$$  \hspace{1cm} (6.12)

Considering now the sparsity constraint on the columns of the matrix $A$, this choice of step size is clearly not optimal. As shown in [BD10] in the context of IHT algorithms, other – more sophisticated – alternatives could be suggested for the choice of $\eta_t$, but they are out of the scope of this work. We have found that the square root of the optimal step size provides good convergence in practice, and we use this value in our experiments.

**Gram Matrix Update:** As explained in detail in [RZE08], the computational cost of the OMP sparse coding algorithm can be reduced by pre-computing (and storing) the Gram matrix of the dictionary $D$, given by $G = D^T D$. In a regular online learning
Algorithm 6.2 Online Sparse Dictionary Learning (OSDL) algorithm.

Data: Training samples \( \{y_i\} \), base-dictionary \( B \), initial sparse matrix \( A_0 \)
Initialization: \( G_B = B^T B; U = 0 \)

For \( t = 1, \ldots, T \)
1. Draw a mini-batch \( Y_t \) at random
2. \( X_t \leftarrow \) Sparse Code \( (Y_t, B, A_t, G_t) \)
3. \( S = \text{supp}(X_t) \)
4. \( \nabla f(A_t^S) = B^T(Y_t - BA_t^S X_t^S) X_t^{ST} \)
5. \( E = B \nabla f(A_t^S) X_t^S \)
6. \( \eta = \|\nabla f(A_t^S)\|_F / \|E\|_F \)
7. \( U_t^S + \gamma U_t^S + \eta_t \nabla f(A_t^S) \)
8. \( A_t^{S+1} = \mathcal{P}_k [A_t^S - U_t^S] \)
9. \( C = A_t^{ST+1} G_B A_t^{S+1} \)
10. \( G_{col}^{t+1} = C \)
11. \( G^{row}_{t+1} = C^T \)

EndFor

Result: Sparse Dictionary \( A \)

scheme, this would be infeasible due to the need to recompute this matrix for each example. In our case, however, there are only a few atoms that get updated per batch. We exploit this by updating only the respective rows and columns of the symmetric matrix \( G \) each time. Moreover, this update can be done efficiently due to the sparsity of the dictionary \( A \).

**Momentum Term:** Stochastic algorithms often introduce different online strategies to regularize the learning process. In our case, we incorporate in our algorithm a momentum term \( U_t \) (which affects only the atoms used by the current batch), and controlled by a parameter \( \gamma \). This term helps to attenuate oscillations and can speed up the convergence by incorporating information from the previous gradients.

**Dictionary Pruning:** Algorithms that tackle this kind of non-convex optimization problems are prone to local minimum traps. Common strategies to alleviate this problem include the replacement of (almost) unused atoms (for example, by a training sample) and the pruning of atoms which are too similar. We incorporate these strategies here as well, checking for these cases once every few iterations.

The main components of the algorithm are depicted in Algorithm 6.2.

**6.3.3 Complexity and Run-time Considerations**

In this section we want to address the computational cost of the proposed online learning scheme and compare it to the Sparse K-SVD algorithm. As was thoroughly discussed in [RZE10], the sparse dictionary enables an efficient sparse coding step. In particular, any multiplication by \( D \), or its transpose, has a complexity of \( T_D = O(km + T_B) \), where \( m \)
is the number of atoms in $B$ (assume for simplicity that this is the same as the columns in $A$), $k$ is the atom sparsity and $T_B$ is the complexity of applying the base dictionary. For the separable case, this reduces to $T_B = O(n\sqrt{m})$. The complexity analysis done in [RZE10] for the OMP algorithm remains valid here, and we refer the reader to the discussion therein.

We now explore the complexity of the dictionary update step. We denote by $N$ the number of examples in each mini-batch. In the OSDL algorithm, calculating the gradient $\nabla f(A^S)$ has a complexity of $T_{\nabla f} = O(T_D N + nN|S|)$. Recall that $S$ is the set of atoms used by the current samples, and that $|S| < m$; i.e., the update is applied only on a subset of the atoms. Obtaining $E$ in Algorithm 6.2 involves applying the separable dictionary to $N$ $m$-dimensional signals (after the matrix multiplication), and so it has a complexity of $T_E = O(T_B N + mn|S|)$. The calculation of the step size involves calculating the Frobenius norm of two matrices, and yields a complexity of $T_\eta = O(|S|(m + n))$. The same holds for the update of the variable $U$.

