Image Denoising

Homework n°4 - Experimental report

1 EPLL: An Image Denoising Method Using a Gaussian Mixture Model Learned on a Large Set of Patches

The Expected Patch Log-Likelihood (EPLL) denoising method [1] uses a Gaussian prior, similarly to Non-Local Bayes algorithm.
However, a first difference is that EPLL does not use a simple Gaussian model, but a Gaussian *mixture*.
Moreover, the prior used by EPLL is based not on the image itself, but on a database: it is a "global" algorithm.

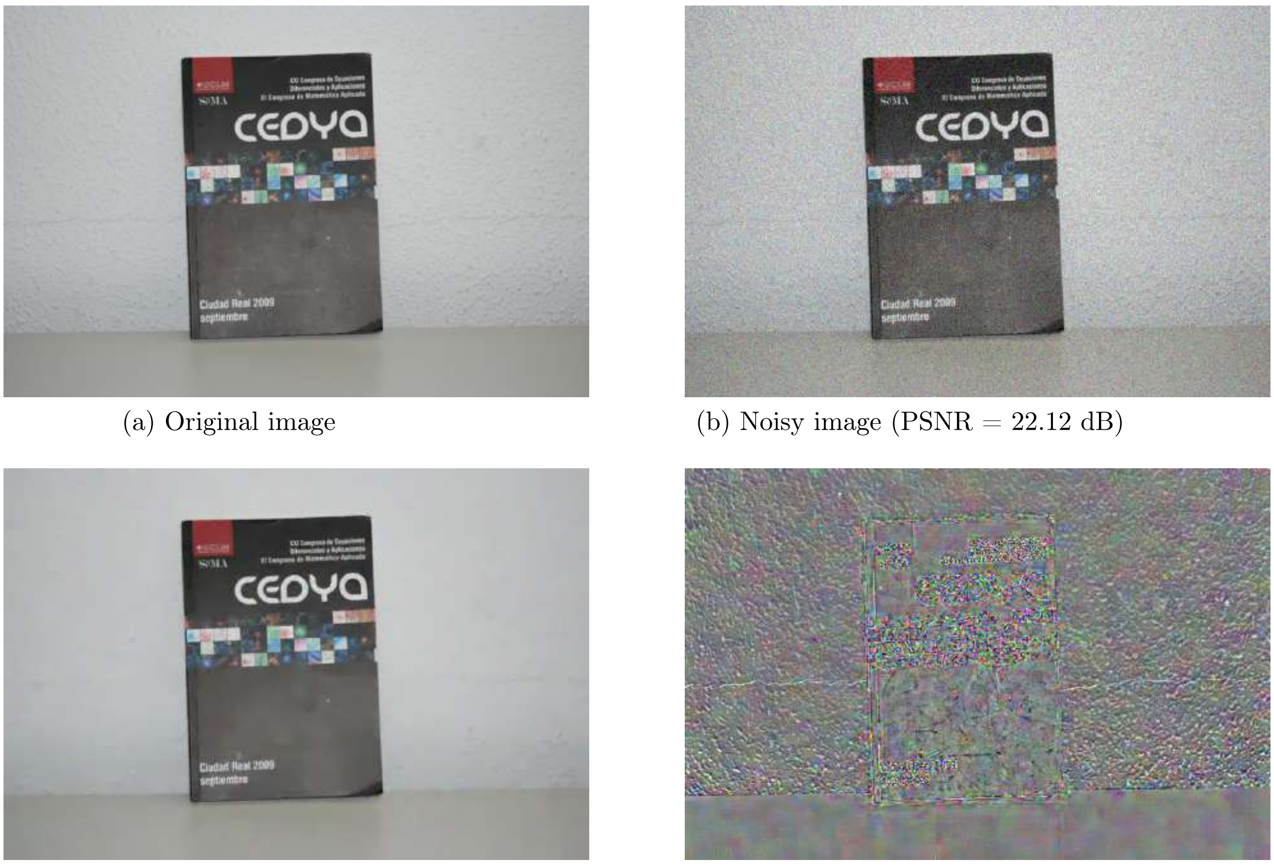
First, we will analyze the impact of the different parameters in the EPLL algorithm. Then, we will compare this algorithm to the ones we have studied previously.

