# Expected Patch Log Likelihood with a Sparse Prior

Jeremias Sulam and Michael Elad

Computer Science Department, Technion, Israel
{jsulam,elad}@cs.technion.ac.il

**Abstract.** Image priors are of great importance in image restoration tasks. These problems can be addressed by decomposing the degraded image into overlapping patches, treating the patches individually and averaging them back together. Recently, the Expected Patch Log Likelihood (EPLL) method has been introduced, arguing that the chosen model should be enforced on the final reconstructed image patches. In the context of a Gaussian Mixture Model (GMM), this idea has been shown to lead to state-of-the-art results in image denoising and deblurring. In this paper we combine the EPLL with a sparse-representation prior. Our derivation leads to a close yet extended variant of the popular K-SVD image denoising algorithm, where in order to effectively maximize the EPLL the denoising process should be iterated. This concept lies at the core of the K-SVD formulation, but has not been addressed before due the need to set different denoising thresholds in the successive sparse coding stages. We present a method that intrinsically determines these thresholds in order to improve the image estimate. Our results show a notable improvement over K-SVD in image denoising and inpainting, achieving comparable performance to that of EPLL with GMM in denoising.

**Keywords:** K-SVD, EPLL, MAP, Sparse Representations, Image Restoration.

## 1 Introduction

Inverse problems in image processing consist of recovering an original image that has been degraded. Denoising, debluring and inpaiting are specific and common such examples. Put formally, these problems attempt to recover an underlying image $\mathbf{x}$ given the measurement $\mathbf{y}$ such that

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{A}$ is a known linear operator and $\mathbf{n}$ represents measurement noise, assumed to be independent and normally distributed. In dealing with this problem, it is common to work with image priors as regularizers and develop a Maximum a Posteriori (MAP) estimator for the unknown image $\hat{\mathbf{x}}$. This can be formulated as an optimization problem where we look for an estimate which is close enough

to the measured image while being likely under this prior. Most state of the art methods employ, either implicitly of explicitly, some prior knowledge of this form [6, 10, 8, 4].

Learning specifics priors from real data has shown to enable better performance under this approach [7, 12]. However, this learning process is computationally hard and it is usually restricted to small dimensions, which leads naturally to the modeling of small image patches [1, 15]. Such methods attempt to address the image restoration problem by breaking the image into small overlapping patches, solving their MAP estimate, and tiling the results back together by averaging them [6, 4, 3]. While this is a common and practical strategy, it is also known to cause visible texture-like artifacts in the final image. Recently, Zoran and Weiss [16] proposed a general framework based on the simple yet appealing idea that the *resulting final* patches should be likely under some specific prior, and not the intermediate ones. Their approach is based on maximizing the *Expected Patch Log Likelihood* (EPLL) which yields the average likelihood of a patch on the final image under some prior. This idea is general in the sense that it can be applied to any patch-based prior for which a MAP estimator can be formulated. In particular, the authors in [16] employed the classic Gaussian Mixture Model prior achieving state of the art results in image denoising and deblurring.

The concept of sparsity is a recurring idea in most state of the art restoration methods; namely, a natural signal or image patch can be well represented by a linear combination of a few atoms from a dictionary [2, 8]. This leads to the natural question, could we use the EPLL framework with a sparsity-inspired prior? If so, how is this related to existing methods that explicitly target this problem and what is there to gain from this approach? In this paper we explore and formally address these questions, showing that indeed benefit can be found in employing EPLL with a patch sparsity-based prior.

## 2 Expected Patch Log Likelihood

We begin by briefly reviewing the EPLL framework as described in [16]. Given an image $\mathbf{x}$, the Expected Patch Log Likelihood under some prior $p$ is defined as

$$EPLL_p(\mathbf{x}) = \sum_i log\ p(\mathbf{P}_i\mathbf{x}), \tag{2}$$

where $\mathbf{P}_i$ extracts the $i^{th}$ patch from $\mathbf{x}$. Therefore, given the corruption model in Eq. (1) we can propose to minimize the following cost function:

$$f_p(\mathbf{x}|\mathbf{y}) = \frac{\lambda}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||_2^2 - EPLL_p(\mathbf{x}), \tag{3}$$

where the first term represents the log likelihood of the image. To get around the hard optimization of this function, the authors in [16] propose to use a *Half*

