The noise clinic

Miguel Colom, Marc Lebrun, Jean-Michel Morel

CMLA, ENS Cachan

August 2012

How to try it

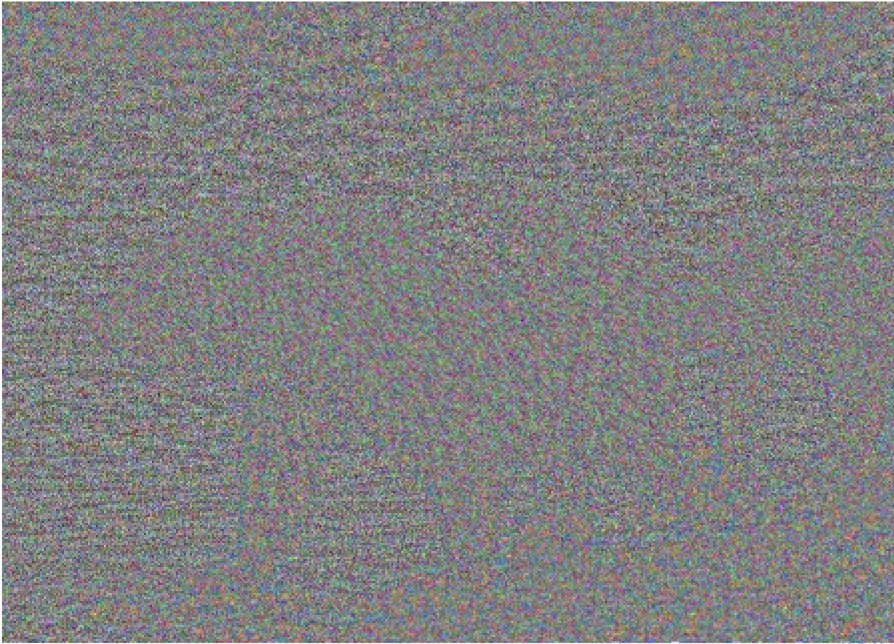
A prototype of *noise clinic* is currently on line at
`http://dev.ipol.im/~colom/ipol_demo/noise_clinic/`
(username: demo, password: demo).
Other algorithms at Image Processing on Line `http://www.ipol.im/`:
DCT-denoising
TV-denoising
K-SVD
NL-means
NL-Bayes
Soon: *PLE, BLS-GSM*

Outline I

Noisy

Denoised

Difference



Noise curves

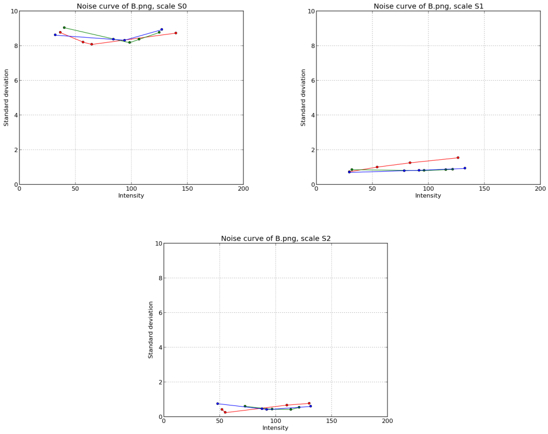
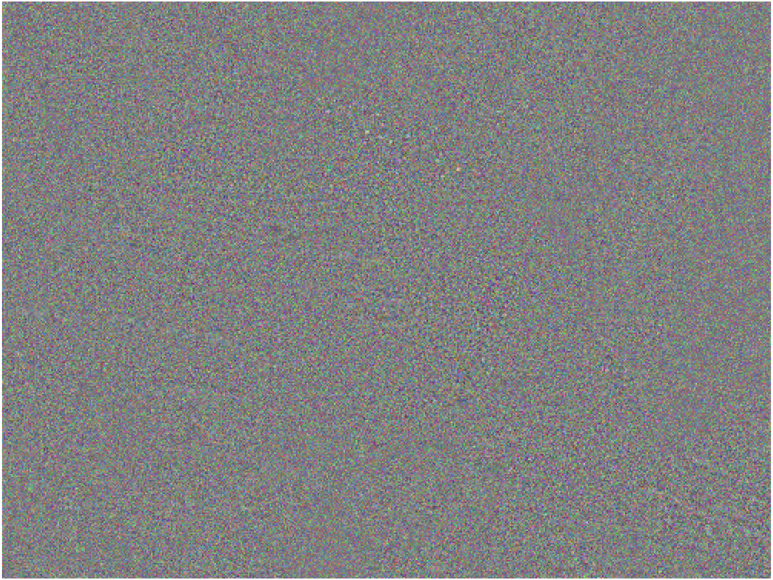


Figure: Noise curves after denoising for image Bears, 3 first scales.

