Master MVA
Image Denoising

Experimental Report 4

Realized By: Eya Ghamgui

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

EPLL Method:

The Expected Patch Log-Likelihood (EPLL) method is considered one of the best denoising algorithms. This method differs from all the previously studied algorithms as it is developed as an external denoising method. It uses a patch-based prior learned using a large set of images. In this case, the prior is chosen as a mixture of Gaussian model whose parameters are estimated with the Expectation Maximization (EM) algorithm. Here, the dataset chosen to compute the parameters affects the denoising quality. Thus, the selection of images for the dataset is a challenging step. Furthermore, this method has the same approach as the PLE method, but they differ in the way they learn the parameters. Indeed, PLE learns a GMM for each noisy image.

The first step of the EPLL is the extraction of clean patches from the dataset. We can say that this method is not based on the extraction of similar patches from the same image but from other patches extracted from different images. After obtaining these patches, the GMM parameters are learned by maximizing the likelihood. Once we have obtained the GMM model, we can denoise the image. We first decompose the noisy image into several overlapping patches. Then, we denoise each patch separately. Finally, an aggregation is applied on the overlapping patches to group the values corresponding to the same data point. This aggregation significantly improves the denoising results by taking advantage of all redundancies.

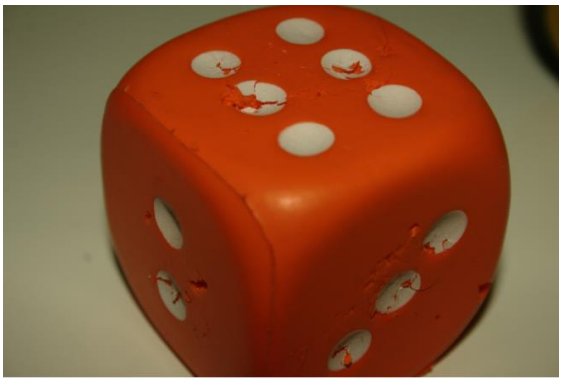
Another version of EPLL is also proposed by applying the prior to different scales of the image simultaneously. This extension is based on the definition of a complex energy. Moreover, it gives more interesting results and higher performance. What is more, the EPLL is applied not only to grayscale images but also to color images. Two approaches are used. The first one consists of denoising each color channel independently as in the grayscale case using the same estimated GMM model learned from grayscale patches. Then, it combines all the denoised channels. The second

approach consists of denoising the color image using a GMM model learned from color patches.

Experiments:

Here, in order to reduce the computational time, we will take the maximum rank considered for the covariance matrices equal to **50%**. Moreover, we will choose the standard deviation of noise equal to **30** for one scale.

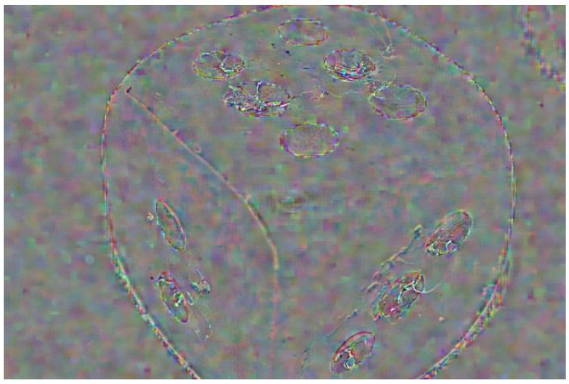
- Example 1:



Input Image

Noisy Image

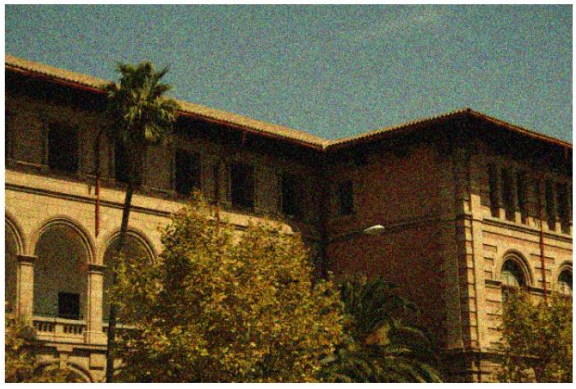
Denoised Image
PSNR = 36.8 dB; RMSE = 3,687



Difference Image

- Example 2:

Input Image



Noisy Image

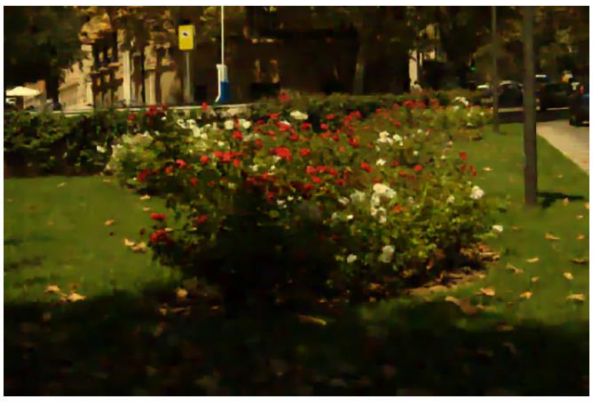
Denoised Image
PSNR = 27.6 dB; RMSE = 10.635

Difference Image

- Example 3:

Input Image

Noisy Image



Denoised Image
PSNR = 26.22 dB; RMSE = 12.455

Difference Image

From the previous results, we can notice that the overall image reconstruction is good. This method succeeded in denoising different types of images and gives high performance. Furthermore, this method performs better in the case of homogeneous images. When the image become more structured, the denoising performance decreases. We can see in the third example, in the case of an image with vegetation, the difference

image shows many colors. Moreover, it contains a lot of structures, which means that the denoising in highly textured areas is not good. This method is not able to differentiate between the noise and small frequencies found in the vegetation area. In addition, the EPLL is not able to reconstruct the edges and the small details. In the first example, we can see that edges and fine details are lost in the denoised image. In addition, artifacts are created in some regions, for example in the sky region in example 2.

Influence of parameters:

In this part, we are going to study the effect of each parameter on the result of the denoising of the "building image" for different values of noise.

- ***Changing the step parameter "s" (rank = 50%):***

σ/ s	4	6	8
20	29.11 dB	28.92 dB	28.50 dB
40	26.47 dB	26.25 dB	25.36 dB
60	24.96 dB	24.79 dB	23.52 dB

Denoised Image (σ = 20 ; s =4)

Difference Image

Denoised Image (σ = 40 ; s =6)

Difference Image

Denoised Image (σ = 60 ; s =8)

Difference Image

From the previous result, we can say that by increasing both the step and the standard deviation of the noise, the performance of this algorithm becomes worse. For a fixed noise value, when we increase the step size, the PSNR decreases. For a fixed step size, when we increase the noise, the denoising quality decreases. Here, the algorithm responds well in the case of low noise values. However, when its value increases, we can notice that we lose more details in the image and it becomes blurred. In addition, the difference image becomes more colorful, and it shows many structures.

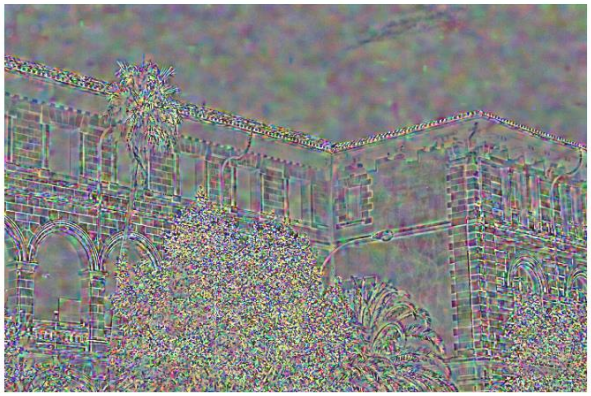
We can say that the denoising of a pixel in an image requires more estimates to decrease the variance of the noise. Also, by increasing the step, we are taking fewer patches when denoising a pixel, which makes the denoising results worse. Moreover, for a high noise value and a high step value, we can say that the image breaks down into several patches, especially in homogeneous regions such as the sky in the last image which is circled in red. This means that the denoising of each pixel requires more patches to achieve better consistency between the denoised pixels.

⇨ In the next steps, we are going to choose the step value "s" the smallest possible.