*Quadratic Splitting* strategy by defining auxiliary patches $\{\mathbf{z}^i\}$ for each patch $\mathbf{P}_i\mathbf{x}$, and then minimizing

$$c_{p,\beta}(\mathbf{x}, \{\mathbf{z}^i\}|\mathbf{y}) = \frac{\lambda}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||_2^2 + \sum_i \frac{\beta}{2}||\mathbf{P}_i\mathbf{x} - \mathbf{z}^i||_2^2 - log\ p(\mathbf{z}^i) \tag{4}$$

iteratively, while increasing the value of $\beta$. Note that for $\beta \to \infty$, $\mathbf{z}^i \to \mathbf{P}_i\mathbf{x}$, so this parameter controls the distance between the auxiliary patches and the patches of the image $\mathbf{x}$. For a fixed value of $\beta$, the cost function is again broken into a two step inner minimization: first fix $\{\mathbf{z}^i\}$ and solve for $\mathbf{x}$ by

$$\mathbf{x} = \frac{\lambda\mathbf{A}^T\mathbf{y} + \beta\sum_i \mathbf{P}_i^T\mathbf{z}^i}{\lambda\mathbf{A}^T\mathbf{A} + \beta\sum_i \mathbf{P}_i^T\mathbf{P}_i}.^1 \tag{5}$$

Then, fix $\mathbf{x}$ and solve for $\{\mathbf{z}^i\}$ by solving the MAP estimate for each patch under the prior in consideration. This process should be repeated 4-5 times, before increasing $\beta$ and repeating the whole process again. Each time, the patches are taken from *the image estimate* at each iteration.

Within the EPLL scheme, the choice of $\beta$ is crucial. In [16] the authors set this parameter manually to be $\frac{1}{\sigma^2}[1, 4, 8, 16, 32, \dots]$, where $\sigma$ is the noise standard deviation. In the same work it is also suggested that $\beta$ could be determined as $\beta = \frac{1}{\sigma^2}$, where $\sigma$ is estimated in every iteration by an *off-the-shelf* white Gaussian noise estimator.

## 3 EPLL with a Sparse Prior

In the original formulation, Zoran and Weiss propose to use a Gaussian Mixture Model (GMM) prior which is learnt off-line from a large number of examples. In their case, the MAP estimator for each patch is simply given by the Wiener filter solution for the Gaussian component with the highest conditional weight [16]. However, the EPLL approach is a generic framework for potentially any patch-based prior. We now turn to explore the formulation of an equivalent problem with a sparsity inducing prior.

### 3.1 Cost function formulation

Consider the signal $\mathbf{z} = \mathbf{D}\alpha$, where $\mathbf{D}$ is a redundant dictionary of size $n \times m$ ($n < m$), and the vector $\alpha$ is sparse; i.e., $||\alpha||_0 \ll n$, where the $l_0$ pseudo-norm $||\cdot||_0$ basically counts the non zero elements in $\alpha$. Assuming that this is the model we impose on our patches $\mathbf{z}^i$, Eq. (4) becomes

$$c_{\mu,\beta}(\mathbf{x}, \{\alpha_i\}|\mathbf{y}) = \frac{\lambda}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||_2^2 + \sum_i \frac{\beta}{2}||\mathbf{D}\alpha_i - \mathbf{P}_i\mathbf{x}||_2^2 + \mu_i||\alpha_i||_0. \tag{6}$$

[1] Note that this an abuse of notation as the denominator is a diagonal matrix to be inverted.

In this case, $\mu_i$ reflects the trade-off between the accuracy of the representation and the sparsity of $\alpha_i$. For the case $\beta = 1$, this last expression corresponds exactly to the formulation of the K-SVD denoising algorithm in [6], where $\mathbf{A} = \mathbf{I}$. In this work, Elad and Aharon proposed to use a block-coordinate minimization that starts by fixing $\mathbf{x} = \mathbf{y}$, and then seeking the optimal $\alpha_i$ solving the MAP estimator for each patch:

$$\hat{\alpha}_i = \arg\min_{\alpha} \mu_i ||\alpha_i||_0 + ||\mathbf{D}\alpha_i - \mathbf{P}_i\mathbf{x}||_2^2. \tag{7}$$

Though this problem is NP-hard in general, its solution can be well approximated by greedy or pursuit algorithms [5]. In particular, the Orthogonal Matching Pursuit (OMP) [14] can be used with the noise energy as an error threshold to yield an approximation of the solution to Problem (7), and we employ this method in our work due to its simplicity and efficienty [13]. This way, $\mu_i$ is handled implicitly by replacing the second term by a constraint of the form

$$\min_{\alpha} ||\alpha||_0 \;\; \text{subject to} \;\; ||\mathbf{D}\alpha - \mathbf{P}_i\mathbf{x}||_2^2 \leq nc\sigma^2, \tag{8}$$

where $c$ is a constant factor set to 1.15 in [6]. Given the estimated sparse vectors $\{\hat{\alpha}_i\}$, the algorithm proceeds by updating for the unknown image $\mathbf{x}$ which results in an equivalent expression to that in Eq. (5) - for a specific value of $\beta$. When denoising is done locally (training the dictionary on the corrupted patches) the dictionary gets updated together with the sparse vectors by using a K-SVD step. This adaptive method that trains the dictionary on the noisy image itself has proven to be better than using a dictionary trained offline.