Noisy

Denoised

Difference



Noise curves

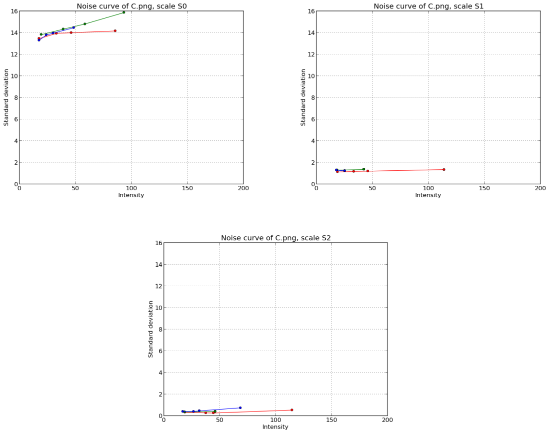
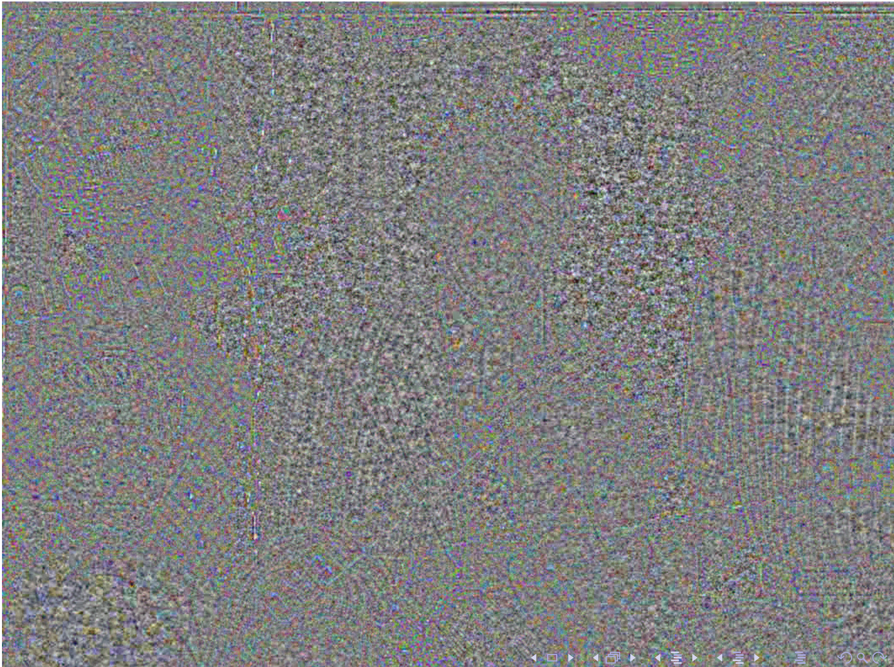


Figure: Noise curves after denoising for image Frog, 3 first scales.

Noisy

Denoised

Difference



Noise curves

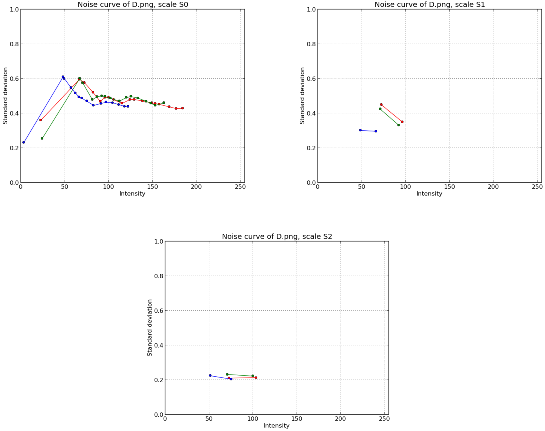
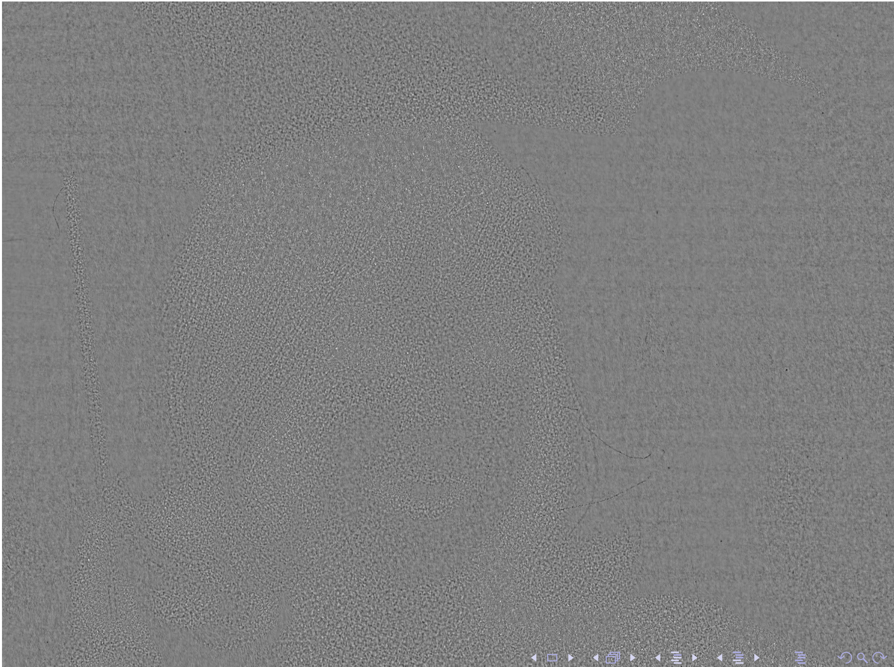


Figure: Noise curves after denoising for image Old Picture, 3 first scales.