The update of the sparse dictionary $A$ is carried out by a simple hard thresholding step. This is done by sorting the elements in each column of the candidate $A^S_{t+1}$, and keeping the first $k$ ones. This step has a time complexity of $T_A = O(|S|m \log(m))$. This a an extremely cheap update when compared to the Sparse K-SVD algorithm. In Sparse K-SVD, each sparse atom is updated using the OMP algorithm (after some variable manipulations). This greedy selection of the coefficients is substantially more expensive, growing in order of $O(k^3)$. Note that this imposes a severe limitation when the number of non-zeros per atom $k$ is significantly increased, as is necessary in high dimensional data. In our case, the update of the matrix $A$ is driven by the dimension of the atoms $m$, and not by the number of non-zeros in them.

We also want to mention that this complexity comparison is to be taken in the respective context of each algorithm: while the Sparse K-SVD update is significantly more expensive, this is done only a few times (say, 10) during training, putting the main computational load in the Sparse Coding stage. In the case of the OSDL, having a much simpler update enables us to shift to an online methodology, where the dictionary update is performed every mini-batch.

Finally, to understand how these two algorithms compare in practice, we show a comparison of the run time of both methods in Experiment 1. Such a comparison is not always possible, since the Sparse K-SVD method is limited by the amount of memory needed to store the examples, and has to perform several data sweeps. We thus set a scenario in which the Sparse K-SVD algorithm is still feasible: we train dictionaries for different patch sizes on about 250,000 natural image patches. The atom sparsity is set to 10% of the number of atoms in $B$ and the signal sparsity to 5%. The redundancy of $B$ is determined by the redundancy of the Cropped Wavelets alone as the matrix $A$ is set to be square in this experiment. In order to make the comparison fair, we run the OSDL for 7 iterations and the Sparse K-SVD for 15 iterations, and then measure the time each algorithm takes in reaching the 90% of the final error. As we see from the
results in Fig. 6.1, OSDL not only converges faster from a data-sweep point of view, but it also takes about an order of magnitude less than Sparse K-SVD in run-time in a relatively small-data setup. This experiment was run on a 64-bit operating system with an Intel Core i7 microprocessor, with 8 Gb of RAM, in Matlab.

To compare not only the speed but also the quality of the end result we propose the following experiment, again comparing the OSDL algorithm with Sparse K-SVD. We learn an adaptive sparse dictionary from patches of size $32 \times 32$ from the popular image Lena. The atom sparsity is set to 40, and each patch is coded with 20 non-zeros, after subtracting their mean values. We train dictionaries with both the OSDL algorithm and Sparse K-SVD. Training was done on 150,000 examples, leaving 30,000 as a test set. Both methods use the Cropped Wavelets dictionary as their base dictionary $B$, built with a Symlet Wavelet with 8-taps. The redundancy of this base dictionary is of 1.75 (in 1-D), and the matrix $A$ was set as square, resulting in a final redundancy for the entire dictionary of just over 3.

It is clear from Fig.6.2 that OSDL provides a much faster convergence than Sparse
K-SVD. Moreover, the online method provides better generalization, as depicted by the small difference between training and test data – which is only slightly higher than the mean value of the training error. In the Sparse K-SVD case, the representation error in the testing set is relatively higher. More can be said about the dictionaries that are obtained by both methods, shown in Fig.6.3. The dictionary obtained by Sparse-KSVD (middle) contains very similar atoms, and retains some of the structure from the original Cropped Separable Wavelet dictionary, shown in the left. The dictionary obtained by OSDL, on the other hand, provides a somewhat richer collection of atoms which might help explain the lower representation error.
Figure 6.3: Experiment 2: Set of dictionary atoms. Top: Base separable and cropped wavelets dictionary. Center: Sparse Dictionary obtained with Sparse K-SVD. Bottom: Sparse Dictionary obtained with OSDL.
6.4 Experiments

6.4.1 Application to Image Restoration

In the context of image restoration, most state-of-the-art algorithms take a patch-based approach. While the different algorithms differ in the models they enforce on the corrupted patches (or the prior they chose to consider, in the context a Bayesian formulation) the general scheme remains very much the same: overlapping patches are taken from the degraded image, which are then restored more or less independently, before they are merged back together by averaging.