The initial claim in [6] is that the above block-coordinate minimization should be iterated. In practice, however, repeating this process is problematic since after updating $\mathbf{x}$, the noise level has changed and it is spatially varying. Therefore, the sparse coding stage has no known thresholds to employ. Thus, the algorithm in [6] does not iterate after updating $\mathbf{x}$.

Increasing $\beta$, as practised in [16], forces the distance $||\mathbf{D}\alpha_i - \mathbf{P}_i\mathbf{x}||_2$ to be smaller. Therefore, iterating the above algorithm for increasing values of $\beta$ is equivalent to iterating the process described for the K-SVD with smaller thresholds. As we see, the algorithm proposed in [6] applies only the first iteration of the EPLL scheme with a sparse-enforcing prior, therefore losing important denoising potential. A synthetic example is shown in Fig. 1, where we compare the algorithms in [6] and [16] with the method proposed in this paper.

We now turn to address the matter of the threshold design for later stages of the K-SVD in order to practice the EPLL concept in an effective way.

### 3.2 Sparse coding thresholds

Consider the threshold in the sparse coding stage, at each iteration $k$, to be $\nu_k^2$. Naturally, in the first iteration of the process that aims to minimize Eq. (6) we set this threshold to be exactly the noise energy $\sigma^2$ for all patches; i.e. $\nu_1^2 = \sigma^2$. In the following iterations, however, instead of trying to estimate the remaining

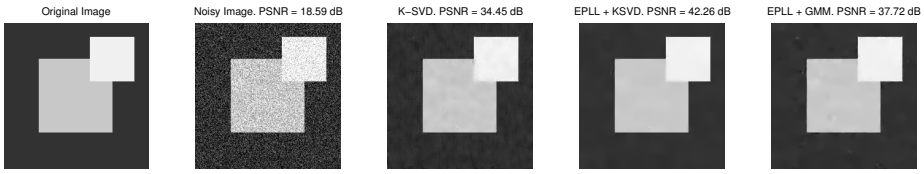


**Fig. 1.** Denoising of a synthetic image ($\sigma = 30$). A similar demonstration was presented in [16], showing the benefits of the EPLL framework under a GMM approach. Note the texture-like resulting artifacts in the result by K-SVD. This problems is notably reduced by the EPLL with a Sparse Prior, the method we present in this work. We include for comparison the result by [16]. The evolution of the Peak Signal to Noise Ratios are depicted in Fig 4.

noise with an *off-the-shelf* algorithm, we propose an intrinsic alternative by using the information we already have about each patch.

Consider the general problem of estimating the remaining noise after applying K-SVD on the noisy image; i.e., the first iteration of our method. From a global perspective, the estimated image can be expressed as

$$\hat{\mathbf{x}} = \frac{\left(\lambda \mathbf{A}^T + \sum_i \mathbf{P}_i^T \mathbf{D}_{S_i} \mathbf{D}_{S_i}^+ \mathbf{P}_i\right)}{\lambda \mathbf{A}^T \mathbf{A} + \sum_i \mathbf{P}_i^T \mathbf{P}_i} \mathbf{y}, \tag{9}$$

where $S_i$ denotes the support of the sparse vector $\hat{\alpha}_i$ chosen in the OMP, and $\mathbf{D}_{S_i}$ is the set of the corresponding atoms in the dictionary. Leaving aside the selection of the support of each sparse vector, we can represent this operation by a linear operator as

$$\hat{\mathbf{x}} = \mathbf{L}(\mathbf{x} + \mathbf{n}). \tag{10}$$

Assuming for a moment that $\mathbf{x} \approx \mathbf{L}\mathbf{x}$, we could express the remaining noise as $\mathbf{n_r} = \mathbf{L}\mathbf{n}$, from which we could obtain the full covariance matrix as $Cov(\mathbf{n_r}) = \sigma^2 \mathbf{L}\mathbf{L}^T$. Then, we could either take into consideration the full covariance matrix, or make the simplifying assumption of white noise by considering just the diagonal of $Cov(\mathbf{n_r})$. Though appealing, this approach does not work in practice because $||\hat{\mathbf{x}} - \mathbf{L}\mathbf{x}||_2$ is considerably large, and thus the estimate of the remaining noise is considerably low. Also, note that $\mathbf{L}$ is a band matrix of size $N^2 \times N^2$, where $N$ is the number of pixels, and so the estimation of its covariance matrix is computationally intractable for practical purposes.

