Sparse learned representations for image restoration

Julien Mairal[1] Michael Elad[2] Guillermo Sapiro[3]

[1] INRIA, Paris-Rocquencourt, Willow project - INRIA/ENS/CNRS UMR 8548, Paris, FRANCE
E-mail: julien.mairal@inria.fr
[2]The Technion - Israel Institute of Technology, Haifa 32000, ISRAEL
E-mail: elad@cs.technion.ac.il
[3]University of Minnesota, Minneapolis MN 55455, USA
E-mail: guille@ece.umn.edu

Keywords: sparsity, image processing, sparse coding, color, image restoration

Sparse representations of signals have drawn considerable interest in recent years. The assumption that natural signals, such as images, admit a sparse decomposition over a redundant dictionary leads to efficient algorithms for handling such sources of data. In particular, the design of well adapted dictionaries for images has been a major challenge. The K-SVD has been recently proposed for this task (Aharon et al. (2006)) and shown to perform very well for various grayscale image processing tasks. In this paper, we address the problem of learning dictionaries for color images and extend the K-SVD-based grayscale image denoising algorithm that appears in (Elad and Aharon (2006)). This work (Mairal et al. (2008a)) puts forward ways for handling nonhomogeneous noise and missing information, paving the way to state-of-the-art results in applications such as color image denoising, demosaicking, and inpainting. A multiscale extension of these algorithms (Mairal et al. (2008b)), which has led to state-of-the-art results in denoising, is also briefly presented.

1 Introduction

Consider a signal $\mathbf{x} \in \mathbb{R}^n$. We say that it admits a sparse approximation over a dictionary $\mathbf{D} \in \mathbb{R}^{n \times k}$, composed of k elements referred to as atoms, if one can find a linear combination of a "few" atoms from $\mathbf{D}$ that is "close" to the signal $\mathbf{x}$. The so-called *Sparseland* model suggests that such dictionaries exist for various classes of signals, and that the sparsity of a signal decomposition is a powerful model in many image processing applications (Elad and Aharon (2006), Mairal et al. (2008a;b)).

For natural images, dictionaries such as wavelets of various sorts (Mallat (1999)), curvelets, contourlets, wedgelets, bandlets and steerable wavelets are all attempts to design dictionaries that fulfill the above model assumption.

Aharon et al. (2006) introduce the K-SVD algorithm, a way to learn a dictionary, instead of exploiting pre-defined ones as described above, that leads to sparse representations on training signals. The follow-up work reported in (Elad and Aharon (2006)) proposes a novel and highly effective image denoising algorithm for the removal of additive white Gaussian noise with gray-scale images. Their proposed method includes the use of the K-SVD for learning the dictionary from the noisy image directly.

In this paper, we extend the algorithm reported in (Elad and Aharon (2006)) to color images (and to vector-valued images in general), and then show the applicability of this extension to other inverse problems in color image processing. The extension to color can be easily performed by a simple concatenation of the RGB values to a single vector and training on those directly, which gives already better results than denoising each channel separately. However, such a process produces false colors and artifacts, which are typically encountered in color image processing. This work, presented in (Mairal et al. (2008a)) also describes an extension of the denoising algorithm to the proper handling of non-homogeneous noise, inpainting and demosacking and we present briefly the multiscale extension from Mairal et al. (2008b).

2 Modified K-SVD algorithms for various image processing tasks

2.1 The original K-SVD algorithm

In this section, we briefly review the main ideas of the K-SVD framework for sparse image representation and denoising. The reader is referred to (Elad and Aharon (2006), Mairal et al. (2008a)) for more details.

Let $\mathbf{x}_0$ be a clean image and $\mathbf{y} = \mathbf{x}_0 + \mathbf{w}$ its noisy version with $\mathbf{w}$ being an additive zero-mean white Gaussian noise with a known standard deviation σ. The algorithm aims at finding a sparse approximation of every $\sqrt{n} \times \sqrt{n}$ overlapping patch of $\mathbf{y}$, where n is fixed a-priori. This representation is done over an adapted dictionary $\mathbf{D}$, learned for this set of patches. These approximations of patches are averaged to obtain the reconstruct image. This algorithm can be described as the minimization of an energy:

$$\left\{\hat{\alpha_{ij}}, \hat{\mathbf{D}}, \hat{\mathbf{x}}\right\} = \underset{\mathbf{D},\alpha_{ij},\mathbf{x}}{\arg\min}\, \lambda||\mathbf{x}-\mathbf{y}||_2^2 + \sum_{i,j} \mu_{ij}||\alpha_{ij}||_0 + \sum_{ij} ||\mathbf{D}\alpha_{ij} - \mathbf{R}_{ij}\mathbf{x}||_2^2 \ . \tag{1}$$