1.1 First results

First, we use $\sigma = 20$, 4 steps, 100% of the covariance matrices, and 2 scales.
The results for two different images are shown in Figure 1 and Figure 2.

The results are overall good. However, when zooming in, we see the edges are not very well denoised, see Figure 3.
Moreover, in the book image, we see that the asperities on the wall have disappeared (as can also be seen in the difference images). And in the building image, we see in the denoised image that color red remains in the tree, whereas no red was present initially: the denoising algorithm was not able to differentiate the noise, and the initial texture.
There tends to be a lot of blur, for instance in the textured areas.

So this method is in general good, yielding high PSNR. However, in presence of edges or small details (such as the asperities of the wall), it does not perform well, and it tends to be better on homogeneous areas.



(a) Original image

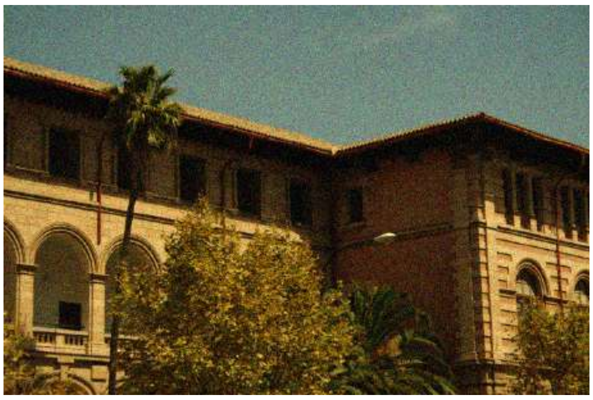
(b) Noisy image (PSNR = 22.12 dB)

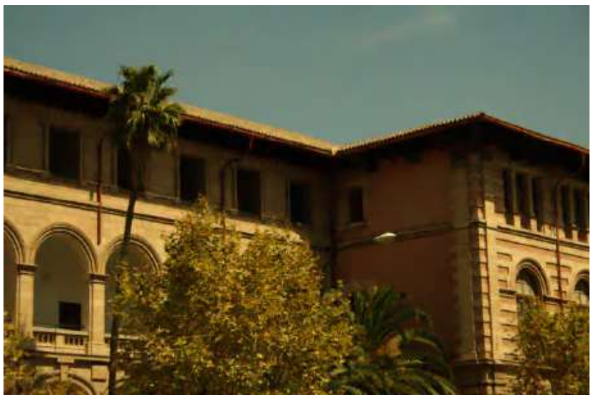
(c) Denoised image (PSNR = 35.53 dB)

(d) Difference

Figure 1: EPLL on a first image

(a) Original image


(b) Noisy image (PSNR = 22.10 dB)


(c) Denoised image (PSNR = 30.27 dB)

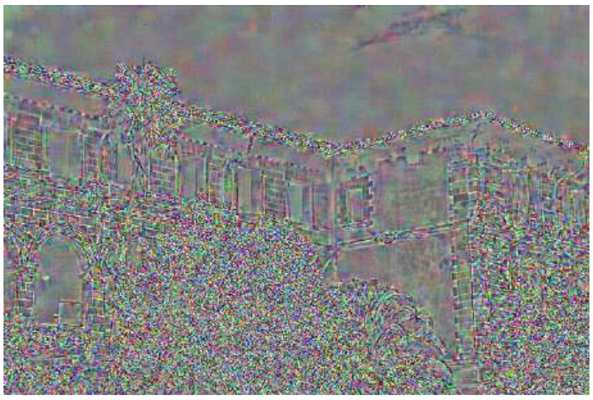
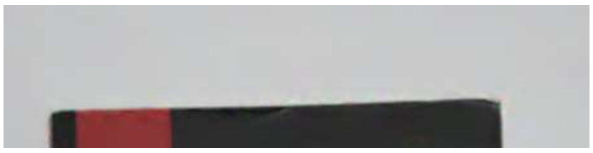