We thus turn to a similar but local alternative that will enable a practical solution. Each patch consists of the true underlying vector $\mathbf{z}_{0i}$ and a noise component $\mathbf{v}_i$, $\mathbf{z}_i = \mathbf{z}_{0i} + \mathbf{v}_i$. Given the chosen support $S_i$, $\hat{\mathbf{z}}_i$ is obtained as a projection onto the span of the selected atoms:

$$\hat{\mathbf{z}}_i = \mathbf{D}_{S_i} \mathbf{D}_{S_i}^+ \mathbf{z}_i = \mathbf{D}_{S_i} \mathbf{D}_{S_i}^+ \left(\mathbf{z}_{0i} + \mathbf{v}_i\right). \tag{11}$$

Assuming now that $\mathbf{z}_{0i} \approx \mathbf{D}_{S_i} \mathbf{D}_{S_i}^+ \mathbf{z}_{0i}$ (if the correct support of the signal was chosen by the OMP), the contribution of the noise to the patch estimate would

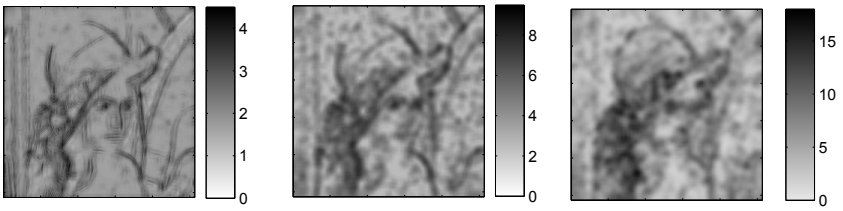


**Fig. 2.** Left: plot of the diagonal of the covariance matrix $Cov(\mathbf{n_r})$ after the first iteration of denoising the image Lena ($\sigma = 20$). Center: the corresponding plot of the estimated $\mathbf{R}^k$ in Eq. (13), and right: the corresponding average of the standard deviation per patch of the true error image.

be given by $\hat{\mathbf{v}}_i^r = \mathbf{D}_{S_i}\mathbf{D}_{S_i}^+\mathbf{v}_i$. This is an analogue assumption to that made for Eq. (10), but now for each patch instead of the global image. This way, considering the covariance matrix of the remaining noise $Cov(\hat{\mathbf{v}}_i^r)$, the mean squared error estimate at the $i^{th}$ patch and iteration $k$ will be given by $\frac{1}{n}tr\{Cov(\hat{\mathbf{v}}_i^r)\}$, leading to

$$\left(\hat{\sigma}_i^k\right)^2 = |S_i|\frac{\nu_k^2}{n}. \tag{12}$$

Therefore, the estimate of the remaining noise in each patch is simply proportional to the number of atoms used for that patch. Note that the remaining noise is no longer white after the back projection step, but we make this assumption in order to simplify further derivations.

Generalizing this patch analysis to the entire image, we can estimate the average remaining noise in the image $\mathbf{x}$ by performing an estimate in the spirit of Eq. (5), tilling back and averaging the local estimates as

$$\mathbf{R}^k = \frac{\lambda\nu_k^2\mathbf{I} + \sum_i \mathbf{P}_i^T\mathbf{1}\ \left(\hat{\sigma}_i^k\right)^2}{\lambda\nu_k^2\mathbf{I} + \sum_i \mathbf{P}_i^T\mathbf{P}_i} = \Phi\left((\hat{\sigma}_i^k)^2\right), \tag{13}$$

where the operator $\Phi(\cdot)$ relocates the local estimates $\hat{\sigma}_i^k$ with the corresponding weighting. This way, $\mathbf{R}^k$ stands for an estimation of the energy of the remaining noise pixel-wise, equivalent -but not equal- to the diagonal of $Cov(\mathbf{n_r})$. An example is shown in Fig. 2 for the popular image Lena. We see that $\mathbf{R}^k$ provides a fair estimate of the information in the diagonal of the full covariance matrix of the remaining noise $Cov(\mathbf{n_r})$, and that it is closer to the average of the standard deviation per patch of the true error image. The reader should also note that computing $\mathbf{R}^k$ is considerably cheaper than the computation of the operator in (10), since we only compute the local covariance matrices and their weighted average, and the matrix in the denominator of Eq. (13) is a diagonal one. Therefore we use $\mathbf{R}^k$ to derive the threshold for the next iteration.

From this point two possibilities arise: use $\mathbf{R}^k$ to evaluate a local patch-based noise energy, eventually denoising each patch with a different threshold,

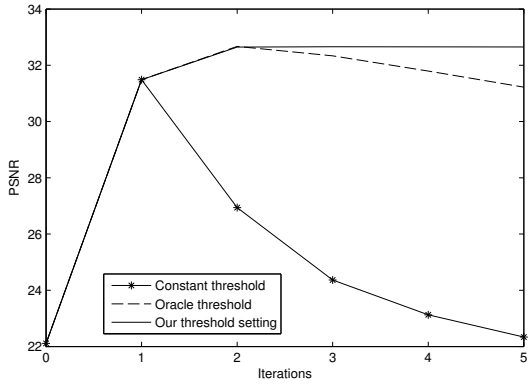


**Fig. 3.** PSNR evolution of the EPLL scheme with a sparse-representation prior for denoising the image Lena ($\sigma = 20$) and three different threshold settings: a) using a constant threshold for all the iterations (equal to the initial noise energy $\sigma^2$); b) using an oracle threshold by setting it to be the variance of the real error image (having access to the original image); and c) our threshold setting method.

or finding a new global and common threshold for all the patches. Adopting the first alternative was found not to yield significant improvements in our results. Thus, in the following we adopt the second global alternative.