➢ ***Changing the value of the scale (rank = 50%):***

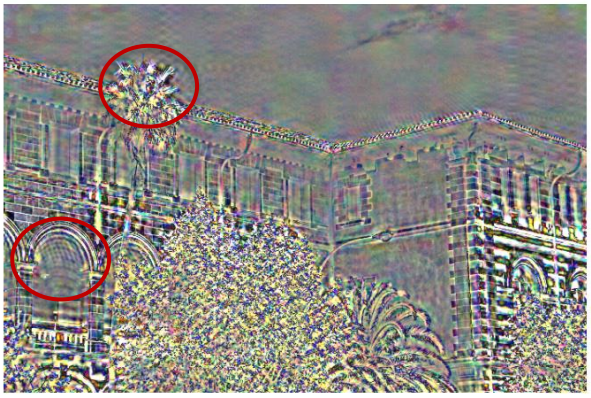
Low value of noise σ = 20:

Denoised Image (scale =1)
PSNR = 29.26 dB



Difference Image

Denoised Image (scale =3)
PSNR = 26.34 dB



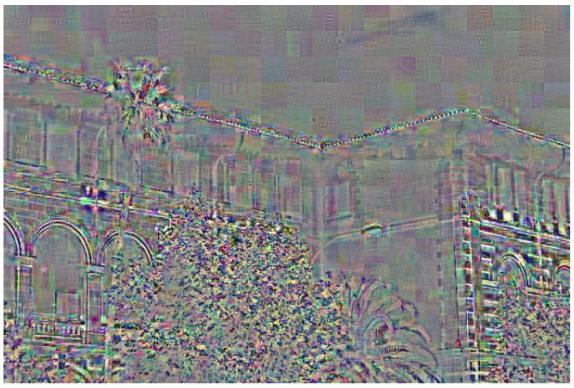
Difference Image

High value of noise σ = 60:

Denoised Image (σ = 60 ; scale =1)
PSNR = 23.52 dB

Difference Image

Denoised Image (σ = 60 ; scale = 3)
PSNR = 23.50 dB



Difference Image

From the previous results, for a fixed value of noise, the performance of the algorithm decreases when we increase the scale. This increase is higher for a small value

of noise. Thus, we can say that using the multi-scale is not helping the denoising of the image.

For a small noise value, we can notice that the one-scale denoising gives better results than multi-scale denoising. In addition, we obtained a high value of PSNR equal to 29.26 dB and the image is well denoised and preserves all the small details. However, for a high scale, we can notice artifacts created in almost all regions of the image, especially in the regions circled in red. This result is due to the fact that each scale is computed using convolutions of the image with a sinc-like function. Due to Gibbs effect, ringing artefacts are created in the resulted image. In a nutshell, for a low noise value, denoising is considered easy and adding multiple scales will make the denoising step more complicated. Thus, the artifacts created by this method will damage the image. In addition, multi-scale denoising tends to destroy fine textures and blur the image.

For a large noise value, one can say that the PSNRs of one scale and of multiple scales are not very far. However, from a visual point of view, the results of the one-scale denoising are worse than the multi-scale ones. This is because using different scales increases the redundancies for a pixel, which is necessary due to the high amount of noise in the image. Therefore, the artifacts in this case are negligible compared to the noise value and their contribution will be considered as an improvement of the image quality and the reconstruction of fine details in the image. Finally, we can say that the multi-scale denoising method significantly improves the treatment of low frequency noise.

- ***Changing the value of the maximum rank considered for the covariance matrices:***

σ/ rank	50 %	75 %	100 %
20	28.91 dB	29.99 dB	30.14 dB
60	24.84 dB	25.06 dB	25.14 dB

Denoised Image (σ = 20; rank = 50%)

Difference Image

Denoised Image (σ = 60; rank = 100%)

Difference Image

From the PSNR values, we notice that the results have improved when we increase the maximum rank considered for the covariance matrices. This result is obvious because we are taking into account more information about the images. Thus, with a high maximum rank, we have much better denoised images. Moreover, we have almost the same performance between one-scale and multi-scale approaches.

We can consider that the changes in PSNR and denoising results improve slightly when the rank value increases. From 50 % to 100 %, it increases by 0.23 dB for σ = 20 and by 0.3 dB for σ = 60. If we compare the performance increase and the computational time, we can say that the results obtained with a rank equal to 50% are acceptable because we save a lot of computational time.

- ***Changing the color space:***

σ / color space	RGB	OPP
20	29.1 dB	29.1 dB
60	25.03 dB	25.05 dB

Denoised Image (σ = 60; RGB)

Difference Image

Denoised Image (σ = 60; OPP)

Difference Image

From these results, we can say that PSNRs of both methods are equal in the case of low noise value. However, the change of color space slightly improves the denoising results for a large noise value.