(d) Difference

Figure 2: EPLL on a second image


(a) Edge for the first image

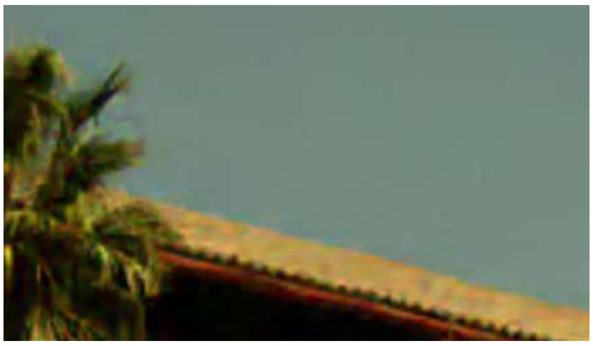

(b) Edge for the second image

Figure 3: Edges for both images

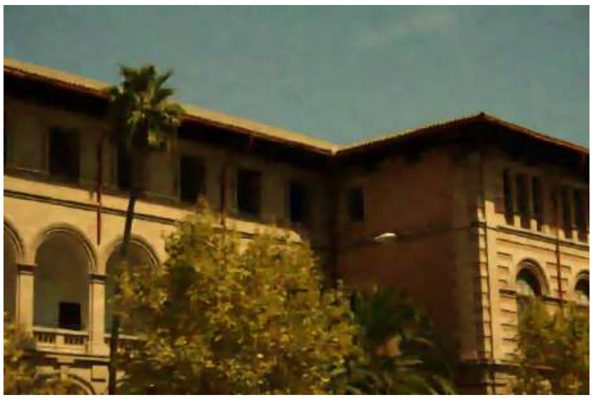
1.2 Impact of the parameters

1.2.1 Step

Figure 4 presents some results for different steps, on the same image as Figure 2. The other parameters are set to 1 scale, 50% of the covariance matrices. Images can be seen in Figure 5.

σ/step	4	6	8
20	29.12 dB	28.91 dB	28.47 dB
50	25.63 dB	25.43 dB	24.36 dB
100	23.25 dB	23.19 dB	21.33 dB
Time	40 s	23 s	16 s

Figure 4: PSNR for different steps and noises


(a) Step 4

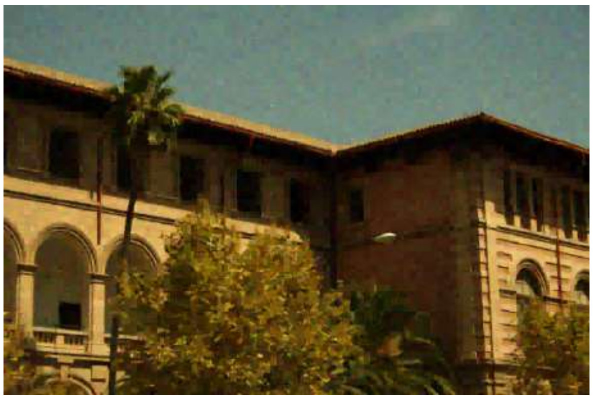

(b) Step 8

Figure 5: Using two different steps

We see that, all other parameters being fixed, increasing the step leads to a smaller PSNR, as well as a shorter execution time.
This is explained by the fact that increasing the step means using fewer patches when denoising a pixel, hence a smaller execution time, but also a larger error (as can be seen in the above images).
So it is preferable to use a small step, even though it means more computations.

1.2.2 Maximum rank for the covariance matrices

Here, we set the scale to 1, and the step to 6, and use different maximum ranks for the covariance matrices in Figure 6.

σ/max. rank	50%	75%	100%
20	28.95 dB	30.02 dB	30.13 dB
50	25.44 dB	25.81 dB	25.93 dB
100	23.14 dB	23.21 dB	23.22 dB
Time	23 s	28 s	34 s

Figure 6: PSNR for different ranks and noises

We see that a higher rank leads to a better PSNR, while increasing the computation time.