The reader should bare in mind that the thresholds should tend to zero as we iterate, corresponding to $\beta \to \infty$. Certainly, this implies that our thresholds will not reflect the *real* remaning noise. As an example, in Fig. 3 we present the evolution of the PSNR by the proposed method for the image Lena for different thresholds. We see that if the threshold is not changed with the iterations, the PSNR of the resulting image $\mathbf{x}$ decreases after the first iteration. On the other hand, if we set the threshold to be the variance of the real remaining noise (by having access to an oracle and the original image), the PSNR initially increases but eventually decreases since the threshold do not tend to zero. We include for comparison the results of our threshold-setting method.

This way, in what follows we propose a method that provides decreasing thresholds and which has been proven to be robust. In the subsequent iterations, we set the threshold $\nu_k^2$ to be the mode of the values in $\mathbf{R}^k$ . Furthermore, we have found that the multiplication by a constant factor $\delta$ improves the performance in our method. To this end, assuming independence between the remaining noise and the patch estimate, and considering the residual per patch $\mathbf{r}_i = \mathbf{z}_i - \hat{\mathbf{z}}_i$, we have that $\tilde{\sigma}_i^2 = \sigma^2 - Var(\mathbf{r}_i)^2$. With these estimates we can perform an analogue of Eq. (13) and obtain its mode, $\tilde{\nu}^2$. We then define the factor $\delta = \tilde{\nu}^2/\nu_k^2$, and set the thresholds for the next iteration to be $\nu_k^2 = \delta \cdot mode(\mathbf{R}^k)$. A full description of our algorithm is depicted in Algorightm 1.

In the following iterations the assumption about the independence between the remaining noise and the patch estimate will be very week, and so $\tilde{\sigma}_i^2$ will not

[2] The variance is calculated as $Var(\mathbf{r}) = \frac{1}{n-1}\sum_j (\mathbf{r}_j - \overline{\mathbf{r}})^2$, where $\overline{\mathbf{r}}$ is the mean of $\mathbf{r}$.

**Algorithm 1:** EPLL with a Sparse Prior, given the noisy image $\mathbf{y}$ with a noise standard deviation of $\sigma$ and an initial dictionary $\mathbf{D}_0$.

**Initialization:** $\mathbf{x} = \mathbf{y}$. $\mathbf{D} = \mathbf{D}_0$, $\delta = 1$, $k = 1$, $\nu_k^2 = \sigma^2$.

**for** $OuterIter = 1 : 3 - 4$ **do**

- $\{\mathbf{D}^{k+1}, \mathbf{x}^{k+1}\} = \underset{\alpha_i, \mathbf{D}, \mathbf{x}}{argmin}\ \lambda||\mathbf{x}^k - \mathbf{y}||_2^2 + \sum_i ||\mathbf{D}^k\alpha_i - \mathbf{P}_i\mathbf{x}^k||_2^2 + \mu_i||\alpha_i||_0$, by K-SVD with error threshold $\nu_k^2$;
- get local estimates $\left(\hat{\sigma}_i^k\right)^2 = |S_i|\frac{\nu_k^2}{n}\ ,\ \forall i$;
- get global estimate $\mathbf{R}^k = \Phi\left((\hat{\sigma}_i^k)^2\right)$ with Eq. (13);
- **if** $k = 1$ **then**
  - $\nu_{k+1}^2 = mode(\mathbf{R}^k)$;
  - $\tilde{\sigma}_i^2 = \sigma^2 - Var(\mathbf{r}_i),\ \forall i$;
  - $\tilde{\nu}^2 = mode\left(\Phi(\tilde{\sigma}_i^2)\right)$;
  - $\delta = \tilde{\nu}^2/\nu_{k+1}^2$;
- $\nu_{k+1}^2 = \delta \cdot mode(\mathbf{R}^k)$;
- $k = k + 1$;

**Output:** $\mathbf{x}, \mathbf{D}$.

be accurate. Thus, $\delta$ is determined after the first iteration only and kept fixed for the subsequent steps, while the estimate $\nu_k^2$ provides decreasing estimates every time. An example of the obtained $\nu_k^2$'s can be seen in Fig. 4.

## 4 Results

To gain some insight into the performance of our method and as a motivating example, in Fig. 1 we present the denoising results on a synthetic image obtained by the regular K-SVD algorithm, and the one achieved by applying the EPLL approach with the sparse-enforcing prior. A similar demonstration was presented in [16], and we include the results of this method as well. The K-SVD denoised image presents texture artifacts common to patch-based algorithms, while in the image denoised with our method the final patches are far more likely under the prior that we try to learn from the image itself.