In this equation, $\hat{\mathbf{x}}$ is the estimator of $\mathbf{x}_0$, and the dictionary $\hat{\mathbf{D}} \in \mathbb{R}^{n\times k}$ is an estimator of the optimal dictionary which leads to the sparsest representation of the patches in the recovered image. The indices $[i, j]$ mark the location of the patch in the image (representing it's top-left corner). The vectors $\hat{\alpha}_{ij} \in \mathbb{R}^k$ are the sparse representations for the $[i, j]$-th patch in $\hat{\mathbf{x}}$ using the dictionary $\hat{\mathbf{D}}$. The notation $||.||_0$ is the ℓ^0 quasi-norm, a sparsity measure, which counts the number of non-zero elements in a vector. The operator $\mathbf{R}_{ij}$ is a binary matrix which extracts the square $\sqrt{n} \times \sqrt{n}$ patch of coordinates $[i, j]$ from the image written as a column vector. The scalars μ_{ij} are not known explicitely, but are estimated using the learning procedure.

The main steps of the K-SVD algorithm are (refer to Figure 1):

- *Sparse Coding Step:* This is performed with Forward Selection (Weisberg (1980)) (also known as Orthogonal Matching Pursuit (OMP) (Davis et al. (1994)). Note that this greedy algorithm provides only an approximated solution of Equation (2). Indeed, the non-convexity of the functional we are considering makes this problem difficult. Indeed, it can be proved that the ℓ^0-approximation problem is in general NP-hard. A well-known approach is the Basis Pursuit (Chen et al. (1998)), which suggests a convexification of the problem by using the ℓ^1 norm instead of ℓ^0.
- *Dictionary Update:* This is a sequence of rank-one approximation problems that update both the dictionary atoms and the sparse representations that use it, one at a time.
- *Reconstruction:* The last step is a simple averaging between the patches' approximations and the noisy image. The denoised image is $\hat{\mathbf{x}}$. Equation (4) emerges directly from the energy minimization in Equation (1).

2.2 Denoising of color images

The simplest way to extend the K-SVD algorithm to the denoising of color images is to denoise each single channel using a separate algorithm with possibly different dictionaries. However, our goal is to take advantage of the learning capabilities of the K-SVD algorithm to capture the correlation between the different color channels. We will show in our experimental results that a joint method outperforms this trivial plane-by-plane denoising. Recall that although in this paper we concentrate on color images, the key extension components here introduced are valid for other modalities of vector-valued images, where the correlation between the planes might be even stronger.

The problem we address is the denoising of RGB color images, represented by a column vector $\mathbf{y}$, contaminated by some white Gaussian noise $\mathbf{w}$ with a known deviation σ, which has been added to each channel. Color-spaces such as YCbCr, Lab, and other Luminance/Chrominance separations are often used in image denoising because it is natural to handle the chroma and luma layers differently, and also because the ℓ_2-norm in these spaces is more reliable and better reflects the human visual system's perception. However, in this work we choose to stay with the original RGB space, as any color conversion changes the structure of the noise.

In order to keep a reasonable computational complexity for the color extensions presented in this work, we use dictionaries that are not particularly larger than those practiced in the gray-scale version of the algorithm. More specifically, in (Elad and Aharon (2006)) the authors use dictionaries with 256 atoms and patches of size 8×8. Applying directly the K-SVD algorithm on (three dimensional) patches of size $8 \times 8 \times 3$ (containing the RGB layers) with 256 atoms, leads already to substantially better results than denoising each channel separately. However, this direct approach produces artifacts – especially a tendency to reduce the color saturation in the reconstruction. We observe that during the algorithm, the greedy sparse coding procedure is likely to follow the "gray" axis, which is the axis defined by $r = g = b$ in the RGB color space.