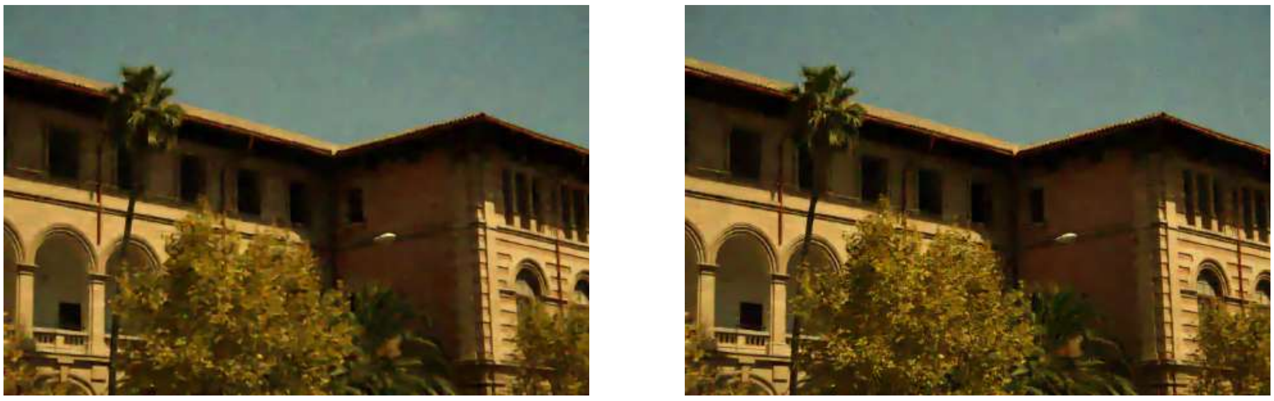

(a) Rank 50% (b) Rank 100%

Figure 7: Using two different ranks

We also see that with a higher rank, the details are much better (see the tree for instance): high frequencies are better handled.
This makes sense, because using a higher rank means using more information (that is, more covariance matrices), hence better results (but also more computations).

1.2.3 Number of scales

Here, we set the step to 6, the maximum rank to 50%, and use different scales in Figure 8.

σ/scale	1	2	3
20	29.11 dB	27.85 dB	26.41 dB
50	25.61 dB	25.34 dB	24.60 dB
100	23.31 dB	23.30 dB	23.04 dB

Figure 8: PSNR for different scales and noises

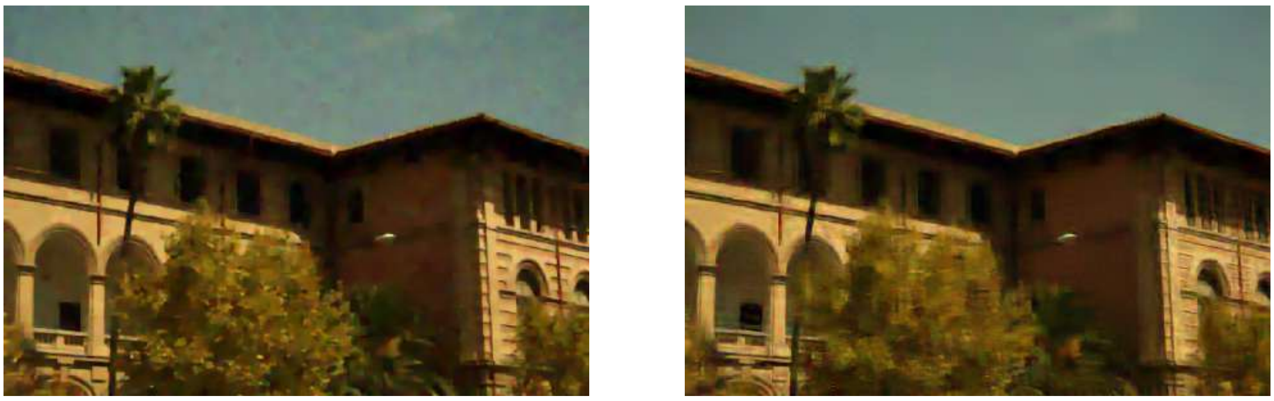

(a) Scale 1 (b) Scale 3

Figure 9: Using different scales

In terms of PNSR, the multi-scale tends to be worse than the single-scale, the difference being more visible for small noises.
However, for $\sigma = 50$, we see in Figure 9 that using more scales leads to a much better result in homogeneous areas (the sky).