Fig. 4 depicts the evolution of the PSNR of the denoised image in each iteration for this experiment. Note that given a fixed dictionary, solving the MAP estimate for each patch with a sparse prior implies applying OMP on each of them. This corresponds to the EPLL+OMP curve. On the other hand, we could minimize Eq. (8) w.r.t $\mathbf{D}$ as well by applying a K-SVD step, updating the dictionary as well as the sparse vectors; this is the curve depicted as EPLL+K-SVD. The constant dotted line corresponds to the original K-SVD algorithm. Note that the result after the first iteration in our method is worse than the

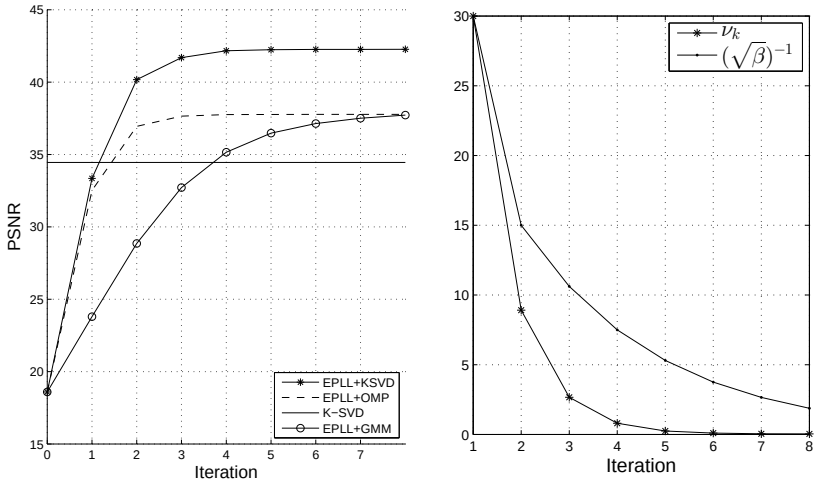


**Fig. 4.** Left: PSNR evolution by EPLL with a sparsity inducing prior on the synthetic image in Fig. 1, compared to the original K-SVD algorithm [6] and the EPLL-GMM of [16]. Right: sequence of thresholds $\nu_k$ determined by the proposed method and the equivalent $1/\sqrt{\beta}$ by the method of [16].

one obtained by K-SVD where $c = 1.15$. Choosing $c = 1$ in our case, however, enables further improvement as we proceed maximizing the Expected Patch Log Likelihood. Notice also that our method converges in considerable fewer iterations than the method of [16]. The right side of Fig. 4 shows the evolution of the thresholds $\nu_k$ used in the successive iterations, as well as the values $1/\sqrt{\beta}$ used by EPLL-GMM.

The improvement obtained by training the dictionary in each iteration of our method is both important and intuitive. It is known that applying K-SVD on a noisy image achieves good denoising results but yields somewhat noisy atoms [6]. By training the dictionary $\mathbf{D}$ in the progressively cleaner estimates $\mathbf{x}$ we obtain cleaner and more well defined atoms, which are later used to perform further denoising. In the top row of Fig. 5 we present 8 atoms trained on a noisy version of the image Lena after the first iteration, while the lower row shows the same atoms after 4 iterations.

### 4.1 Inpaiting

We next present results on image inpainting. In this particular application of image restoration, the signal is the outcome of a linear operator that deletes a number of pixels from the original image $\mathbf{x}$, plus the measurement noise. By considering a sparse prior on the original signal, we can formulate an equivalent problem to that of Eq. (6), where $\mathbf{A}$ is the missing-pixels mask. The corresponding cost function can be minimized in a block coordinate manner, coding for the unknown sparse representation and updating the dictionary. In this case, however, the threshold in the OMP has to consider only the energy of existing pixel in each patch [9]. This again represents the first iteration of the Half Splitting

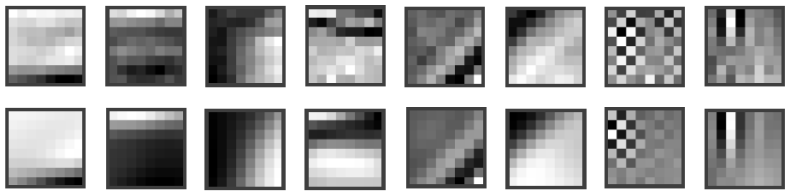

**Fig. 5.** Atoms from a dictionary trained on a noisy version of the image Lena. The top row corresponds to the atoms after the first iteration of our method (essentially, after applying K-SVD), while the lower row corresponds to the same atoms after 4 iterations of the EPLL with a sparsity enforcing prior.

strategy proposed in [16], and we may perform the next iterations by estimating the remaining noise as explained above. Furthermore, after the first iteration our estimate includes values of the missing pixel. We can then make use of the previous denoising strategy to tackle the next iteration, by having knowledge of the supports used to inpaint each patch, as it was previously explained.