Parameters: λ; C (noise gain); J (number of iterations); k (number of atoms); n (size of the patches).
Initialization: Set $\mathbf{x} = \mathbf{y}$; Initialize $\mathbf{D} = (\mathbf{d}_l \in \mathbb{R}^{n\times 1})_{l\in 1\ldots k}$ (e.g., redundant DCT).
Loop: Repeat J times

- *Sparse Coding:* Fix $\mathbf{D}$ and use OMP to compute coefficients $\alpha_{ij} \in \mathbb{R}^{k\times 1}$ for each patch by solving:

$$\forall ij \quad \alpha_{ij} = \arg\min_{\alpha} ||\alpha||_0 \text{ subject to } ||\mathbf{R}_{ij}\mathbf{x} - \mathbf{D}\alpha||_2^2 \leq n(C\sigma)^2. \tag{2}$$

- *Dictionary Update:* Fix all α_{ij}, and for each atom $\mathbf{d}_l$, $l \in 1, 2, \ldots, k$ in $\mathbf{D}$,
  - Select the set of patches ω_l that use this atom, $\omega_l := \{[i,j] | \alpha_{ij}(l) \neq 0\}$.
  - For each patch $[i,j] \in \omega_l$, compute its residual without the contribution of the atom $\mathbf{d}_l$: $e_{ij}^l = \mathbf{R}_{ij}\mathbf{x} - \mathbf{D}\alpha_{ij} + \mathbf{d}_l\alpha_{ij}(l)$.
  - Set $\mathbf{E}_l \in \mathbb{R}^{n\times|\omega_l|}$ as the matrix whose columns are the $\mathbf{e}_{ij}^l$, and $\alpha^l \in \mathbb{R}^{|\omega_l|}$ the vector whose elements are the $\alpha_{ij}(l)$.
  - Update $\hat{\mathbf{d}}_l$ and the $\alpha_{ij}(l)$ by minimizing:

$$(\mathbf{d}_l, \alpha^l) = \arg\min_{||\mathbf{d}||_2=1,\beta} ||\mathbf{E}_l - \mathbf{d}\beta^T||_F^2. \tag{3}$$

    This rank-one approximation is performed by a truncated SVD of $\mathbf{E}_l$. Here, $\mathbf{E}_l - \mathbf{d}\beta^T$ represents the residual error of the patches from ω_l if we replace $\mathbf{d}_l\alpha^{lT}$ by $\mathbf{d}\beta^T$ in their decompositions.

Reconstruction: Perform a weighted average:

$$\hat{\mathbf{x}} = \Big(\lambda\mathbf{I} + \sum_{ij}\mathbf{R}_{ij}^T\mathbf{R}_{ij}\Big)^{-1}\Big(\lambda\mathbf{y} + \sum_{ij}\mathbf{R}_{ij}^T\mathbf{D}\alpha_{ij}\Big). \tag{4}$$

Figure 1: The single-scale K-SVD-based grayscale image denoising algorithm.

Before proceeding to explain the proposed solution to this color bias and washing effect, let us explain why it happens. As mentioned before, this effect can be seen in the results in (McAuley et al. (2006)), although it has not been explicitly addressed there.

First, at the intuitive level, relatively small dictionaries within the order of 256 atoms, for example, are not rich enough to represent the diversity of colors in natural images. Therefore, training a dictionary on a generic database leads to many gray or low chrominance atoms which represent the basic spatial structures of the images. This behavior can be observed in Figure 2. This result is not unexpected since this global dictionary is aiming at being generic. This predominance of gray atoms in the dictionary encourages the image patches approximation to "follow" the gray axis by picking some gray atoms, and this introduces a bias and color washing in the reconstruction. Examples of such color artifacts resulting from global dictionaries are presented also in Figure 2. Using an adaptive dictionary tends to reduce but not eliminate these artifacts. One might be tempted to solve the above problem by increasing k and thus adding redundancy to the dictionary. This, however, is counter productive in two important ways - the obtained algorithm becomes computationally more demanding, and as the images we handle are getting close in size to the dictionary, over-fitting in the learning process in unavoidable.

We address this color problem by changing the metric during the Sparse Coding stage as it will be explained shortly. Forward selection is a greedy algorithm that aims to approximate a solution of Equation (2). It consists of selecting at each iteration the best atom from the dictionary, which is the one that maximizes its inner product with the residual (minimizing the error metric), and then updating the residual by performing an orthogonal projection of the signal one wants to approximate onto the vectorial space generated by the previously selected atoms. This orthogonalization is important since it gives more stability and a faster convergence for this greedy algorithm.

