

The Noise Clinic

Miguel Colom, Marc Lebrun, Jean-Michel Morel

miguel.colom@cmla.ens-cachan.fr, marc.lebrun.ik@gmail.com, morel@cmla.ens-cachan.fr

CMLA, ENS Cachan

September 2012

Outline I

The noise model

$$\tilde{u} = u + n$$

where u is a (non-available) ideal image, n an additive corrupting noise and \tilde{u} the noisy image.

- u is a (non-available) ideal image,
- n an additive corrupting noise, and
- \tilde{u} the noisy image.
- **We want to measure the variance of n , only knowing a single instance of \tilde{u} .**

Noise Model

An example with $n \sim \mathcal{N}(0, \sigma^2)$ (uniform noise) with $\sigma = 50$:

$$\tilde{u} = u + n$$

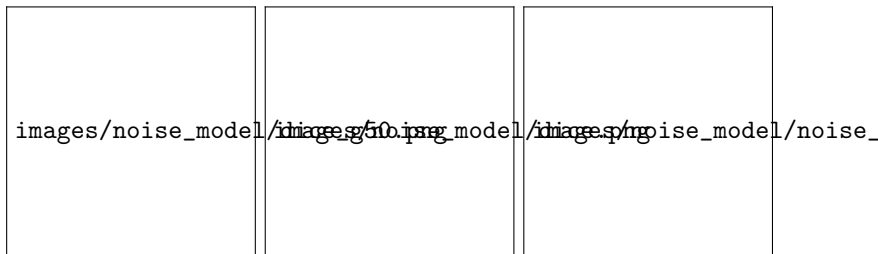


Figure: \tilde{u} (left) = u (center) + n (right). Actually, the image on the right has an offset of 127, in order to visualize the negative values of noise.

Actually, noise in a real photography is never uniform: its variance depends on u :

$$\tilde{u} = u + n(u)$$

- Due to the quantum nature of light, the photon emission by a body is not uniform but impulsive.
- It can be modelled as a **Poisson random process**:

$$f(k; \lambda t) = \frac{e^{-\lambda t} (\lambda t)^k}{k!} \sim \text{Pois}(\lambda t)$$

where k is the number of incident photons counted by the CCD and λ the expected number of incident photons/time unit.

- If k is large enough, $\text{Pois}(\lambda t) \sim \mathcal{N}(\lambda t, \lambda t)$.
- **Any noise estimation method that assumes that the noise is uniform is not realistic.**

Noise Model: Signal-Dependent (SD) noise

- In general, the variance of the SD-noise can be modeled as $\text{Var}(n_i) = a + bu_i$ where $a, b \in \mathbb{R}^+ \cup \{0\}$ are parameters of the noise and i the pixel position.



Figure: Left: noise-free image. Right: corrupted image with SD-noise with $\text{Var}(n_i) = 1 + 15u_i$

General Principles of Noise Estimation

General Principles of Noise Estimation

The general principles of noise estimation:

- 1 **Pre-filter** the image in order to get rid of deterministic tendencies.
- 2 **Local estimations**: get a local estimation of the variance and the mean at each pixel of the image using a small environment (a “**block**”).
- 3 **Classify the estimations by their mean**, in order to **estimate SD-noise**.
- 4 **Discard** those blocks whose variance is **explained mostly by the geometry** of the image and not by the noise.
- 5 Use some **statistic** on the list of estimations to get a **single robust measure** of the noise. Examples: a percentile, the median, the mean, etc.
- 6 **Correct the noise estimation** if it is biased because a percentile has been applied.

General Principles of Noise Estimation: Pre-filtering

Pre-filter the image in order to get rid of deterministic tendencies.

- The geometry of the image may contain $L1, L2, \dots$ signals. This signals contribute to the variance measured on the noisy image and therefore they have to be removed.
- Convolution of the image with some normalized discrete low-pass filters is useful.
- For example, the 3-Laplacian filter removes the $L1, L2$ and $L3$ signals in the image. Other kinds of filters are possible, like those based on the DCT or directional derivatives.