Though this provides an effective option, this locally-focused approach is far from being optimal. As noted in several recent works [SOE14, SE15, RE15], not looking at the image as a whole causes inconsistencies between adjacent patches which often result in texture-like artifacts. A possible direction to seek for a more global outlook is, therefore, to allow for bigger patches.

In this section we intend to show that benefit can indeed be found in managing bigger patches in image restoration, given a algorithm which can cope with the dimension increase. To this end, we present an image denoising experiment of several popular images, for increasing patch sizes. In the context of sparse representations, an image restoration task can be formulated as a Maximum a Posteriori formulation [EA06]. In the case of a sparse dictionary, this problem can be posted as:

\[
\min_{z, x_i, A} \lambda \frac{1}{2} ||z - y||_2^2 + \sum_i ||B_ix_i - R_i z||_2^2 + \mu_i ||x_i||_0, \tag{6.13}
\]

where \( z \) is the image estimate given the noisy observation \( y \), \( R_i \) is an operator that extracts the \( i^{th} \) patch from a given image and \( x_i \) is the sparse representation of the \( i^{th} \) patch. We can minimize this problem by taking a similar approach to what we have done in the dictionary learning problem: use a block-coordinate descent by fixing the unknown image \( z \), and minimizing w.r.t the sparse vectors \( x_i \) and the dictionary (by any dictionary learning algorithm). Lastly, we can fix the sparse vectors and update the image \( z \). Note that even though this process should be iterated (as effectively shown in [SE15]) we stick to the first iteration of this process to make a fair comparison with the K-SVD based algorithms.

For this experiment, denoted as Experiment 3, we use both methods for training the double sparsity model, Sparse K-SVD and OSDL. Each method is run with both base dictionaries: the traditional overcomplete ODCT and the Cropped Wavelets dictionary, presented in this paper. We include as a reference the results of the K-SVD denoising algorithm [EA06], which trains a regular (dense) dictionary with patches of size 8 × 8. The dictionary sparsity was set to be 10% of the signal dimension. Regarding the size of the dictionary, the redundancy was determined by the redundancy of the Cropped Wavelets (as explained in Chapter 5), and setting the sparse matrix \( A \) to be square.
This selection of parameters is certainly not optimal. For example, we could have set the redundancy as an increasing function of the signal dimension. However, learning such increasingly redundant dictionaries is limited by the finite data of each image. Therefore, we use always a square matrix $A$, leaving the study of other alternatives for future work. 10 iterations were used for the K-SVD methods and 5 iterations for the OSDL.

Fig. 6.4 presents the averaged results over the set of 10 publicly available images used by [LBM13], where the noise standard deviation was set to $\sigma = 30$. Note how the original algorithm presented in [RBE10], Sparse K-SVD with the overcomplete DCT as the base dictionary, does not scale well with the increasing patch size. In fact, once the base dictionary is replaced by the Cropped Wavelets dictionary, the same algorithm shows a jump in performance of nearly 0.4dB. A similar effect is observed for the Online Sparse Dictionary Learning algorithm, where using the Cropped Wavelets dictionary performs the best.

Employing even greater patch sizes eventually results in decreased denoising quality, even for the OSDL with Cropped Wavelets. Partially, this could be caused by the limitation of the sparse model for representing fine details as the dimension of the signal grows. As mentioned above, the amount of training data is limited by the size of the image, having approximately 250,000 examples to train on. Once the dimension of the patch increases, the amount of training data might become a limiting factor in the denoising performance.

As a final word about this experiment, we note that treating all patches the same way (with the same patch size) is clearly not optimal. A multi-size patch approach
has already been suggested in [LNDF12], though in the context of the Non-Local Means algorithm. The OSDL algorithm may be the right tool to bring multi-size patch processing to sparse representation-based algorithms, and this remains a topic of future work.