Noisy

Denoised

Difference



Noise curves

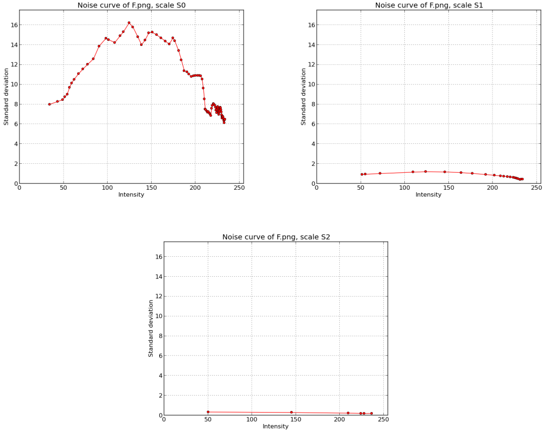
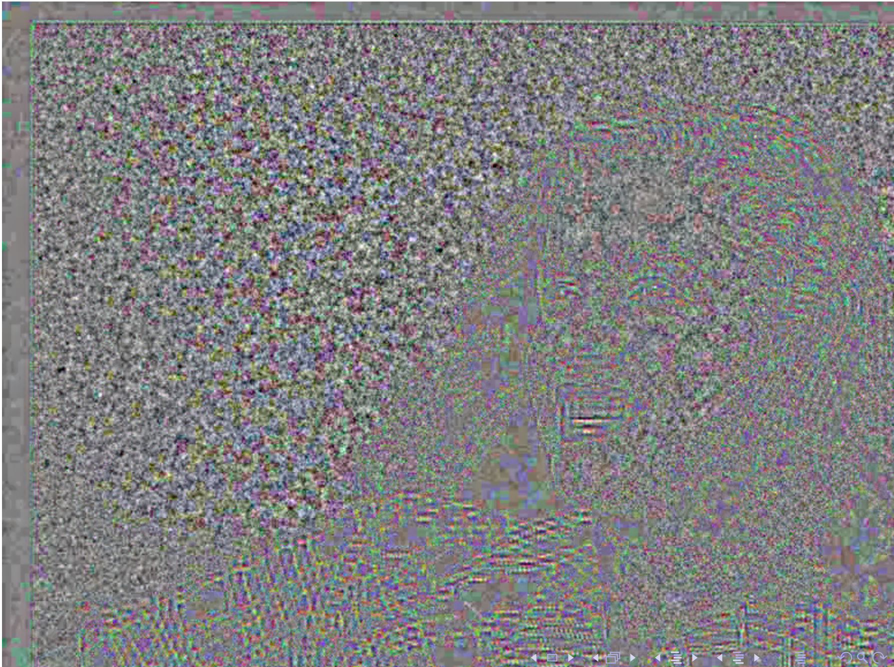


Figure: Noise curves after denoising for image Marilyn 1, 3 first scales.