An additional, more formal way to explain the lack of colors and the color bias in the reconstruction is to note that Forward Selection does not guarantee that the reconstructed patch will maintain the average color of the original one. Therefore, the following relationship for the patch ij, $\mathbb{E}(\mathbf{R}_{ij}\mathbf{y}) = \mathbb{E}(\mathbf{R}_{ij}\mathbf{x}_0 + \mathbf{R}_{ij}\mathbf{w}) = \mathbb{E}(\mathbf{R}_{ij}\hat{\mathbf{x}})$, does not necessarily hold. If the diversity of colors is not important enough in the dictionary, the pursuit is likely to follow some other direction in the patches' space. But with color images and the corresponding increase in the dimensionality, our experiments show that this is clearly the case. To address this, we add a term during Forward selection which tends to minimize also the bias between the input image and the reconstruction on each channel. Considering that $\mathbf{y}$ and $\mathbf{x}$ are two patches written as column vectors $(R, G, B)^T$, we define a new inner product to be used in the OMP step:

$$< \mathbf{y}, \mathbf{x} >_\gamma = \mathbf{y}^T\mathbf{x} + \frac{\gamma}{n^2}\mathbf{y}^T\mathbf{K}^T\mathbf{K}\mathbf{x} = \mathbf{y}^T(\mathbf{I} + \frac{\gamma}{n}\mathbf{K})\mathbf{x}, \tag{5}$$

where K is a binary matrix so that $\frac{1}{n}\mathbf{K}\mathbf{x}$ is a constant $n \times n \times 3$ patch, filled with the average color of the patch $\mathbf{x}$. γ is a new parameter which can be tuned to increase or discard this correction. We empirically fixed this parameter to $\gamma = 5.25$. The first term in this equation is the ordinary Euclidean inner product. The second one takes into account average colors. Examples of results from our algorithm are presented on Figure 2, with different values for this parameter γ, illustrating the efficiency of our approach in reducing the color artifacts.

2.3 Non-homogeneous noise, inpainting and demosaicking

The problem of handling non-homogenous noise is very important as non-uniform noise across color channels is very common in digital cameras. In (Mairal et al. (2008a)), we presented a variation of the K-SVD, which permits to address this issue. Within the limits of this model, we were able to fill-in relatively small holes in the images and we presented state-of-the-art results for image demosaicking.

Let us now consider the case where $\mathbf{w}$ is a white Gaussian noise with a different standard deviation $\sigma_p > 0$ at each location p. Assuming these standard deviations are known, we introduced a vector β composed of weights for each location:

$$\beta_p = \frac{\min_{p' \in \text{ Image }} \sigma_{p'}}{\sigma_p}. \tag{6}$$

This leads us to define a weighted K-SVD algorithm based on a different metric for each patch. Since the fine details of this approach are given in (Mairal et al. (2008a)), we restrict the discussion here to a rough coverage of the main idea.

(a) $8 \times 8 \times 3$ patches (b) Original (c) Original algorithm, $\gamma = 0$, PSNR=28.78 dB (d) Proposed algorithm, $\gamma = 5.25$, PSNR=30.04 dB

Figure 2: On the left: A dictionary with 256 atoms learned on a generic database of natural images, Note the large number of color-less atoms. Since the atoms can have negative values, the vectors are presented scaled and shifted to the $[0, 255]$ range per channel. On the right: Examples of color artifacts while reconstructing a damaged version of the image 2(b) without the improvement here proposed ($\gamma = 0$ in the new metric). Color artifacts are reduced with our proposed technique ($\gamma = 5.25$ in our proposed new metric). Both images have been denoised with the same global dictionary. In 2(c), one observes a bias effect in the color from the castle and in some part of the water. What is more, the color of the sky is piecewise constant when $\gamma = 0$ (false contours), which is another artifact our approach corrected. These images are taken from (Mairal et al. (2008a)) under copyright ©2008 IEEE.

Using the same notations as in (Mairal et al. (2008a)), we denote by $\otimes$ the element-wise multiplication between two vectors, which we use to apply the above vector β as a "mask".

We aim at solving the following problem, which replaces Equation (1):

$$\left\{\hat{\alpha_{ij}}, \hat{\mathbf{D}}, \hat{\mathbf{x}}\right\} = \underset{\mathbf{D},\alpha_{ij},\mathbf{x}}{\arg\min}\, \lambda||\beta \otimes (\mathbf{x} - \mathbf{y})||_2^2 + \sum_{ij} \mu_{ij}||\alpha_{ij}||_0 + \sum_{ij} ||(\mathbf{R}_{ij}\beta) \otimes (\mathbf{D}\alpha_{ij} - \mathbf{R}_{ij}\mathbf{x})||_2^2 \,. \quad (7)$$

As explained in (Mairal et al. (2008a)), there are two main modifications in the minimization of this energy. First, the *Sparse Coding* step takes the matrix β into account by using a different metric within the OMP. Second, the *Dictionary Update* variation is more delicate and uses a weighted rank-one approximation algorithm. Inpainting consists of filling-in holes in images. Within the limits of our model, it becomes possible to address a particular case of inpainting. By considering small random holes as areas with infinite power noise, one can design a mask β with zero values for the missing pixels, and apply it as an non-homogeneous denoising problem. This approach proves to be very efficient. This inpainting case could also be considered as a specific case of interpolation. The mathematical formulation from Equation (7) remains the same, but some values from the matrix β are just equal to zero. Details about this are provided in (Mairal et al. (2008a)), together with a discussion on how to handle the demosaicking problem that has a fixed and periodic pattern of missing values.