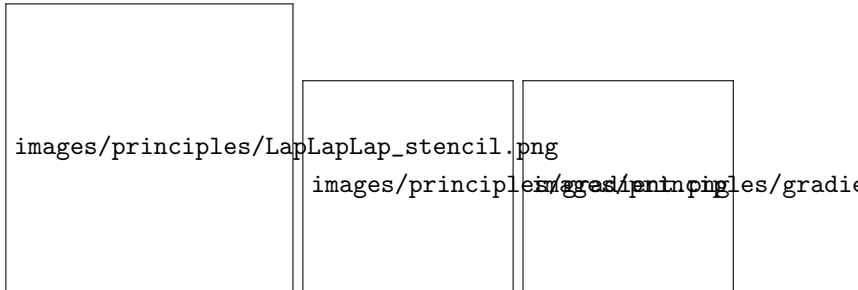


Figure: Left: Stencil of the discretized 3-Laplacian filter. Middle: Noise-free gradient signal contributing to the measured variance. Right: the image after

General Principles of Noise Estimation: Block-based noise estimation

Local estimations.

- The noise is estimated using a small environment (called a “**block**”), that is centered at each pixel of the image. Usually the blocks are small (21×21 at most).
- The **sample variance** of the pixels of the block is computed.
- Finally, **a estimation of the variance of almost each pixel of the image is obtained**. Its **mean** is also kept.

`images/noise_model/dice_block.png`

General Principles of Noise Estimation: Classify the estimations by their mean

Classify the estimations by their mean.

- A local estimation of the variance and mean of each pixel is known.
- The blocks are **classified into disjoint in mean sets** called **bins**.
- Each bin contains those **blocks whose mean belongs to a certain interval**.
- This allows for the creation of a **“noise curve”**, that is, a **link between the intensity of the pixels and their variance**.

images/perc_ponom_comparison/images/perc_ponom_comparison/curve

General Principles of Noise Estimation: Discard blocks

Discard those blocks whose variance is explained mostly by the geometry of the image.

- The geometry of the image increases the measured variance of the blocks, giving **overestimations**.
- To solve this problem, a **small percentile** of the variances of the blocks inside each bin is considered.

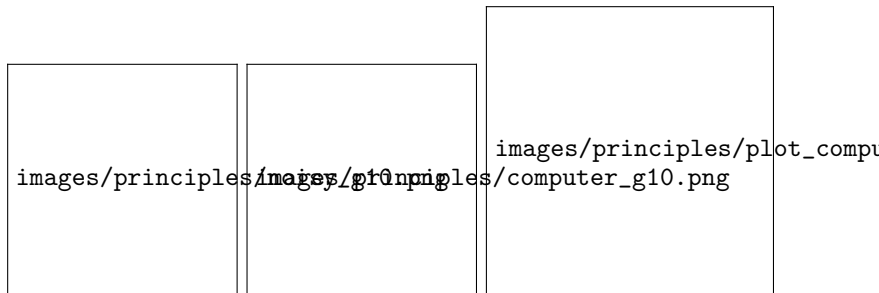


Figure: Left: pure noise of $\sigma = 10$. Center: noise-free image with noise of $\sigma = 10$ added. Right: plot of the sorted variances. **Only small percentiles give a good estimation of σ^2 .**

General Principles of Noise Estimation: Use some statistic on the list of estimations to get a single robust measure

Use some statistic on the list of estimations to get a single robust measure.

- To get a **robust estimate** of the noise, some *statistic* has to be applied **to the list of variances**.
- Typically: a **small percentile** or the **mean** or **median** of the values **below a small percentile**.

General Principles of Noise Estimation: Correct the noise estimation

Correct the noise estimation.

- When a percentile is used, the variance of the noise is **biased** and it **must be corrected**.
- The correction **depends on the percentile, the size of the block and the pre-filter operator**. It can be **empirically determined** by simulations on pure noise.
- For example, with percentile $p = 0.005$, blocks 21×21 and the 3-Laplacian filter this empirical factor learned on pure noise is ≈ 1.21 .

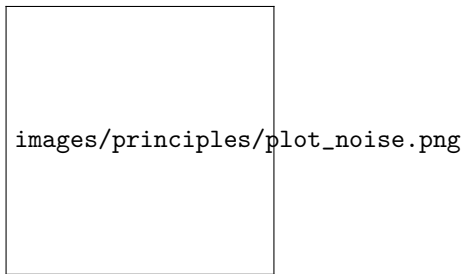


Figure: Ordered variances obtained with a 8×8 block obtained in an image of

State of the Art Noise Estimation Algorithms

Algorithm: Percentile

- Pre-filter: DCT filter with support 8×8 .
- Blocks of size from 5×5 (small images) to 21×21 pixels (big images).
- About 42000 samples/bin needed.
- Discards blocks using the 0.5% percentile.
- Statistic: value at percentile.
- Correction: multiplicative factor to correct the percentile.