Noisy

Denoised

Difference



Noise curves

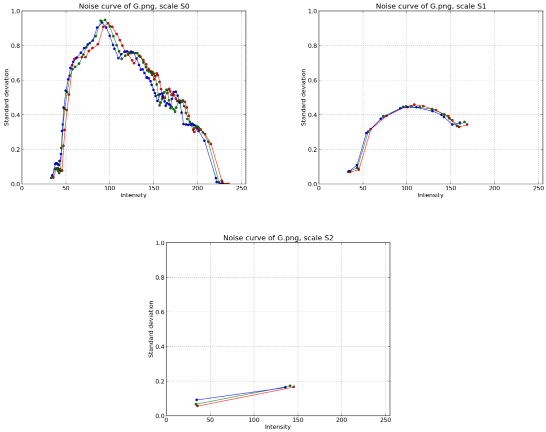


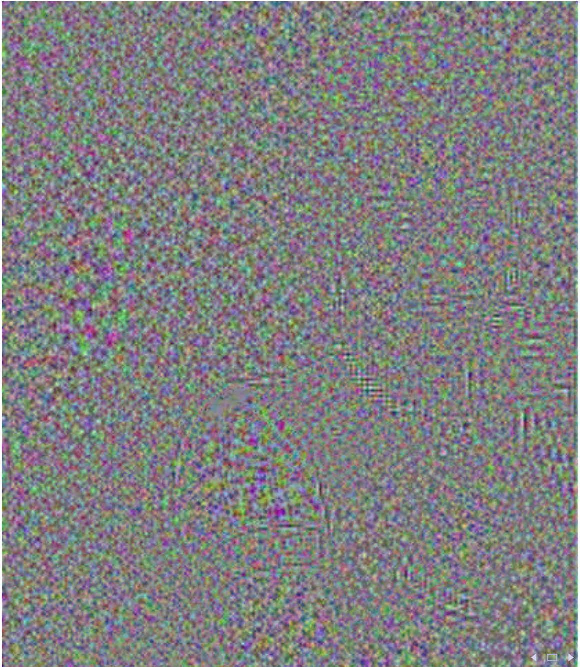
Figure: Noise curves after denoising for image Marilyn 2, 3 first scales.

An example of test image with wrong noise estimation

Noisy

Denoised

Difference



Noise curves

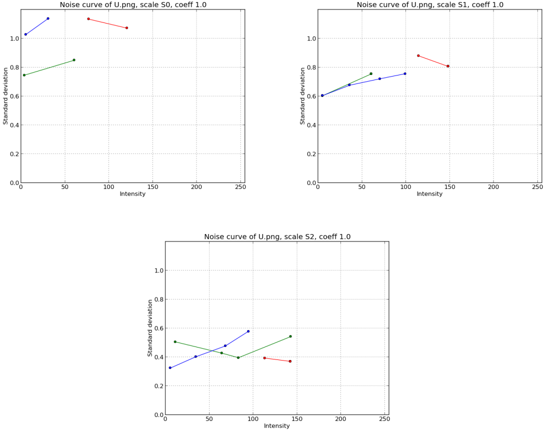


Figure: Noise curves after denoising for image Singer, 3 first scales.

Figure: Comparison setting coefficient $c = 1.0$, $c = 3.0$, $c = 4.0$.

Figure: Comparison setting coefficient $c = 1.0$, $c = 3.0$, $c = 4.0$.

Figure: Comparison setting coefficient $c = 1.0$, $c = 3.0$, $c = 4.0$.

Noise model, and how it becomes complex from raw to JPEG

- Photon emission can be modelled with a random Poisson distribution due to the physical nature of light.

$$P(N=k)=\frac{e^{-\lambda t}(\lambda t)^k}{k!}$$

  where k is the number of photons counted by the CCD, λ the expected number of photons/time unit.
- If N is large enough, $N \sim \mathcal{N}(\lambda t, \lambda t)$.

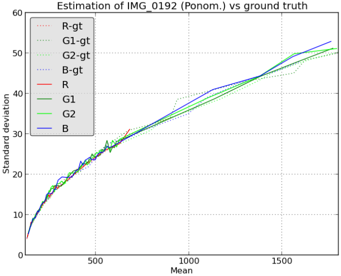


Figure: Left: Canon EOS 30D, ISO 1600, t=1/30s image. Right: noise curve of the raw image obtained with the Ponomarenko et al. algorithm.

- Most of the denoising algorithms only deal with uniform Gaussian noise.
- But raw images follow a Poisson distribution where the variance is proportional to the intensity.
- Solution: use a Variance Stabilizing Transformation:
- Anscombe transformation [1] ($\Rightarrow \sigma^2 \approx 1$).

$$u \mapsto 2\sqrt{u + \frac{3}{8}}$$

(forward)

$$v \mapsto \left(\frac{v}{2}\right)^2 - \frac{3}{8}$$

(inverse)

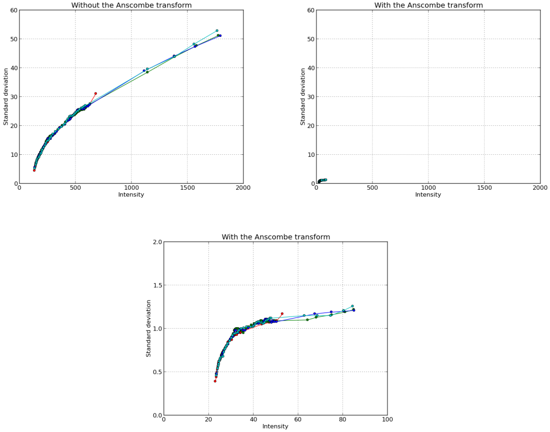


Figure: Up: without and with the Anscombe tranformation. Down: detailed view with the Anscombe transformation.

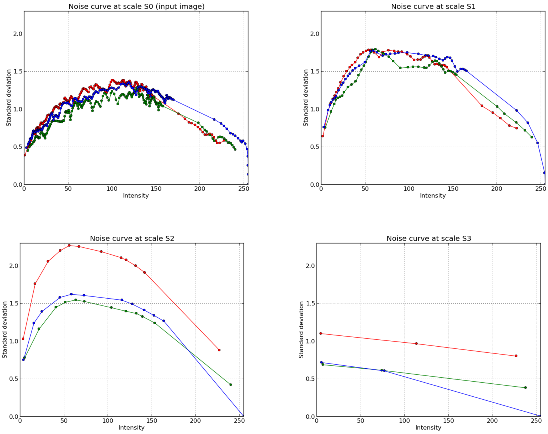


Figure: Typical JPEG noise curves at scales S0, S1, S2 and S3..