2.4 Multiscale extension

In our proposed multiscale framework, we focus on the use of different sizes of atoms simultaneously. Considering the design of a patch-based representation and a denoising framework, we put forward a simple quadtree model of large patches. This is a classical data structure, also used in wedgelets for example (Donoho (1998)). A fixed number of scales, N, is chosen, such that it corresponds to N different sizes of atoms. A large patch of size n pixels is divided along the tree to sub-patches of sizes $n_s = \frac{n}{4^s}$, where $s = 0 \dots N-1$ is the depth in the tree. Then, one different dictionary $\mathbf{D}_s \in \mathbb{R}^{n_s \times k_s}$ composed of k_s atoms of size n_s is learned and used per each scale s.

The overall idea of the multiscale algorithm we propose stays as close as possible to the original K-SVD algorithm, with an attempt to exploit the several existing scales. This aims at solving the same energy minimization problem of Equation (1), with a multiscale structure embedded within the dictionary $\mathbf{D} \in \mathbb{R}^{n \times k}$, which is a joint one, composed of all the atoms of all the dictionaries $\mathbf{D}_s$ located at every

possible position in the quadtree. For the scale s, there exists 4^s such positions. This makes a total of $k = \sum_{s=0}^{N-1} 4^s k_s$ atoms in $\mathbf{D}$. This is illustrated in Figure 3, where an example of a possible multiscale decomposition is presented. One can see on this Figure how the dictionary $\mathbf{D}$ has been build: The atoms of $\mathbf{D}_0$ of size n (for instance the ones associated with α_0 and α_1 on this picture) are also atoms of $\mathbf{D}$. Then, the atoms of $\mathbf{D}_1$ of size $\frac{n}{4}$ are embedded in bigger atoms of size n with zero-padding at four possible positions (see the atoms associated with $\alpha_2, \alpha_3, \alpha_4, \alpha_5$ on the figure). Then, the same idea applies to $\mathbf{D}_2$ and so on.

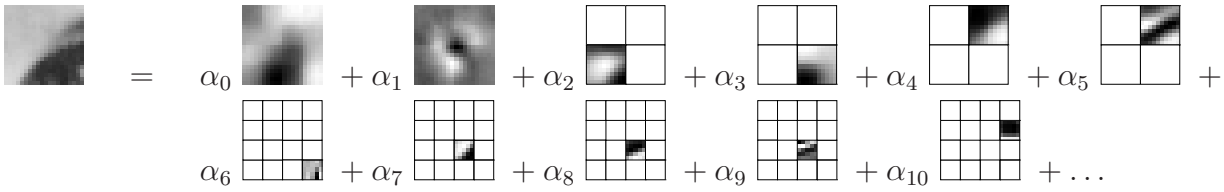


Figure 3: Possible decomposition of a 20×20 patch with a 3-scales dictionary. Image from (Mairal et al. (2008b)) under copyright ©2008 SIAM

3 Experimental validation

3.1 Grayscale image denoising

We now present denoising results obtained within the proposed sparsity framework. In Table 1, our results for $N = 1$ (single-scale) and $N = 2$ scales are carefully compared to the original K-SVD algorithm (Matalon et al. (2005), Elad and Aharon (2006)) and the recent state-of-the-art results reported in (Dabov et al. (2007b)) on a set of 6 standard images (see (Mairal et al. (2008b)) for details). The best results are shared between our algorithm and (Dabov et al. (2007b)). As it can be observed, the differences are insignificant. Our average performance is better for $\sigma \leq 10$ and for $\sigma = 50$, while the results from (Dabov et al. (2007b)) are slightly better or similar to ours for other noise levels.

The PSNR values in Table 1, corresponding to the results in (Dabov et al. (2007b), Matalon et al. (2005), Elad and Aharon (2006)) and our algorithm, are averaged over 5 experiments for each image and each level of noise, to cope with the variability of the PSNR with the different noise realizations.