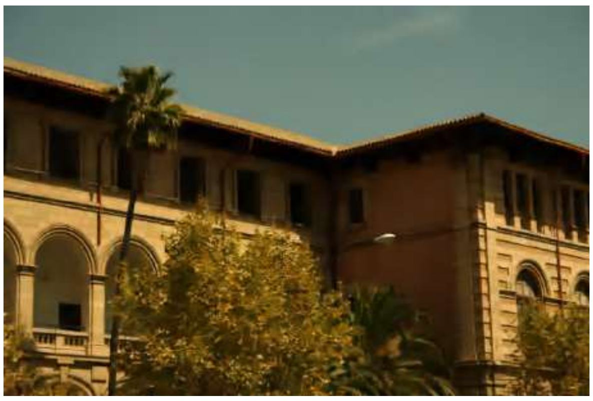
This is coherent with what we had seen with DCT and MS-DCT.

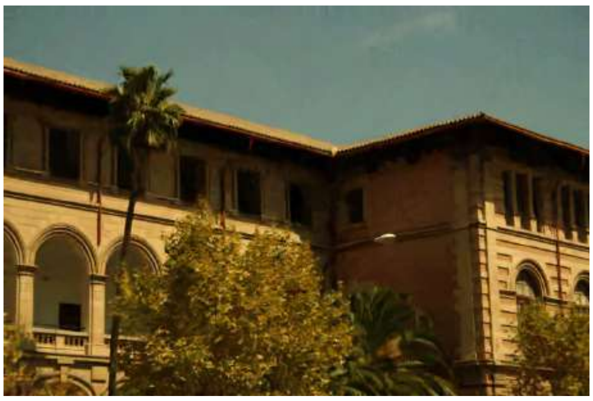
1.2.4 RGB to OPP transform

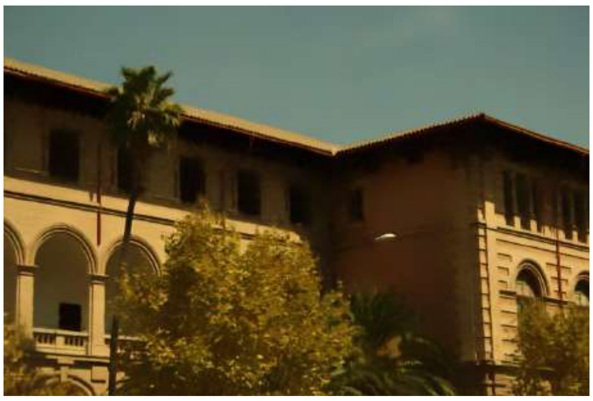
Using OPP transform did not yield significant differences for the images I used (in terms of PSNR and visual results).

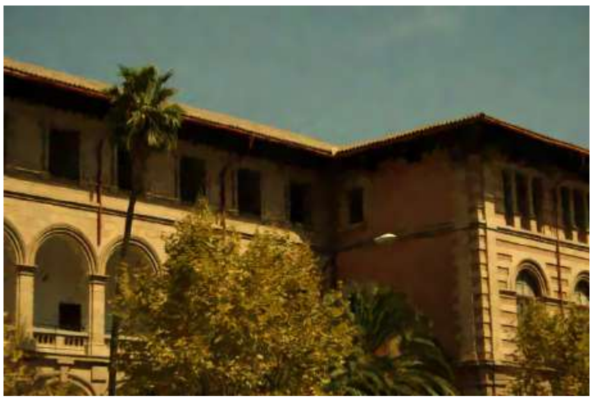
1.3 Comparison with other methods

In Figure 10, we compare MS-DCT, BM3D, NL-Means, NL-Bayes and EPLL.