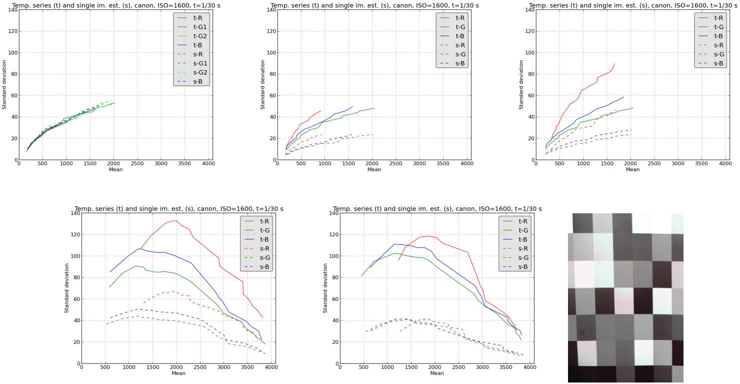


Figure: Effect of the complete IP pipeline for ISO 1600, t=1/30s, Canon EOS 30D: raw image, demosaicing, white balance, gamma correction and JPEG compression using photograph of a calibration pattern.

Noise estimation methods

General principles (for white noise first)

- Block-based methods.
- Get a estimation of the variance of data (noise+signal) inside each block using only the high frequencies of the block spectrum. The DCT[1] works quite well.
- Consider blocks whose variance is under a low quantile (typically 0.5%): keep only those blocks whose variance is explained mostly by the noise.
- Get the final estimation by combining the blocks in the low quantile. Typically, computing their median or using the MAD[2] estimator. Learn on noise the correction factor.

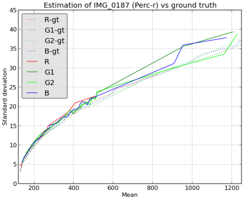
[1] Discrete Cosine Transform.
[2] Median of Absolute Deviations.

Ponomarenko at al. noise estimation method

- Uses 8×8 **overlapping blocks**.
- Sort the blocks by the variance of the **lower frequency** DCT coefficients of the block. Keep the lower (0.5%) quantile,
- The variance of the noise is estimated on the **medium and high frequency** coefficients of the blocks of this lower quantile.
- The **median** value of these variances gives the final variance estimation.

Noise estimation algorithm #2: Percentile method

- The image is **high-passed** by convolving it with a **filter based on the DCT with support** 7×7 to get rid of deterministic tendencies.
- Compute the variance of all 21×21 **overlapping blocks**
- To discard those blocks whose variance is explained by the signal and not by the noise consider only the **small** (0.5%) **quantile**.
- The **median** of variances of this set gives the final variance estimation.



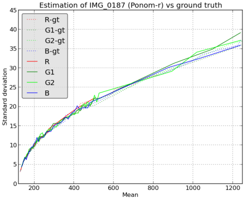


Figure: Validation of the Ponomarenko et al. (left) and the Percentile methods (right) with a raw image with ISO 1250 and exposure time $t=1/30$s.

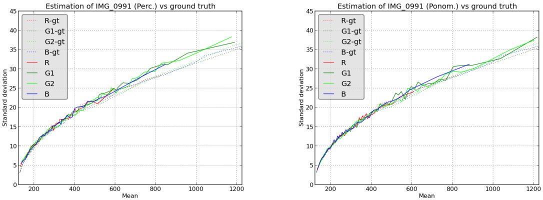


Figure: Validation of the Ponomarenko et al. (left) and the Percentile methods (right) with a raw image with ISO 1250 and exposure time t=1/400s.

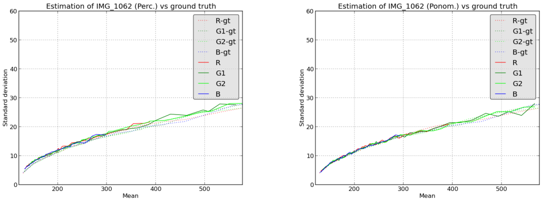


Figure: Validation of the Ponomarenko et al. (left) and the Percentile methods (right) with a raw image with ISO 1600 and exposure time t=1/250s.