(a) Original boat (b) Noisy, $\sigma = 50$ (c) Result (d) Original lena (e) Noisy, $\sigma = 10$ (f) Result

(g) Original house (h) Noisy, $\sigma = 10$ (i) Result (j) Ori. cameraman (k) Noisy, $\sigma = 15$ (l) Result

Figure 4: Examples of denoising results using two scales. The images are taken from (Mairal et al. (2008b)) and are under copyright ©2008 SIAM.

Table 1: PSNR results over 6 standard images of our denoising algorithm compared with some other ones. Each cell is divided into four parts. The top-left part shows the results from the original K-SVD (Aharon et al. (2006)), the top-right from the most recent state-of-the-art (Dabov et al. (2007b)). The bottom-left is devoted to our results for $N = 1$ scale and the bottom-right to $N = 2$ scales. Each time the best results are in bold.

σ	house		peppers		cameraman		lena		barbara		boat	
5	39.37	39.82	37.78	38.09	37.87	38.26	38.60	38.73	38.08	38.30	37.22	37.28
	39.81	**39.92**	38.07	**38.20**	38.12	**38.32**	38.72	**38.78**	**38.34**	38.32	**37.35**	**37.35**
10	35.98	36.68	34.28	**34.68**	33.73	34.07	35.47	**35.90**	34.42	**34.96**	33.64	33.90
	36.38	**36.75**	34.58	34.62	34.01	**34.17**	35.75	35.84	34.90	34.86	33.93	**33.98**
15	34.32	34.97	32.22	**32.70**	31.42	**31.83**	33.70	**34.27**	32.37	**33.08**	31.73	32.10
	34.68	**35.00**	32.53	32.47	31.68	31.72	34.00	34.14	32.82	32.96	32.04	**32.13**
20	33.20	**33.79**	30.82	**31.33**	29.91	**30.42**	32.38	**33.01**	30.83	**31.77**	30.36	**30.85**
	33.51	33.75	31.15	31.08	30.32	30.37	32.68	32.88	31.37	31.53	30.74	30.82
25	32.15	**32.87**	29.73	**30.19**	28.85	**29.40**	31.32	**32.06**	29.60	**30.65**	29.28	**29.84**
	32.39	32.83	30.03	30.04	29.28	39.37	31.63	31.92	30.17	30.29	29.67	29.82
50	27.95	**29.45**	26.13	26.35	25.73	25.86	27.79	**28.86**	25.47	**27.14**	25.95	26.56
	28.24	29.40	26.34	**26.64**	26.06	**26.17**	28.15	28.80	26.08	26.78	26.36	**26.74**
100	23.71	**25.43**	21.75	**22.91**	21.69	22.62	24.46	**25.51**	21.89	**23.49**	22.81	23.64
	23.83	24.84	21.94	22.64	22.05	**22.84**	24.49	25.06	22.07	22.95	22.96	**23.67**

3.2 Color image denoising

We present an example of denoising a non uniform noise on Figure 5, where a white Gaussian noise has been added to each data, but with a different known standard deviation, and a denoising experiment with a uniform noise of standard deviation $\sigma = 25$.

(a) Original (b) Noisy (c) Denoised (d) Original (e) Noisy, $\sigma = 25$ (f) Denoised

Figure 5: On the left: Example of non-spatially-uniform white Gaussian noise. For each image pixel, σ takes a random value from a uniform distribution in $[1; 101]$. The σ values are assumed to be known by the algorithm. The initial PSNR was 12.83 dB, the resulting PSNR is 30.18 dB. On the right: Example of denoising a uniform white Gaussian noise with $\sigma = 25$.

Quantitative experiments, shown in (Mairal et al. (2008b)), show that our method lead to state-of-the-art results, very close to (Dabov et al. (2007a)). We report these quantitative results in Table 2 for 5 standard images.

3.3 Inpainting

For illustrative purposes, we show an example from (Mairal et al. (2008b)), obtained with $N = 2$ scales in Figure 6. This result is quite impressive bearing in mind that it is able to retrieve the brick texture of the wall, something that our visual system is not able to do. In this example, the multiscale version provides an improvement of $2.24dB$ over the single-scale algorithm.

Table 2: PSNR results for our color image denoising experiments. Each cell is composed of four parts: The top-left is devoted to (Dabov et al. (2007a)), the top-right to our 2-scales gray image denoising method applied to each R,G,B channel independently, the bottom-left to the color denoising algorithm with $N = 1$ scale and the bottom-right to our algorithm with $N = 2$ scales. Each time the best results are in bold.