In fact, these two denoising methods follow the same steps, the only change is that for the second method, we apply a color transformation on the channels (OPP transformation). Indeed, the first channel of the OPP transformation is an average of the three colors, which captures the geometry of the patch. Therefore, the denoised version of this component will have better performance. After combining the channels, we will get better denoising quality.

From a visual perspective, the OPP method gives better results, especially for flat regions or color textures.

Comparison with other methods: NL-Means, NL-Bayes, BM3D

Low value of noise σ = 20:

BM3D; PSNR = 31.81 dB

NL-Means; PSNR = 30.98 dB

NL-Bayes; PSNR = 32.27 dB

EPLL (one scale); PSNR = 30.14 dB
Step = 6; rank = 100%

EPLL (multi-scale); PSNR = 30.13 dB
Step = 6; rank = 100%

High value of noise σ = 60:

BM3D; PSNR = 26.52 dB

NL-Means; PSNR = 25.30 dB

NL-Bayes; PSNR = 27.18 dB

EPLL (one scale); PSNR = 25.16 dB
Step = 6; rank = 100%

EPLL (multi-scale); PSNR = 25.47 dB
Step = 6; rank = 100%

For all noise values, we can notice that the NL-Bayes method is more efficient than all other methods. In fact, the denoised image has a very good quality and it is very close to the original image. Until now, the NL-Bayes is the best denoising method.

For a small noise value, the EPLL method has the worst result for one-scale and multi-scale denoising with PSNRs values equal to 30.14 dB and 30.13 dB respectively. We can notice that there is a small difference between the PSNRs of these two approaches. In addition, multi-scale denoising creates ringing artifacts due to the Gibbs effect, especially in the tree and window regions. Thus, multi-scale denoising does not improve the performance in this case. Moreover, the NL-Means method shows artifacts in the form of lines in the sky region, i.e., the homogeneous area, and it gives better denoising performance than the EPLL method and worse than the NL-Bayes method and the BM3D method.

For a large noise value, the EPLL method performs much better than the NL-Means method in the case of multi-scale denoising with a PSNR value equal to 25.47 dB which is higher than 25.3 dB. Indeed, the NL-Means method gives a very blurred image due to the high variance of noise in the image and the EPLL method gives much better results and preserves the finest details. But it still shows artifacts, especially in highly structured zones like the tree or building area.

Furthermore, the results of EPLL are not as good as those of BM3D and NL-Bayes method. In addition, the one-scale approach gives the worst result compared to the other methods and the denoised image has a lot of artifacts. Besides, the smallest structures especially in the vegetation area are blurred.

Zoran-Weiss Gaussians Mixture Model

This document presents two tables, each describing the Gaussian mixture components with their probabilities. Here, the two scientists found that the effective priors computed on a large database of model images are with only 200 Gaussians. Moreover, this paper presents for each component its distribution and eigenvectors. We can notice that there are sparse and non-sparse components. Also, we can remark that there is a large variety between these Gaussians. Looking at the top eigenvectors, we can say that these components describe different information from one Gaussian to another. This is due to the variation in the elements and structures in the images in the dataset. The following pictures are two examples of two different components. The first image is chosen for Gaussian 55 which is considered a sparse component. The second image is chosen for Gaussian 53 which is considered a non-sparse component.