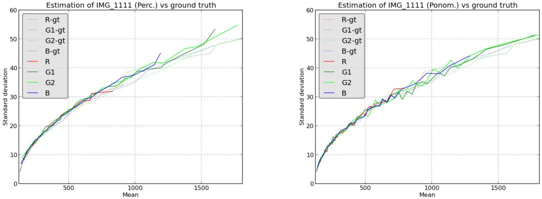
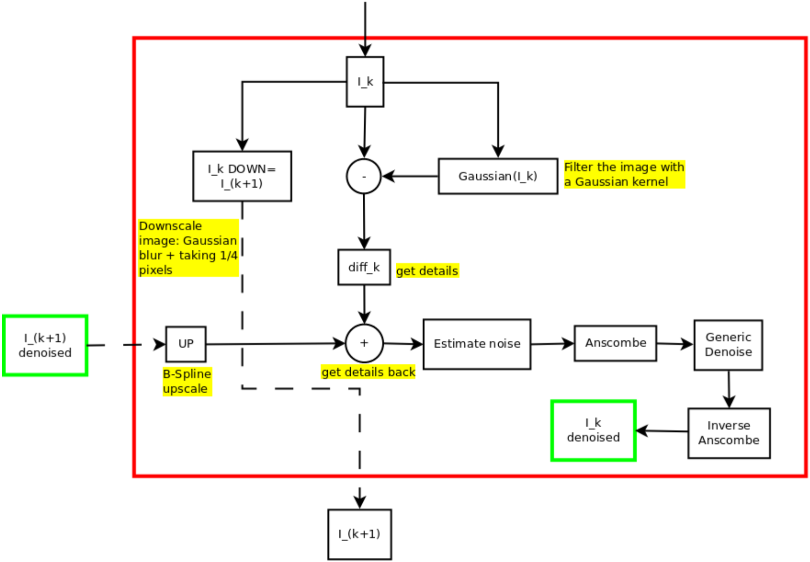
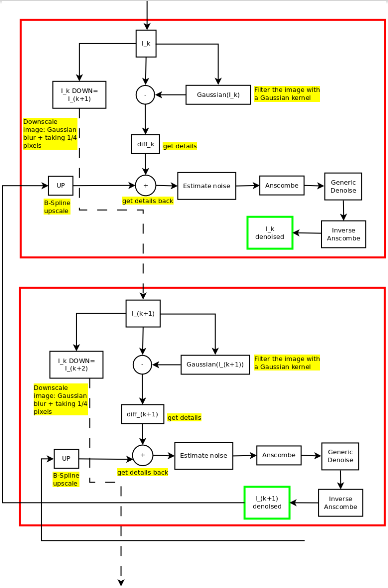


Figure: Validation of the Ponomarenko et al. (left) and the Percentile methods (right) with a raw image with ISO 1600 and exposure time t=1/640s.

Multiscale denoising: principles

- signal dependent noise estimated at each scale
- zoom down followed by Anscombe transform to whiten the noise at each scale
- denoising performed at each scale, bottom-up (coarse to fine)
- Useful even for white noise: the denoising performance extends to very low frequencies


I_k
I_k DOWN=
I_(k+1)
-
Gaussian(I_k)
Filter the image with
a Gaussian kernel
Downscale
image: Gaussian
blur + taking 1/4
pixels
diff_k
get details
I_(k+1)
denoised
UP
+
Estimate noise
Anscombe
Generic
Denoise
B-Spline
upscale
get details back
I_k
denoised
Inverse
Anscombe
I_(k+1)


I_k
I_k DOWN= I_(k+1)
-
Gaussian(I_k)
Filter the image with a Gaussian kernel
Downscale image: Gaussian blur + taking 1/4 pixels
diff_k
get details
UP
+
Estimate noise
Anscombe
Generic Denoise
B-Spline upscale
get details back
I_k denoised
Inverse Anscombe
I_(k+1)
I_k DOWN= I_(k+2)
-
Gaussian(I_(k+1))
Filter the image with a Gaussian kernel
Downscale image: Gaussian blur + taking 1/4 pixels
diff_(k+1)
get details
UP
+
Estimate noise
Anscombe
Generic Denoise
B-Spline upscale
get details back
I_(k+1) denoised
Inverse Anscombe

More on the denoising algorithm: Non local Bayesian denoising

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 $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assume we got a patch Gaussian model $\mathbb{P}(Q) = c.e^{-\frac{(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}}$

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 $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assume we got a patch Gaussian model $\mathbb{P}(Q) = c.e^{-\frac{(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}}$
- hence the variational problem

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

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 $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assume we got a patch Gaussian model $\mathbb{P}(Q) = c.e^{-\frac{(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}}$
- hence the variational problem

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

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

Bayesian denoising in two slides

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

Bayesian denoising in two slides

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

one step estimation $\hat{P}_1 = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I}\right] \mathbf{C}_{\tilde{P}}^{-1}(\tilde{P} - \overline{\tilde{P}})$, where empirically:

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

Bayesian denoising in two slides

$\max_P \ \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \ \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P - \overline{\tilde{P}})^t (\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I})^{-1} (P - \overline{\tilde{P}})$
one step estimation $\hat{P}_1 = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2 \mathbf{I}\right] \mathbf{C}_{\tilde{P}}^{-1} (\tilde{P} - \overline{\tilde{P}})$, where empirically:

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

Iteration ("oracle estimation"): $\hat{P}_2 = \overline{\tilde{P}}^1 + \mathbf{C}_{\hat{P}_1} \left[\mathbf{C}_{\hat{P}_1} + \sigma^2 \mathbf{I}\right]^{-1} (\tilde{P} - \overline{\tilde{P}}^1)$
where

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

All Bayesian or Bayesian-like methods

Method	Denoising principle	Patches	size	Aggr.	Oracle	C
DCT	transform threshold	one	8	yes	yes	y
NL-Means	average	neighborhood	3	yes	yes	n
NL-Bayes	Bayes	neighborhood	3-7	yes	yes	y
PLOW	Bayes, 15 clusters	image	11	yes	yes	y
Shotgun	Bayes	10^{10} patches	3-20	yes	no	n
EPLL	Bayes, 200 clusters	2.10^{10} patches	8	yes	yes	y
BLS-GSM	Bayes in GSM	Image	3	yes	no	n
K-SVD	sparse dictionary	Image	8	yes	yes	y
BM3D	transform threshold	neighborhood	8-12	yes	yes	y
PLE	Bayes, 19 clusters	Image	8	yes	yes	y

See: “Secrets of image denoising cuisine” M. Lebrun, M. Colom, A. Buades, J.M.M., *Acta Numerica*, 2012

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis
- this is equivalent to Bayesian MMSE assuming that this basis gives a Gaussian model for the patch

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis
- this is equivalent to Bayesian MMSE assuming that this basis gives a Gaussian model for the patch
- coefficient thresholding (DCT) or averaging (NL-means) is suboptimal. By making them Bayesian, DCT thresholding becomes a DCT Wiener filter, and NL-means becomes NL-Bayes

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis
- this is equivalent to Bayesian MMSE assuming that this basis gives a Gaussian model for the patch
- coefficient thresholding (DCT) or averaging (NL-means) is suboptimal. By making them Bayesian, DCT thresholding becomes a DCT Wiener filter, and NL-means becomes NL-Bayes
- the mentioned methods differ only by the way the Gaussian mixture is constructed (global or local)

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis
- this is equivalent to Bayesian MMSE assuming that this basis gives a Gaussian model for the patch
- coefficient thresholding (DCT) or averaging (NL-means) is suboptimal. By making them Bayesian, DCT thresholding becomes a DCT Wiener filter, and NL-means becomes NL-Bayes
- the mentioned methods differ only by the way the Gaussian mixture is constructed (global or local)
- Shotgun is the ideal Bayesian MMSE algorithm. It makes no assumption at all (no Gaussianity)

Conclusions in a nutshell:

- all methods except NL-means and Shotgun find an orthogonal or sparse basis for each patch.
- the final estimate is obtained by Wiener filter: applying optimal multiplicative coefficients to the coordinates of the patch on the basis
- this is equivalent to Bayesian MMSE assuming that this basis gives a Gaussian model for the patch
- coefficient thresholding (DCT) or averaging (NL-means) is suboptimal. By making them Bayesian, DCT thresholding becomes a DCT Wiener filter, and NL-means becomes NL-Bayes
- the mentioned methods differ only by the way the Gaussian mixture is constructed (global or local)
- Shotgun is the ideal Bayesian MMSE algorithm. It makes no assumption at all (no Gaussianity)
- All methods must be compared after applying the three denoising improvement tricks: color transform, aggregation, oracle iteration.

All Bayesian or Bayesian-like methods, references

DCT denoising: G. Sapiro and G. Yu, IPOL 2011.
NL-Bayes: A. Buades, M. Lebrun, J.M.M., IPOL 2012.
PLOW: P. Chatterjee and P. Milanfar, TIP 2011.
Shotgun: A. Levin and B. Nadler, CVPR 2011.
EPLL: D. Zoran and Y. Weiss, ICCV 2011.
BLS-GSM: J. Portilla, V. Strela, M.J. Wainwright, and E.P. Simoncelli, TIP 2003.
KSVD: M. Elad, M. Aharon, TIP 2006.
BM3D: K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, TIP 2007.
PLE: G. Yu, G. Sapiro, and S. Mallat, TIP 2010.

Results of denoising a pure noise image ($\sigma = 30$).