Reference [?]: *Secrets of image denoising cuisine*. Acta Numerica, 2012.
(Lebrun, M. and Colom, M. and Buades, A. and Morel, J.M.)

Algorithm: Ponomarenko *et al.*

- High-pass filter: the noise is estimated only with the middle and high frequency coefficients of the 8×8 block once it has been decomposed by the DCT-II transform.
- Blocks of size 8×8 .
- About 42000 samples/bin needed.
- Discards blocks using the 0.5% percentile.
- Statistic: median of the variances under the percentile.
- **Correction: NONE.**

Reference [?]: *An Automatic Approach to Lossy Compression of AVIRIS Images*. IEEE International Geoscience and Remote Sensing Symposium, 2007. (N. N. Ponomarenko and V. V. Lukin and M. S. Zriakhov and A. Kaarna and J. T. Astola.)

Algorithm: Ponomarenko *et al.*

There is a significant difference between the *Percentile* and the *Ponomarenko* algorithms:

- *Percentile* sorts the blocks according with its variance and the applies a percentile after filtering the blocks with a low-pass filter.
- *Ponomarenko* sorts the blocks according with its variance **computed using the low-frequency coefficients** of the block, but estimates the noise with the **middle and high frequency coefficients**.
- This difference with the *Percentile* makes *Ponomarenko* give better results, since it **separates better the noise from the signal**.

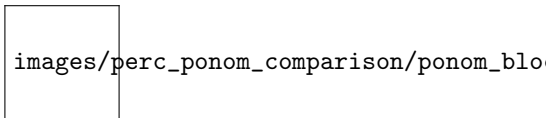


Figure: Ponomarenko method block. In green, the low-frequency coefficients used to sort the blocks. In blue, the medium and high frequency coefficients used to measure the noise of the block.

Algorithms: RMSE performance

To measure the performance of the algorithms, the following RMSE is used:

$$E_{i,\sigma}^{(1)} = \sqrt{\frac{1}{|B|} \sum_{b=0}^{|B|-1} |\hat{\sigma}_{i,b} - \sigma|^2}$$

- $|I|$ is the number of images.
- i is the image index ($0 \leq i < |I|$)
- B the number of bins
- b the index of the bin ($0 \leq b < |B|$)
- σ is the standard deviation of the simulated noise.
- $\hat{\sigma}_{i,b}$ the estimated noise for the image i at the bin b

Algorithms: Percentile. RMSE performance. Test images

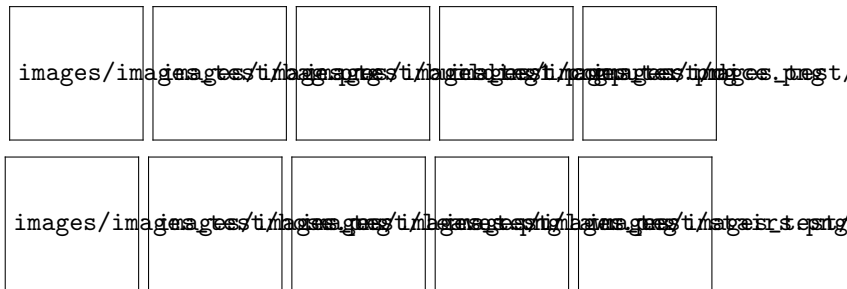


Figure: Set of noise-free images used to test the noise estimation algorithms with uniform noise. From left to right and from top to bottom: bag, building1, computer, dice, flowers2, hose, leaves, lawn, stairs and traffic.