(a) MS-DCT (29.23 dB)


(b) BM3D (29.85 dB)


(c) NL-Means (28.75 dB)


(d) NL-Bayes (30.07 dB)

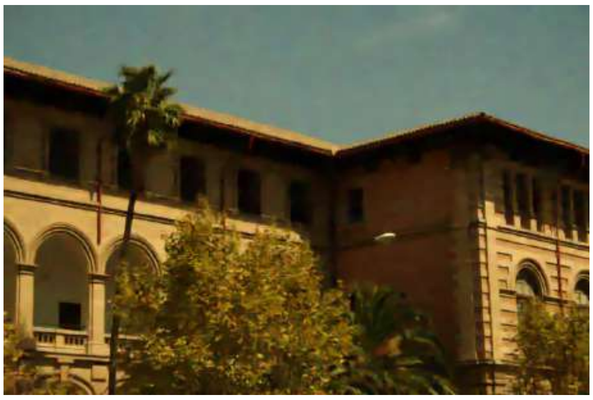
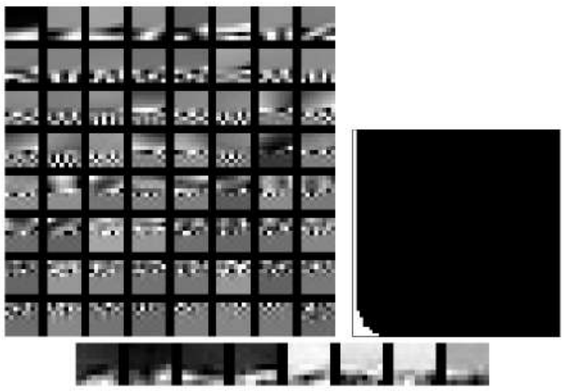

(e) EPLL (28.42 dB)

Figure 10: Comparison of different methods

We notice that the best PSNR is obtained with NL-Bayes, while the worst one is obtained with EPLL.
We see that the image obtained with MS-DCT is blurry, the one with EPLL has issues near the edges, the cloud has disappeared in NL-Means, etc. So it is hard to really compare all these methods, since they each have their strengths and weaknesses.
If I were to judge visually, I would say that the image obtained with BM3D is the best one.

2 Zoran-Weiss Gaussians Mixture Model

The PDF document describes a Gaussian mixture with 400 components, trained from some database.
We see that the weights range from 0.00003 to 0.00363.


(a) 40-th Gaussian

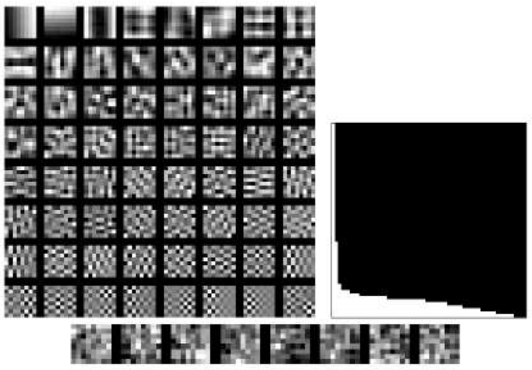

(b) 141-st Gaussian

Figure 11: Two components

We see in Figure 11 that some components focus on edges (and are sparse, see the 40-th Gaussian), while others focus on textures (and are not sparse, see 141-st Gaussian). The first eigenvectors (associated with highest eigenvalues) show what each component mainly focuses on.
In general, looking at all the Gaussians in the document, we see there is quite some variety: the different Gaussians focus on different elements of the images.

References

[1] Samuel Hurault, Thibaud Ehret, and Pablo Arias. EPLL: An Image Denoising Method Using a Gaussian Mixture Model Learned on a Large Set of Patches. *Image Processing On Line*, 8:465–489, 2018. https://doi.org/10.5201/ipol.2018.242.