6.4.2 Adaptive Image Compression

Image compression is the task of reducing the amount of information needed to represent an image, such that it can be stored or transmitted efficiently. In a world where image resolution increases at a surprising rate, more efficient compression algorithms are always in demand. In the section, we do not attempt to provide a complete solution to this problem, but rather show how our online sparse dictionaries approach could indeed aid a compression scheme.

Most (if not all) compression methods rely on sparsifying transforms. In particular, JPEG2000, one of the best performing and popular algorithms available, is based on the 2-D wavelet transform. Dictionary learning has already been shown to be beneficial in this application. In [BE08], the authors trained several dictionaries for patches of size $15 \times 15$ on pre-aligned face pictures. These offline trained dictionaries were later used to compress images, of this type, by sparse coding the respective patches of each picture. The results reported in [BE08] surpass those of JPEG2000, showing the great potential of similar schemes.

In the experiment we are presenting here (Experiment 4), we go beyond the locally based compression scheme and propose to perform naive compression by just keeping a certain number of coefficients through sparse coding, where each signal is the entire target image. To this end, we use the same data set as in [BE08] consisting of over 11,000 examples, and re-scaled them to a size of $64 \times 64$. We then train a sparse dictionary on these signals with OSDL, using the cropped wavelets as the base dictionary for 15 iterations. For a fair comparison with other non-redundant dictionaries, in this case we chose the matrix $A$ such that the final dictionary is non-redundant (a rectangular tall matrix). A word of caution should be said regarding the relatively small training data set. Even though we are training just over 4000 atoms on only 11,000 samples, these atoms are only 250-sparse. This provides a great reduction to the degrees of freedom during training. A subset of training examples can be seen in Fig. 6.5a and some of the obtained atoms in Fig. 6.6a.

The results of this naive compression are shown in Fig. 6.7a for a testing set (not included in the training). As we see, the obtained dictionary performs substantially better than wavelets – on the order of 8dB at a given coefficient count. Partially, the performance of our method is aided by the cropped wavelets, which in themselves perform better than the regular 2-D wavelet transform. However, the adaptability of the matrix $A$ results in a much better compression-ratio. Interestingly, a substantial difference in performance with respect to the cropped Wavelets is obtained after training.
Figure 6.5: Experiment 4: a) Subset of examples from database of aligned face images. b) Subset of examples from “Cropped Labeled Faces in the Wild” database.

Figure 6.6: Experiment 4: a) Subset of atoms from a sparse dictionary trained with OSDL on a database of aligned face images. b) Subset of atoms from a sparse dictionary trained with OSDL on “Cropped Labeled Faces in the Wild” database.

Figure 6.7: Experiment 4: a) Compression results (as in ratio of kept coefficients) by Wavelets, Cropped separable wavelets, PCA and OSDL on aligned faces. b) Compression results for the “Cropped Labeled Faces in the Wild” database.
with OSDL, even while the redundancy of the obtained dictionary is less (by about half) than the redundancy of its base-dictionary.

As one could observe from the obtained dictionary atoms, some of them might resemble PCA-like basis elements. Therefore, and for the sake of comparing with an adaptable transformation, we include the results by compressing the testing images with a PCA transform, obtained from the same training set – essentially, performing a dimensionality reduction. As one can see, the PCA results are better than Wavelets due to the regular structure of the aligned faces, but they are still relatively far from the results achieved by OSDL.

Lastly, we show that this naive compression scheme, based on the OSDL algorithm, does not rely on the regularity of the aligned faces. To support this claim, we perform a similar experiment on images obtained for the “Cropped Labeled Faces in the Wild” database [SL09]. This database includes images of subjects found on the web, and its cropped version consists of $64 \times 64$ images including only the face of the different subjects. These face images are in different positions, orientations, resolutions and illumination conditions. We trained a second dictionary for this database, which consists of just over 13,000 examples, with the same parameters as in the previous case, and the compression is evaluated on a testing set not included in the training. Some images and atoms are shown in Fig. 6.5b and 6.6b, respectively. As shown in Fig. 6.7b, our scheme still performs about 5dB better than the regular Wavelet transform. In particular, note how the PCA results are now inferior, due to the lack of regularity of the images.