Algorithms: RMSE performance of *Percentile*

Image / $E_{i,\sigma}^{(1)}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.77	0.67	0.52	0.45	0.96	1.03	2.76
building1	0.35	0.25	0.56	0.69	0.93	1.48	1.71
computer	0.37	0.40	0.61	0.70	0.93	1.06	2.73
dice	0.13	0.14	0.18	0.25	0.60	1.60	2.00
flowers2	0.17	0.14	0.17	0.30	0.87	1.59	3.32
hose	0.88	0.64	0.52	0.47	0.50	1.60	1.83
leaves	1.45	1.14	1.03	0.83	0.84	1.22	1.94
lawn	1.00	1.22	0.91	0.69	0.81	1.45	1.95
stairs	0.94	0.91	0.68	0.60	0.78	0.83	1.25
traffic	0.46	0.45	0.62	0.68	1.01	1.55	2.30
Flat image	0.03	0.03	0.17	0.16	0.16	1.44	2.34
ALL	0.73	0.67	0.61	0.57	0.80	1.37	2.26

Table: Percentile $E_{i,\sigma}^{(1)}$ RMSE with simulated uniform noise for the images of Fig. 9 using 8×8 blocks, percentile 0.5% and 7 bins. The last row is the RMSE using the estimated $\hat{\sigma}_{i,b}$ of all the images.

Algorithms: RMSE performance of *Ponomarenko*

Image / $E_{i,\sigma}^{(1)}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.85	0.48	0.33	0.58	0.44	1.24	1.67
building1	0.19	0.12	0.07	0.18	0.27	0.80	1.53
computer	0.21	0.12	0.16	0.25	0.32	1.97	1.18
dice	0.12	0.08	0.15	0.17	0.34	1.05	1.80
flowers2	0.18	0.09	0.10	0.29	0.38	1.53	1.10
hose	0.87	0.61	0.38	0.45	0.60	1.62	1.13
leaves	1.47	1.09	0.62	0.58	0.49	1.49	2.50
lawn	1.52	1.23	0.66	0.51	0.29	1.31	1.69
stairs	0.61	0.34	0.38	0.32	0.44	1.03	1.05
traffic	0.13	0.10	0.22	0.21	0.63	1.33	0.85
flat image	0.02	0.05	0.05	0.13	0.38	1.35	0.83
ALL	0.77	0.56	0.35	0.37	0.43	1.37	1.47

Table: Ponomarenko $E_{i,\sigma}^{(1)}$ RMSE with simulated uniform noise for the images of Fig. 9 using 8×8 blocks, percentile 0.5% and 7 bins. The last row is the RMSE using the estimated $\hat{\sigma}_{i,b}$ of all the images.

Algorithms: mean RMSE *Percentile* vs *Ponomarenko*

Method / $E_{i,\sigma}^{(1)}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
<i>Percentile</i>	0.73	0.67	0.61	0.57	0.80	1.37	2.26
<i>Pononarenko</i>	0.77	0.56	0.35	0.37	0.43	1.37	1.47

Table: Mean $E_{i,\sigma}^{(1)}$ RMSE comparison between *Percentile* and *Ponomarenko*.

Conclusion: the *Ponomarenko et al.* method is currently the best *state of the art* noise estimation method. The *Percentile* method has a similar but lower performance, with the exception of very low noise for which it gives more accurate estimations.

Checking the estimations against the GT of the camera

Checking against the GT: ISO 1250, $t=1/30s$

images/gt_ponom_perc/IMG_0187.png

images/gt_ponom_perc/images/IMG_0187_perc/ISO_1250_030

Figure: Validation of the Percentile (left) and the Ponomarenko et al. methods (right) with a raw image with ISO 1250 and exposure time $t=1/30s$

Checking against the GT: ISO 1250, $t=1/400s$

images/gt_ponom_perc/IMG_0991.png

images/gt_ponom_perc/images/IMG_0991/ISO1250/IMG4099

Figure: Validation of the Percentile (left) and the Ponomarenko et al. methods (right) with a raw image with ISO 1250 and exposure time $t=1/400s$

Checking against the GT: ISO 1600, $t=1/250s$



Figure: Validation of the Percentile (left) and the Ponomarenko et al. methods (right) with a raw image with ISO 1600 and exposure time $t=1/250s$

Checking against the GT: ISO 1600, $t=1/640s$

images/gt_ponom_perc/IMG_1111.png

images/gt_ponom_perc/images/IMG_1111_perc/ISO1600IMG640.1

Figure: Validation of the Percentile (left) and the Ponomarenko et al. methods (right) with a raw image with ISO 1600 and exposure time $t=1/640s$

Effect of the JPEG encoding

Effect of the JPEG encoding

- As part of the JPEG compression, some high-frequency coefficients of the 8×8 blocks are set to zero.
- **Problem:** most noise estimation algorithms measure the noise at the high-frequency coefficients.
- Therefore, the noise estimation algorithm will give a variance close to zero. But this is not correct, because a low-frequency noise has been left.
- The noise is no longer white but colored.
- **Solution:** estimate the noise using a multi-scale strategy and down-sample the image.