Table 1 shows the results on inpainting the popular images *peppers* and *Lena* with 25%, 50% and 75% missing pixels, with additive white Gaussian noise ($\sigma = 20$). As it can be seen, the EPLL scheme leads to a slight improvement in the K-SVD inpainting results, with increased effect for higher missing pixels rates. The same concept could be applied to more sophisticated algorithms that use a sparsity-based prior, such as the state-of-the-art method of [11].

### 4.2 Denoising

We conclude this paper by presenting results on denoising of 12 images from the Kodak database, for different noise levels. We compare here the performance of the K-SVD denoising algorithm in [6] and our approach of the EPLL framework with a sparse prior (EPLL-K-SVD, where the dictionary is also updated in each iteration). In all cases we performed 4 iterations of this method, as this was found to be a convenient compromise between runtime and performance. For both K-SVD methods, an initial dictionary with 1024 atoms was trained on overlapping $8 \times 8$ patches from 9 training images using K-SVD. We include for completion the results achieved by the EPLL with a Gaussian Mixture Model (GMM) as the image prior from [16].

| Missing Pixels | 25% | | 50% | | 75% | |
|---|---|---|---|---|---|---|
| K-SVD | 29.67 | 28.81 | 27.92 | 27.27 | 23.64 | 23.86 |
| EPLL+K-SVD | **29.71** | **28.85** | **28.18** | **27.39** | **23.81** | **24.07** |

**Table 1.** Inpaiting results in terms of Peak Signal to Noise Ratio (PSNR) for 25%, 50% and 75% missing pixels for the images *peppers* (left subcolumns) and *Lena* (right subcolumns), with additive white Gaussian noise ($\sigma = 20$).

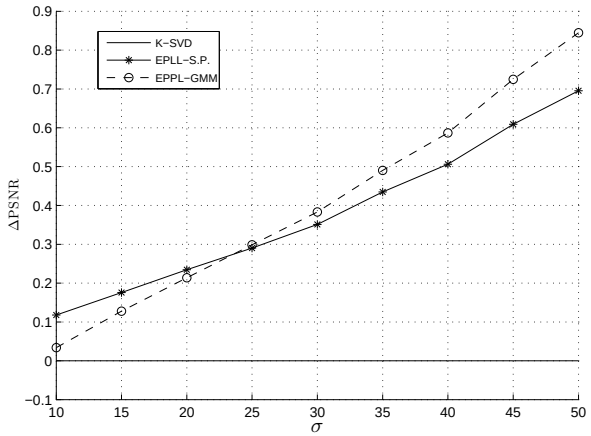


**Fig. 6.** Denoising results averaged over 12 images from the Kodak Dataset with respect to K-SVD [6] by EPLL with GMM [16] and the method presented here: EPLL with Sparse Prior, in terms of the Peak Signal to Noise Ratio (PSNR).

In Fig. 6 we present the relative increase in PSNR, averaged over all 12 images. The EPLL with a Sparse enforcing Prior shows a clear improvement over the regular K-SVD. Furthermore, the complete implementation of the denoising algorithm closes the gap between the original K-SVD and EPLL-GMM, having comparable performance: our method achieves the best results for lower noise energy while EPLL with GMM is better for higher noise levels. In Fig.7 and Fig.8 we present two examples of denoised images by the three methods. Note how artifacts are notably reduced in the resulting images processed by our method.

## 5 Conclusion

Maximizing the Expected Patch Log Likelihood with a sparse inducing prior leads naturally to a formulation of which the K-SVD algorithm represents the first iteration. In its original form, this method performed only one update of the image due to technical difficulties in assessing the remaining noise level. In this paper we have shown how to go beyond this first iteration, intrinsically determining the coding threshold in each step. This work completes the one in [6], providing the full path to the numerical minimization of the original cost function and exploiting all the potential of the sparse inducing prior.

Our algorithm shows a clear improvement over K-SVD in all the experiments. In denoising in particular, EPLL with a sparse prior achieved comparable performance to the state of the art method of EPLL with a GMM prior. Interestingly, both priors yield comparable results when applied within the EPLL framework. An approach like the one presented here could be employed in other applications where a MAP estimator for a sparse prior is used for image restoration.