### 6.4.3 Pursuing Universal Big Dictionaries

Dictionary learning has shown how to take advantage of sparse representations in specific domains, however dictionaries can also be trained for more general domains (i.e., natural images). For relatively small dimensions, several works have demonstrated that it is possible to train general dictionaries on patches extracted from non-specific natural images. Such general-purpose dictionaries have in turn been used in many applications in image restoration [BDE09], outperforming the analytically-defined transforms.

Using our algorithm we want to tackle the training of such universal dictionaries for large image patches of sizes $32 \times 32$ and $64 \times 64$ (i.e., of dimension 1024 and 4096, respectively). In the $32 \times 32$ case we train a sparse dictionary with a total redundancy of 6: the Cropped Wavelet dictionary introduced a redundancy of around 3, and the matrix $\mathbf{A}$ was set to have a redundancy of 2. The atom sparsity was set to 200, and each example was coded with 60 non-zero in the sparse coding stage. Training was done on 10 Million patches taken from natural images from the Berkeley Segmentation Dataset [MFTM01]. The OSDL algorithm was run for two data sweeps.

We evaluate the quality of such a trained dictionary in an M-Term approximation experiment on 600 patches (or little images). Comparison is done with regular and with the separable-cropped Wavelets (the last one being the base-dictionary of the double
Figure 6.8: Experiment 5: Subset of the general (sparse) dictionary for patches of size $32 \times 32$ obtained with OSDL trained over 10 million patches from natural images.

sparsity model, and as such the starting point of the training). We also want to compare our results with the approximation achieved by some more sophisticated multi-scale transforms, such as Contourlets. Contourlets are a more advanced extension of the multi-scale analysis to two dimensions, providing an optimal approximation rate for piece-wise smooth 2-D functions with discontinuities along twice differentiable curves [DV05]. This is a slightly redundant transform due to the Laplacian Pyramid used for the multi-scale decomposition (redundancy of 1.33). Note that traditionally, hard-thresholding is used to obtain an M-term approximation, as implemented in the code made available by the authors. However, this is not optimal in the case of redundant dictionaries. We therefore construct an explicit Contourlet synthesis dictionary, and apply the same greedy pursuit approach we employ throughout the paper. Thus we fully leverage the approximation power of this transform, making the comparison fair.

Moreover, and to provide a more complete picture of the different transforms, we include also the results obtained for Cropped Contourlets. Since Contourlets are not separable we use a 2-D extension of the cropping procedure detailed in Section 5 to construct a Cropped Contourlets synthesis dictionary. The lack of separability makes this dictionary considerably less efficient computationally. As in Cropped Wavelets, we naturally obtain an even more redundant dictionary (redundancy factor of 5.3). Note that this is higher than the obtained redundancy for the Cropped Wavelets (redundancy factor of 3) we use in this experiment, theoretically giving the Cropped Contourlets an additional advantage.

Another option to consider is to use undecimated multi-scale transforms. The Undecimated Wavelet Transform (UDWT) [Mal08] and the Non-Subsampled Contourlet Transform (NSCT) [CZD06, ER06] are shift-invariant versions of the Wavelet and Contourlet transforms, respectively, and are obtained by skipping the decimation step at each scale. This greater flexibility in representation, however, comes at the cost of
a huge redundancy, which becomes a prohibiting factor in any pursuing scheme. A
similar undecimated scheme could be proposed for the corresponding cropped transforms,
however, but this is out of the scope of this work.

A subset of the obtained dictionary is shown in Fig. 6.8, where the atoms have been
sorted according to their entropy. Very different types of atoms can be observed: from
the piece-wise-constant-like atoms, to textures of different scales and edge-like atoms. It
is interesting to see that Fourier type atoms, as well as Contourlet and Gabor-like atoms
naturally arise out of the training. In addition, such a dictionary naturally obtains some
flavor of shift-invariance. As can be seen in Fig. 6.9, similar patterns may appear in
different locations in different atoms. An analogous question could be posed regarding a
rotation invariance. Furthermore, we could consider enforcing these, or other, properties
explicitly in the training. These, and many more questions, are the lines of on-going
work.