Multi-scale: Down-sample the image

- **Down-sampling:** creating a new image by substituting each block of four pixels of an image by its mean.
- The low-frequency noise (colored noise) looks like **color spots**.
- The frequency of the noise is related with the size of the spots. The bigger the spot, the lower the frequency.
- If the image is sub-sampled, the size of the spots is divided by two and therefore the frequency of the noise is increased.
- After several down-sampling iterations, the colored noise is **whitened**.
- Also, **the standard deviation of the noise is divided by two** after the down-sampling. Indeed, if n_i are samples of the noise, then
$$\text{Var}\left(\frac{n_1+n_2+n_3+n_4}{4}\right) = \frac{1}{16}\text{Var}(n_1 + n_2 + n_3 + n_4) = \frac{4\sigma^2}{16} = \frac{\sigma^2}{4} \Rightarrow$$
$$\text{Std}\left(\frac{n_1+n_2+n_3+n_4}{4}\right) = \frac{\sigma}{2}.$$

Multi-scale: Down-sampling a pure noise image

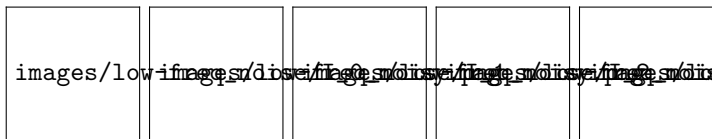
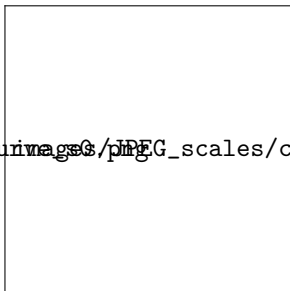


Figure: Four iterations of the down-sampling operation. The spots of the colored noise increase in frequency when down-sampled. After enough iterations, the noise can be considered white. $\sigma = 200$ at the first scale. Low-pass filter: convolution with a Gaussian G of $\sigma_G = 4.8$.

Multi-scale: Example of the four first scales in a JPEG image



Variance Stabilizing Transform

Variance Stabilizing Transform (VST)

- **Problem:**

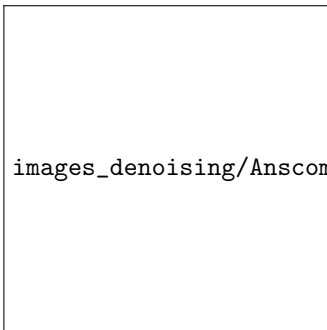
- 1 Most of current state-of-the-art denoising algorithms only deals with white Gaussian noise, i.e. $\tilde{u} \simeq u + \sigma^2 n$ (where $n \sim \mathcal{N}(0, 1)$);
- 2 In most cases of natural images the noise is signal-dependent, i.e. $\tilde{u} \simeq u + g(u)n$;

- **Solution:**

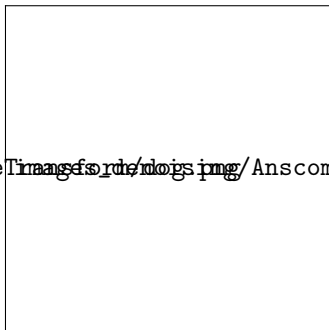
- We are looking for a variance stabilizing transform a such that $a(\tilde{u})$ has uniform deviation;
- $a(\tilde{u}) \simeq a(u) + a'(u)g(u)n$;
- Forcing the noise term to be constant, $a'(u)g(u) = c$ we get $a'(u) = \frac{c}{g(u)}$, which leads to

$$a(u) = \int_0^u \frac{cdt}{g(t)}$$

Variance Stabilizing Transform - example



Original SD-noise image
(before VST)



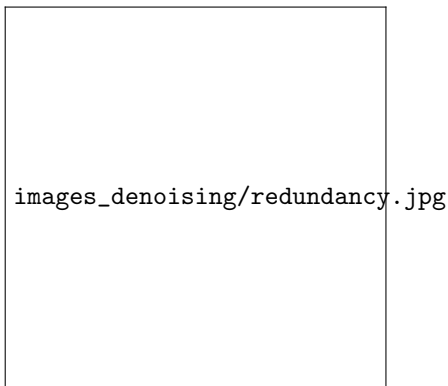
Uniform noise image
(after VST)

images_denoising/AnscombeTransform.png/images_denoising/AnscombeTransform

Redundancy of Natural Images

Redundancy

- one of the most important principles of patch-based denoising algorithms;
- it states that in all natural images similar patches can easily be found:



- then for each patch in the image it is possible to find similar patches.