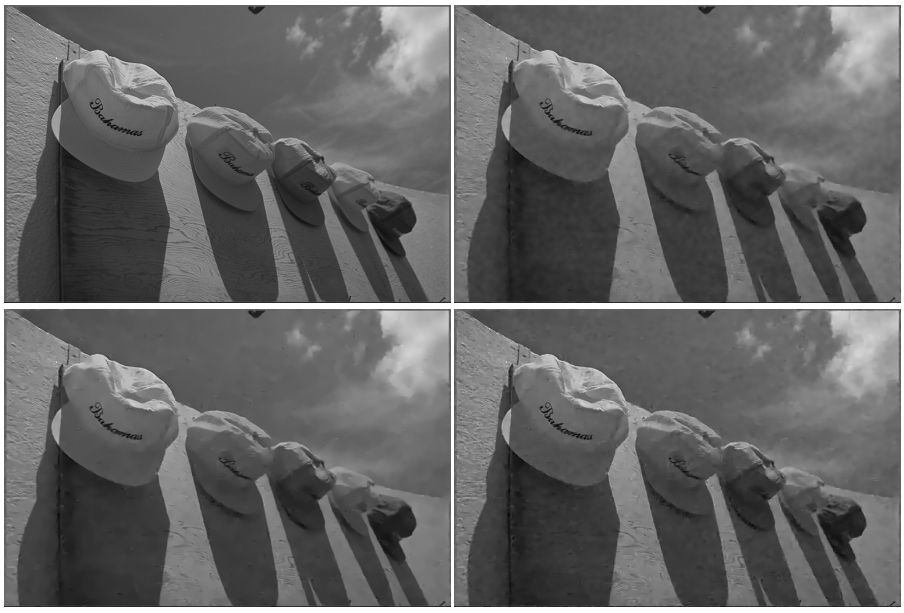

**Fig. 7.** Denoising results of an image from the Kodak Database corrupted with a noise standard deviation of $\sigma = 25$. Top left: original image. Top right: K-SVD (PSNR = 32.14 dB). Bottom left: EPLL with Sparse Prior (PSNR = 32.42 dB). Bottom Right: EPLL with GMM (PSNR = 32.25 dB).

**Fig. 8.** Denoising results of an image from the Kodak Database, initially corrupted with additive white Gaussian nose ($\sigma = 25$). From left to right: Original Image, K-SVD (PSNR = 31.42 dB), EPLL with Sparse Prior (PSNR = 31.83 dB), and EPLL with GMM (PSNR = 31.85 dB).

**Acknowledgment.** This research was supported by the European Research Council under European Union's Seventh Framework Program, ERC Grant agreement no. 320649, by the Intel Collaborative Research Institute for Computational Intelligence and by the J.D. Erteschik Fund for Practical Research.

## References

1. Aharon, M., Elad, M., Bruckstein, A.M.: K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. IEEE Transactions on Signal Process. 54(11), 4311–4322 (2006)
2. Bruckstein, A.M., Donoho, D.L., Elad, M.: From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images. SIAM Review. 51(1), 34–81 (Feb 2009)
3. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. Conference on Computer Vision and Pattern Recognition, CVPR IEEE. pp. 60–65 (2005)
4. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising with block-matching and 3D filtering. Proc. SPIE-IS&T Electron. Imaging 6064, 1–12 (2006)
5. Donoho, D., Elad, M., Temlyakov, V.: Stable recovery of sparse overcomplete representations in the presence of noise. Information Theory, IEEE Transactions on 52(1), 6–18 (Jan 2006)
6. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. Image Process. 15(12), 3736–3745 (Dec 2006)
7. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online Dictionary Learning for Sparse Coding. In: 26th International Conference on Machine Learning. Montreal, Canada (2009)
8. Mairal, J., Bach, F., Sapiro, G.: Non-local Sparse Models for Image Restoration. 12th IEEE International Conference on Computer Vision. 2, 2272–2279 (2009)
9. Mairal, J., Elad, M., Sapiro, G., Member, S.: Sparse Representation for Color Image Restoration. IEEE Transactions of Image Processing 17(1), 53–69 (2008)
10. Portilla, J., Strela, V., Wainwright, M.J., Simoncelli, E.P.: Image denoising using scale mixtures of Gaussians in the wavelet domain. IEEE Transactions on Image Processing. 12(11), 1338–51 (Jan 2003)
11. Romano, Y., Protter, M., Elad, M.: Single Image Interpolation via Adaptive Non-Local Sparsity-Based Modeling. to appear in IEEE Transactions on Image Processing.
12. Roth, S., Black, M.J.: Fields of Experts. International Journal of Computer Vision. 82(2), 205–229 (Jan 2009)
13. Rubinstein, R., Zibulevsky, M., Elad, M.: Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit. Tech. - Comput. Sci. Dep. - Technical Report. pp. 1–15 (2008)
14. Tropp, J.: Greed is Good: Algorithmic Results for Sparse Approximation. IEEE Transactions on Information Theory 50(10), 2231–2242 (Oct 2004)
15. Weiss, Y., Freeman, W.T.: What makes a good model of natural images? In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (Jun 2007)
16. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. 2011 International Conference on Computer Vision, ICCV. pp. 479–486 (Nov 2011)