Figure 6.9: Experiment 5: Atoms of size 32 × 32 with recurring patterns at different
locations.

The approximation results are shown in Fig. 6.10. As can be seen, Contourlets
performs slightly better than Wavelets. The cropping of the atoms significantly enhances
the results for both transforms. Interestingly, Cropped Wavelets show a slight advantage
for over Cropped Contourlets. The Trainlets, obtained with the proposed OSDL, give
the highest PSNR, showing that considerable advantage can be obtained by using a
trained dictionary over a fixed analytic one.

As a last experiment, we want to show that our scheme can be employed to train an
adaptive dictionary for even higher dimensional signals. In this experiment, we perform
a similar training with OSDL on patches (or images) of size 64 × 64, using an atom
sparsity of 600, on 10 million examples from the same database. The Cropped Wavelets
dictionary has a redundancy of 2.44. We set the matrix A to be square, so the only
redundancy comes from the Cropped Wavelets. Similar runs were done with different
redundancy settings (for example, setting the overall redundancy to be the same as
Contourlets, or with a tall A giving the overall dictionary less redundancy than the base
dictionary). The results were not dramatically different although higher redundancy
does give somewhat better results, as expected.
As shown in Fig. 6.11, the relative performance of the different methods is similar to the $32 \times 32$ case. When working on images of $64^2$ pixels, the advantage of Contourlets over Wavelets is slightly bigger. The Trainlets again give the best approximation performance, giving a glimpse into the potential gains achievable when training can be effectively done at larger signal scales. Finally, it is not possible to show here the complete trained dictionary, but we do include some selected atoms from it in Fig. 6.12. We obtain many different types of atoms: from the very local curvelets-like atoms, to more global Fourier atoms, and more.

### 6.5 Conclusion

This work shows that dictionary learning can be up-scaled to tackle a new level of signal dimensions. Using two-dimensional Separable Cropped Wavelets as a base-dictionary within the Double Sparsity model, allows this approach to now handle larger and larger signals. In order to handle the vast data sets needed to train such a big model, we propose an Online Sparse Dictionary Learning algorithm, employing SGD ideas in the
dictionary learning task. We show how, using these methods, dictionary learning is no longer limited to small signals, and can now be applied to obtained Trainlets, high dimensional trainable atoms.

While OMP proved sufficient for the experiments shown in this work, considering other sparse coding algorithms might be beneficial. In addition, the entire learning algorithm was developed using a strict $l_0$ pseudo-norm, and its relaxation to other convex norms opens new possibilities in terms of training methods. Another direction is to extend our model to allow for the adaptability of the separable base-dictionary itself, incorporating ideas of separable dictionary learning thus providing a completely adaptable structure. Understanding quantitatively how different parameters affect the learned dictionaries, such as redundancy and atom sparsity, will provide a better understanding of our model. These questions, among others, are part of ongoing work.

Figure 6.11: Experiment 6: M-term approximation of general image patches of size $64 \times 64$ for different methods.
Figure 6.12: Experiment 6: Subset of the general (sparse) dictionary for patches of size $64 \times 64$ obtained with OSDL trained over 10 million patches from natural images.
Bibliography


Demonstrating the usefulness of Separable Cropped Wavelets* on natural images, we find that they achieve superior results compared to standard methods in terms of quality and runtime. We show that Sparse K-SVD algorithms, such as OSDL, are able to capture these benefits efficiently and robustly in real-world scenarios. Additionally, we demonstrate that Sparse K-OSDL and Cropped Wavelets* are able to provide a fast and accurate solution to the SVD problem, which is crucial for efficient image processing.

[SOZE16] Forsyth et al.
Online Sparse Dictionary Learning

A problem that arises when learning a dictionary for large-scale images is that the dictionary generation method is not effective.