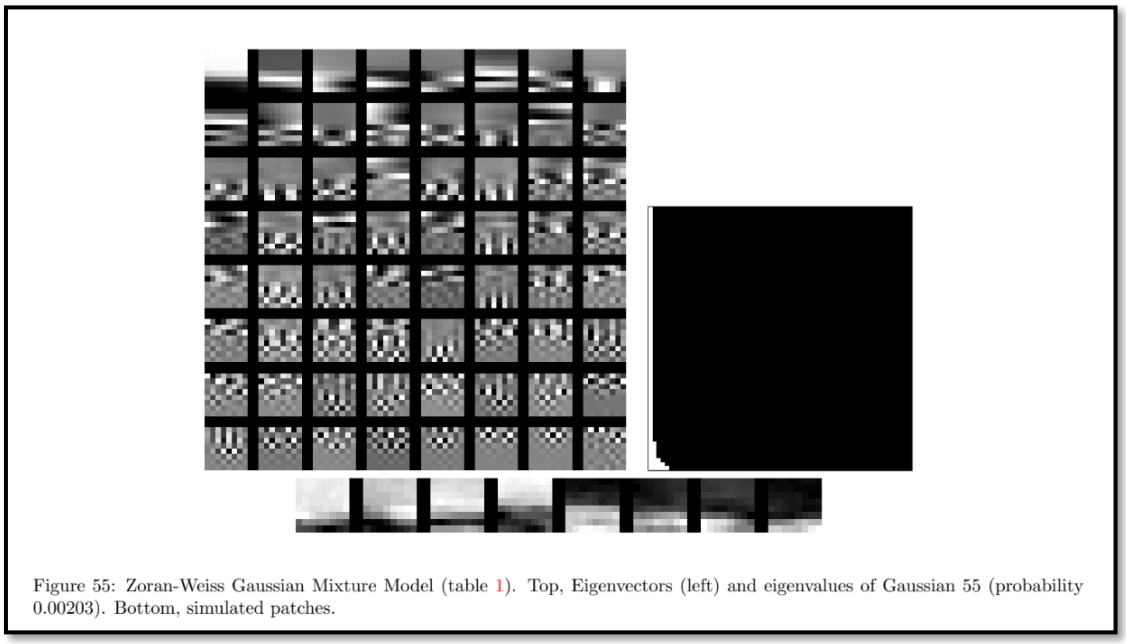


Figure 55: Zoran-Weiss Gaussian Mixture Model (table 1). Top, Eigenvectors (left) and eigenvalues of Gaussian 55 (probability 0.00203). Bottom, simulated patches.

Gaussian 55

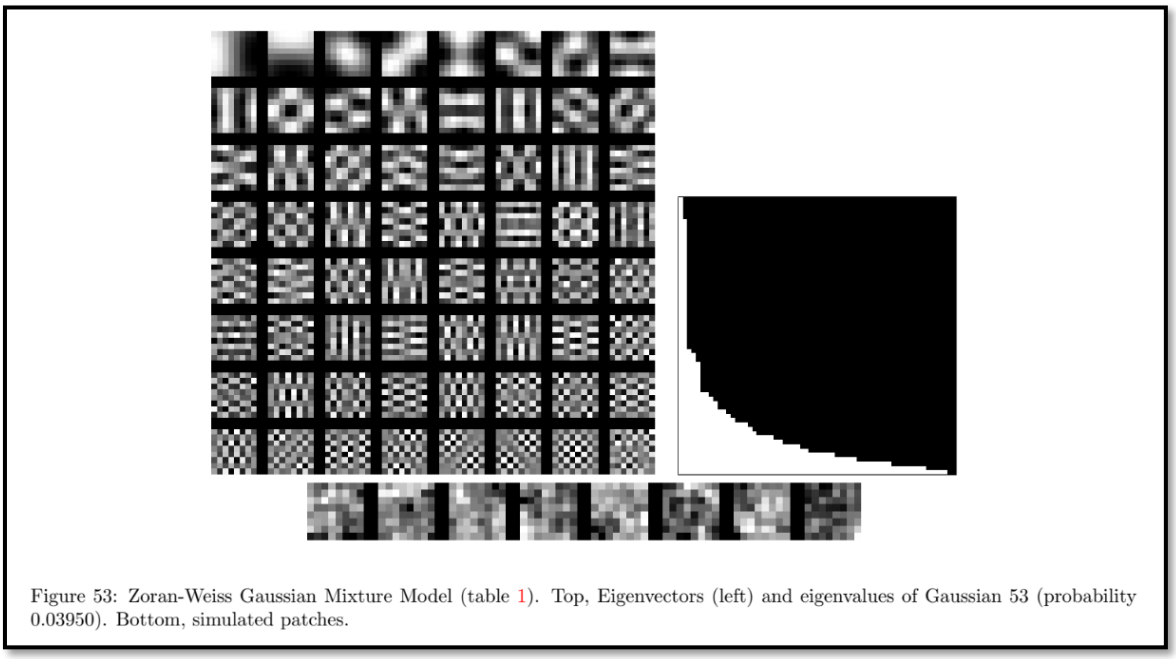


Figure 53: Zoran-Weiss Gaussian Mixture Model (table 1). Top, Eigenvectors (left) and eigenvalues of Gaussian 53 (probability 0.03950). Bottom, simulated patches.

Gaussian 53

We can notice that the top eigenvectors of the non-sparse Gaussian show a lot of variation. These eigenvectors correspond to highly structured patches of the images, such as the vegetation region. In the case of the sparse Gaussian, the eigenvectors show less information, these regions may correspond to less structured regions such as an edge, points, or homogeneous areas.