NL-Bayes denoising algorithm

Bayesian denoising in two slides

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$

Bayesian denoising in two slides

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$

Bayesian denoising in two slides

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assuming that we have a patch Gaussian model

$$\mathbb{P}(Q) = c \cdot e^{-\frac{(Q-\bar{P})^t \mathbf{C}_P^{-1} (Q-\bar{P})}{2}}$$

Bayesian denoising in two slides

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assuming that we have a patch Gaussian model
$$\mathbb{P}(Q) = c \cdot e^{-\frac{(Q-\bar{P})^t \mathbf{C}_P^{-1} (Q-\bar{P})}{2}}$$
- hence the variational problem

$$\begin{aligned} \max_P \mathbb{P}(P|\tilde{P}) &\Leftrightarrow \max_P \mathbb{P}(\tilde{P}|P)\mathbb{P}(P) \\ &\Leftrightarrow \max_P e^{-\frac{\|P-\tilde{P}\|^2}{2\sigma^2}} e^{-\frac{(P-\bar{P})^t \mathbf{C}_P^{-1} (P-\bar{P})}{2}} \\ &\Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\bar{P})^t \mathbf{C}_P^{-1} (P-\bar{P}). \end{aligned}$$

Bayesian denoising in two slides

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assuming that we have a patch Gaussian model
$$\mathbb{P}(Q) = c \cdot e^{-\frac{(Q-\bar{P})^t \mathbf{C}_P^{-1} (Q-\bar{P})}{2}}$$
- hence the variational problem

$$\begin{aligned} \max_P \mathbb{P}(P|\tilde{P}) &\Leftrightarrow \max_P \mathbb{P}(\tilde{P}|P)\mathbb{P}(P) \\ &\Leftrightarrow \max_P e^{-\frac{\|P-\tilde{P}\|^2}{2\sigma^2}} e^{-\frac{(P-\bar{P})^t \mathbf{C}_P^{-1} (P-\bar{P})}{2}} \\ &\Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\bar{P})^t \mathbf{C}_P^{-1} (P-\bar{P}). \end{aligned}$$

- An empirical covariance matrix $\mathbf{C}_{\tilde{P}}$ can be obtained for the patches \tilde{Q} similar to \tilde{P} . P and the noise n being independent,
$$\mathbf{C}_{\tilde{P}} = \mathbf{C}_P + \sigma^2 \mathbf{I}; \quad E\tilde{Q} = \bar{P}$$

Bayesian denoising in two slides

- $\max_P \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \frac{\|P - \tilde{P}\|^2}{\sigma^2} + (P - \bar{P})^t (\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I})^{-1} (P - \bar{P})$

Bayesian denoising in two slides

- $\max_P \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \frac{\|P - \tilde{P}\|^2}{\sigma^2} + (P - \bar{P})^t (\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I})^{-1} (P - \bar{P})$
- One step estimation

$$\hat{P}_1 = \bar{P} + [\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I}] \mathbf{C}_{\tilde{P}}^{-1} (\tilde{P} - \bar{P}),$$

where empirically:

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P}) - 1} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} (\tilde{Q} - \bar{P})(\tilde{Q} - \bar{P})^t, \quad \bar{P} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \tilde{Q}.$$

Bayesian denoising in two slides

- $\max_P \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \frac{\|P - \tilde{P}\|^2}{\sigma^2} + (P - \tilde{P})^t (\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I})^{-1} (P - \tilde{P})$
- One step estimation

$$\hat{P}_1 = \tilde{P} + [\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I}] \mathbf{C}_{\tilde{P}}^{-1} (\tilde{P} - \tilde{P}),$$

where empirically:

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P}) - 1} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} (\tilde{Q} - \tilde{P})(\tilde{Q} - \tilde{P})^t, \quad \tilde{P} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \tilde{Q}.$$