σ	castle		mushroom		train		horses		kangaroo		**Average**	
5	**40.84**	38.27	40.20	37.65	39.91	36.52	**40.46**	37.17	39.13	35.73	40.11	37.07
	40.77	40.79	**40.26**	**40.26**	**40.04**	40.03	40.44	40.45	**39.26**	39.25	40.15	**40.16**
10	36.61	34.25	**35.94**	33.46	34.85	31.37	**35.78**	32.70	34.29	31.20	35.49	32.60
	36.51	**36.65**	35.88	35.92	34.90	**34.93**	35.67	35.75	34.31	**34.34**	35.45	**35.52**
15	**34.39**	31.95	**33.61**	31.21	31.95	28.53	33.18	30.48	31.63	29.05	32.95	30.24
	34.22	34.37	33.51	33.58	31.98	**32.04**	33.11	**33.19**	31.71	**31.75**	32.91	**32.99**
20	**32.84**	30.52	**31.99**	29.74	29.97	26.79	31.44	29.13	29.85	27.77	31.22	28.79
	32.63	32.77	31.86	31.97	29.97	**30.01**	31.35	**31.47**	29.99	**30.07**	31.16	**31.26**
25	**31.68**	29.47	**30.84**	28.69	28.45	26.55	30.19	28.21	28.65	26.90	29.96	27.96
	31.45	31.59	30.67	30.75	28.50	**28.53**	30.19	**30.28**	28.82	**28.87**	29.93	**30.00**

(a) Original (b) Damaged (c) Restored, $N = 2$

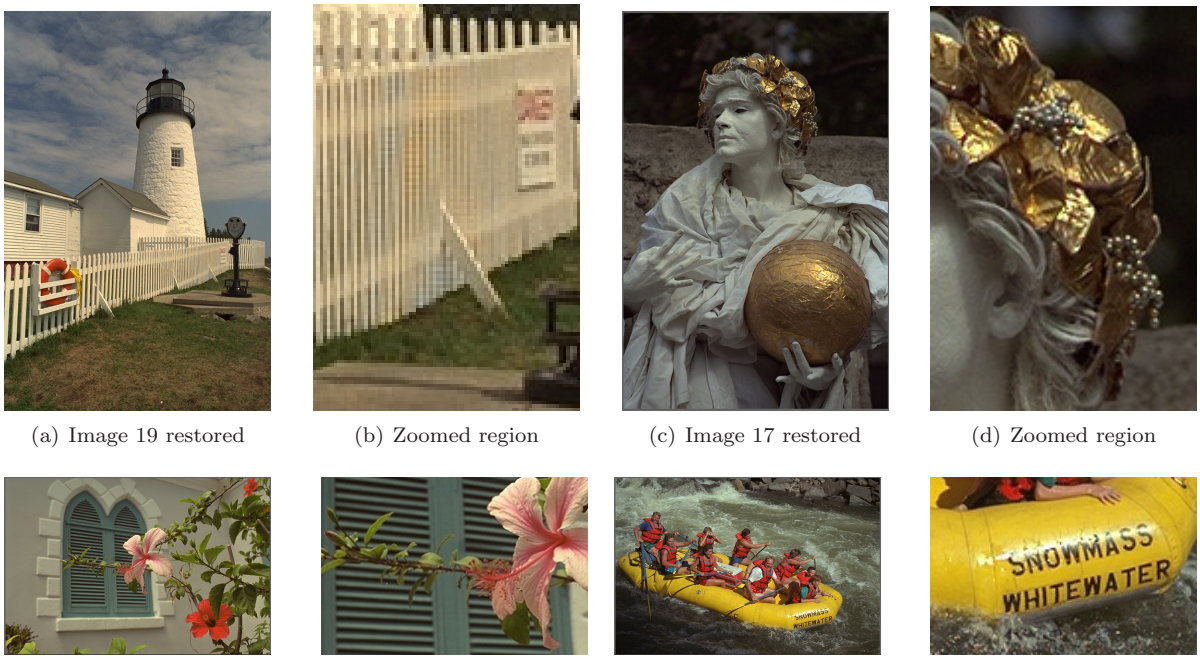
Figure 6: Inpainting using $N = 2$ and $n = 16 \times 16$ (third image), or $N = 1$ and $n = 8 \times 8$ (fourth image). $J = 100$ iterations were performed, producing an adaptive dictionary. During the learning, 50% of the patches were used. A sparsity factor $L = 10$ has been used during the learning process and $L = 25$ for the final reconstruction. The damaged image was created by removing 75% of the data from the original image. The initial PSNR is $6.13dB$. The resulting PSNR for $N = 2$ is $33.97dB$. The