One effective way to approach this problem is by using the model of cropped wavelets [4]. The model is a modified K-SVD algorithm that uses a pre-defined wavelet transform for the training data. The cropped wavelets are used to represent the patches of the images.

The model is trained using a stochastic gradient descent algorithm that updates the dictionary parameters online. The algorithm uses a hard thresholding technique to update the dictionary coefficients. This method is more efficient than traditional K-SVD algorithms.

The cropped wavelets are used to represent the patches of the images. This method is more effective than traditional wavelet transforms, which are not suitable for large-scale images. The cropped wavelets are used to represent the patches of the images.

The cropped wavelets are used to represent the patches of the images. This method is more effective than traditional wavelet transforms, which are not suitable for large-scale images. The cropped wavelets are used to represent the patches of the images.
K-SVD on Wavelets

\[ D_{eq} = W_s D \]

features of Wavelets. The dictionary Wavelets is given by an expansion of the dictionary elements \( W_s \) and the coefficients \( D \) in the following manner:

\[ D_{eq} = W_s D \]

coefficients are obtained by solving a least-squares problem.

The dictionary Wavelets is a linear combination of the elements \( W_s \) and the coefficients \( D \) in the following manner:

\[ D_{eq} = W_s D \]

coefficients are obtained by solving a least-squares problem.

The dictionary Wavelets is a linear combination of the elements \( W_s \) and the coefficients \( D \) in the following manner:

\[ D_{eq} = W_s D \]

coefficients are obtained by solving a least-squares problem.
תקציר

motivation of the research is the extension of wavelet models. at the heart of the approach is the adaptation of models to the data. the basic property of the data is the scale of the basic model. therefore, the model should be used to represent the data with this property.

Motivation: the increase of local processing techniques is reaching the edge of possibility, at least in some applications such as image compression.

the globular model makes it possible to break through this barrier.

all this leads to a paradigm of research that is not limited by the scale, but also by the images. this work took several years of work on the development of the wavelets (Wavelets).

background: the domain of the wavelet models is divided, in our view, into two major groups:

1. wavelet models (analytic wavelets)
2. wavelet models (learned wavelets)

Analytic wavelets are fixed models, created from a closed formula. these models are used in various algorithms due to mathematician properties. in our work, many of these models are used.

the disadvantages of these models is that they are not adaptable to all families; therefore, their performances are not optimal. learned wavelets are created from examples. this property is possible in good implementations, often in many implementations.

price in operation, storage is required for the wavelet. in addition, high power consumption of learned wavelets requires many training examples.

in our work, we are aiming to achieve a good balance between the two worlds. the presented work presents new ways to use the learned wavelets with the advantages of the unstructured wavelets.
המחקר בוצע בהנחייתו של פרופ' מיכאל אלעד, בפקולטה למדעי המחשב.

תודה

ברצוני להודות למכה, פרופ' מיכאל אלעד על הנחייתו המופיעה לכל אנדר החותר במיתוס ועל אמונתו-belief-ומתנתו של מחקר עלי וברצוני להודות למכון למדעי מחשב, פרופ', מיכאל אלעד על תרומתו, והודות רבה למסך עלי ולמי התוכנית להודד בἝλληνικά פורום לкрутנוטה, על עלי וברצוני להודות על תרומתם של ד"ר מיכאל ציבולסקי ולדר שים-ששל hindi police. על עלי וברצוני להודות על תרומתם של רן גלעד ודוד אלפרע על השיחות הרבות והעורכות של הכסף, וברצוני להודות על תרומת רביה של מדענות של רותי בומויוד על הנסותי ענומת, שלמי אני לארץ.

מסתגר בчувתי ובאמניה גוזרים בני גיר).

הכרת תודה מסורה למכון מחקר זו.
לימוד מילונים מורבי סקלה
לשם
יזוג דלי על תמונות

תפקיד על מחקר

שלימים מחולק של הדרישות וключения התואר
ודקטר לפילוסופיה

בעז אופיר

הונง לנט טכנולוגיה - מכון טכנולוגי לישראל
ארד ב' התחילה חיפה מאי 2016
לימוד מילונים מורבי סקלה
לשם
ייצוג דיליל של תמונות

בעז אופיר