- Iteration (“oracle estimation”) :

$$\hat{P}_2 = \tilde{P}^{-1} + \mathbf{C}_{\hat{P}_1} [\mathbf{C}_{\hat{P}_1} + \sigma^2 \mathbf{I}]^{-1} (\tilde{P} - \tilde{P}^{-1})$$

where

$$\mathbf{C}_{\hat{P}_1} \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1) - 1} \sum_{\hat{Q}_1 \in \mathcal{P}(\hat{P}_1)} (\hat{Q}_1 - \tilde{P}^{-1})(\hat{Q}_1 - \tilde{P}^{-1})^t, \quad \tilde{P}^{-1} \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)} \sum_{\hat{Q}_1 \in \mathcal{P}(\hat{P}_1)} \tilde{Q}.$$

Results

Single Scale VS Multi-Scales

`images_denoising/Results/Dice_noisy_60.png`

Single Scale VS Multi-Scales

images_denoising/Results/Dice_nl-bayes_60.png

Single Scale VS Multi-Scales

`images_denoising/Results/Dice_msd-mean_60.png`

Single Scale VS Multi-Scales

`images_denoising/Results/Girl_noisy_30.png`

Single Scale VS Multi-Scales

images_denoising/Results/Girl_nl-bayes_30.png

Single Scale VS Multi-Scales

`images_denoising/Results/Girl_msd-mean_30.png`

Influence of the Number of Scales - Noisy Image

`images_denoising/Results/synthetic.png`

Influence of the Number of Scales - Noise Clinic (3 scales)

`images_denoising/Results/synthetic_mean3.png`

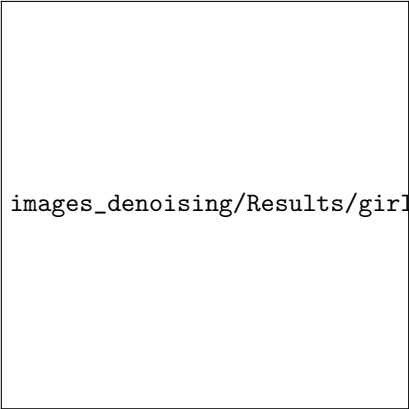
Influence of the Number of Scales - Noise Clinic (4 scales)

`images_denoising/Results/synthetic_mean4.png`

Influence of the Number of Scales - Noisy Image

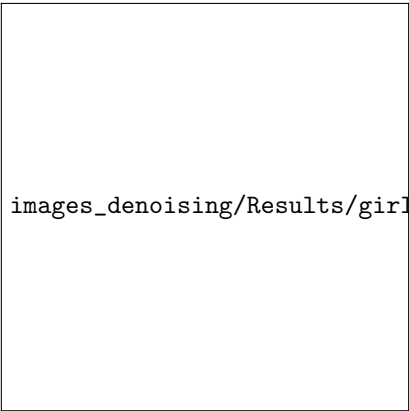


Influence of the Number of Scales - Noise Clinic (3 scales)



`images_denoising/Results/girl1_night_area_mean3.pr`

Influence of the Number of Scales - Noise Clinic (4 scales)



`images_denoising/Results/girl_night_area_mean4.pr`

Results of Natural Images - Noisy Image

`images_denoising/Results/Cards.png`

Results of Natural Images - Noise Clinic (4 scales)

`images_denoising/Results/Cards_mean4.png`

Results of Natural Images - Noisy Image

`images_denoising/Results/Frog.png`

Results of Natural Images - Noise Clinic (3 scales)

`images_denoising/Results/Frog_mean3.png`

Results of Natural Images - Noisy Image

`images_denoising/Results/Postcard.png`

Results of Natural Images - Noise Clinic (3 scales)

`images_denoising/Results/Postcard_mean3.png`

Results of Natural Images - Noisy Image

`images_denoising/Results/Lena.png`

Results of Natural Images - Noise Clinic (3 scales)

`images_denoising/Results/Lena_mean3.png`

Results of Natural Images - Noisy Image

images_denoising/Results/Marylin.png

Results of Natural Images - Noise Clinic (3 scales)

images_denoising/Results/Marylin_mean3.png

How to try it

A prototype of *noise clinic* is currently on line at
http://dev.ipol.im/~colom/ipol_demo/noise_clinic/
(username: demo, password: demo).

Other algorithms at Image Processing On Line <http://www.ipol.im/>:

BM3D

DCT-denoising

K-SVD

NL-Bayes

NL-means

TV-denoising

Soon: *PLE, BLS-GSM*