3.4 Demosaicking

We now present results for demosaicking images. We ran our experiments on 24 images from the Kodak image database, which can be found in (Mairal et al. (2008a)). We simulated the mosaic effects using the Bayer pattern and compare the results in terms of PSNR using a globally trained dictionary with atoms of size $6 \times 6 \times 3$, with a patch-sparsity factor $L = 20$, and compare to bilinear interpolation, the results given by Kimmel's algorithm (Kimmel (1999)), three very recent methods and state-of-the-art results presented in (Chung and Chan (2006)).

As it is shown in (Mairal et al. (2008a)), for some very difficult regions of some images, our generic color image restoration method, when applied to demosaicking, does not give visually better results than the best interpolation-based methods, which are tuned to track the classical artifacts from the demosaicing problem. On the other hand, on average, our algorithm performs better than the state of the art (Chung and Chan (2006)), improving by a significant 0.39dB the average result on this standard dataset when compared to the best demosaicking algorithm so far reported.

Examples of visual results are presented on Figure 7.



(a) Image 19 restored (b) Zoomed region (c) Image 17 restored (d) Zoomed region

(e) Image 7 restored (f) Zoomed region (g) Image 14 restored (h) Zoomed region

Figure 7: *Examples of demosaicing results.*

4 Conclusion

In this paper we present various K-SVD based algorithms from (Mairal et al. (2008a;b)) that are able to learn multiscale sparse image representations for various image processing tasks. Using a shift-invariant sparsity prior on natural images, the proposed framework achieves state-of-the-art grayscale denoising, color denoising, inpainting and demosaicking results. We chose to work with natural color images because of their practical relevance and because it is a very convenient data source in order to compare our algorithm with other results available in the literature. One main characteristic we have not discussed in this paper is the capability of this algorithm to be adapted to other types of vectorial data. Future work will consist of testing the algorithm on multi-channel images such as LANDSAT. Another interesting future direction consists of learning dictionaries for specific classes of images such as MRI and astronomical data. We strongly believe that this framework could provide cutting edge results in modeling this kind of data as well.

Our current efforts are devoted in part to the design of faster algorithms, which can be used with any number of scales. One direction we are pursuing is to combine the K-SVD with image pyramids.

Acknowledgments

Work partially supported by NSF, ONR, NGA, DARPA, the McKnight Foundation, and the Israeli Science Foundation grant No. 796/05.

References

Aharon, M., Elad, M., and Bruckstein, A. M. (2006). The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Trans. SP*, 54(11):4311–4322.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61.

Chung, K.-H. and Chan, Y.-H. (2006). Color demosaicing using variance of color differences. *IEEE Transactions on Image Processing*, 15(10):2944–2955.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007a). Color image denoising by sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. In *Proc. ICIP*, San Antonio, Texas, USA.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007b). Image denoising by sparse 3d transform-domain collaborative filtering. *IEEE Trans. IP*, 16(8):2080–2095.

Davis, G. M., Mallat, S., and Zhang, Z. (1994). Adaptive time-frequency decompositions. *SPIE J. of Opt. Engin.*, 33(7):2183–2191.

Donoho, D. L. (1998). Wedgelets: Nearly minimax estimation of edges. *Annals of statistics*, 27(3):859–897.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. IP*, 54(12):3736–3745.

Kimmel, R. (1999). Demosaicing: image reconstruction from color ccd samples. *IEEE Transactions on Image Processing*, 8(9):1221–1228.

Mairal, J., Elad, M., and Sapiro, G. (2008a). Sparse representation for color image restoration. *IEEE Trans. IP*, 17(1):53–69.

Mairal, J., Sapiro, G., and Elad, M. (2008b). Learning multiscale sparse representations for image and video restoration. 7(1):214–241.

Mallat, S. (1999). *A Wavelet Tour of Signal Processing, Second Edition*. Academic Press, New York.

Matalon, B., Elad, M., and Zibulevsky, M. (2005). Improved denoising of images using modeling of the redundant contourlet transform. In *Proceedings of the SPIE conference wavelets*, volume 5914.

McAuley, J., Caetano, T., Smola, A., and Franz, M. O. (2006). Learning high-order mrf priors of color images. *Proc. 23rd Intl. Conf. Machine Learning (ICML 2006)*, pages 617–624.

Weisberg, S. (1980). *Applied Linear Regression*. New York.