Method	PSNR	RMSE
NL-Bayes	45.45	1.36
BM3D	45.03	1.43
NL-means	41.45	2.16
TV denoising	41.06	2.26
DCT denoising	40.91	2.30
K-SVD	38.44	3.05

PSNR table for $\sigma = 20$, 30 and 40

	$\sigma = 20$				
NL-Bayes	BM3D	BLS-GSM	K-SVD	NL-means	DCT denoising
33.45	33.22	32.61	32.25	31.98	32.20

PSNR table for $\sigma = 20$, 30 and 40

	$\sigma = 20$				
NL-Bayes	BM3D	BLS-GSM	K-SVD	NL-means	DCT denoising
33.45	33.22	32.61	32.25	31.98	32.20
	$\sigma = 30$				
31.37	31.17	30.31	30.48	29.77	29.83

PSNR table for $\sigma = 20$, 30 and 40

	$\sigma = 20$				
NL-Bayes	BM3D	BLS-GSM	K-SVD	NL-means	DCT denoising
33.45	33.22	32.61	32.25	31.98	32.20
	$\sigma = 30$				
31.37	31.17	30.31	30.48	29.77	29.83
	$\sigma = 40$				
30.15	29.71	28.94	28.90	28.25	28.05

Figure: Original, noisy, DCT sliding window, BLS-GSM

Figure: Original, noisy, NL-means, K-SVD

Figure: Original, noisy, BM3D and Non-local Bayes.

Ideal Bayesian method: Shotgun NL-means (A. Levin, B. Nadler 2011)

$$\mathbb{P}(\tilde{P} \mid P) = \frac{1}{(2\pi\sigma^2)^{\frac{\kappa^2}{2}}} e^{-\frac{||P-\tilde{P}||^2}{2\sigma^2}}, \tag{1}$$

Ideal Bayesian method: Shotgun NL-means (A. Levin, B. Nadler 2011)

$$\mathbb{P}(\tilde{P} \mid P) = \frac{1}{(2\pi\sigma^2)^{\frac{\kappa^2}{2}}} e^{-\frac{||P-\tilde{P}||^2}{2\sigma^2}}, \tag{1}$$

Given a noisy patch $\tilde{P}$ its optimal estimator for the Bayesian minimum squared error (MMSE) is by Bayes' formula

$$\hat{P} = \mathbb{E}[P \mid \tilde{P}] = \int \mathbb{P}(P \mid \tilde{P}) P dP = \int \frac{\mathbb{P}(\tilde{P} \mid P)}{\mathbb{P}(\tilde{P})} \mathbb{P}(P) P dP. \tag{2}$$

Ideal Bayesian method: Shotgun NL-means (A. Levin, B. Nadler 2011)

$$\mathbb{P}(\tilde{P} \mid P) = \frac{1}{(2\pi\sigma^2)^{\frac{\kappa^2}{2}}} e^{-\frac{||P-\tilde{P}||^2}{2\sigma^2}}, \tag{1}$$

Given a noisy patch $\tilde{P}$ its optimal estimator for the Bayesian minimum squared error (MMSE) is by Bayes' formula

$$\hat{P} = \mathbb{E}[P \mid \tilde{P}] = \int \mathbb{P}(P \mid \tilde{P}) P dP = \int \frac{\mathbb{P}(\tilde{P} \mid P)}{\mathbb{P}(\tilde{P})} \mathbb{P}(P) P dP. \tag{2}$$

Using a huge set of M natural patches,
$\mathbb{P}(P)dP \simeq \frac{1}{M}$ and $\mathbb{P}(\tilde{P}) \simeq \frac{1}{M}\sum_i \mathbb{P}(\tilde{P} \mid P_i)$. Thus

$$\hat{P} \simeq \frac{\frac{1}{M}\sum_i \mathbb{P}(\tilde{P} \mid P_i) P_i}{\frac{1}{M}\sum_i \mathbb{P}(\tilde{P} \mid P_i)}.$$

Shotgun NL-means

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$

Shotgun NL-means

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$
- **Input**: Very large set of $M = 2^{10}$ patches P_i extracted from a large set of noiseless natural images (20000)

Shotgun NL-means

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$
- **Input**: Very large set of $M = 2^{10}$ patches P_i extracted from a large set of noiseless natural images (20000)
- **Output**: Denoised image $\hat{u}$.

Shotgun NL-means

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$
- **Input**: Very large set of $M = 2^{10}$ patches P_i extracted from a large set of noiseless natural images (20000)
- **Output**: Denoised image $\hat{u}$.
- **for all patches $\tilde{P}$ extracted from $\tilde{u}$:** Compute the MMSE denoised estimate of $\tilde{P}$

$$\hat{P} \simeq \frac{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i) P_i}{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i)}$$

where $\mathbb{P}(\tilde{P} \mid P_i)$ is known from the Gaussian noise distribution.

Shotgun NL-means

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$
- **Input**: Very large set of $M = 2^{10}$ patches P_i extracted from a large set of noiseless natural images (20000)
- **Output**: Denoised image $\hat{u}$.
- **for all patches $\tilde{P}$ extracted from $\tilde{u}$:** Compute the MMSE denoised estimate of $\tilde{P}$

$$\hat{P} \simeq \frac{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i) P_i}{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i)}$$

  where $\mathbb{P}(\tilde{P} \mid P_i)$ is known from the Gaussian noise distribution.
- (Aggregation) : for each pixel $\mathbf{j}$ of u, compute the denoised version $\hat{u}_{\mathbf{j}}$ as the average of all values $\hat{P}(\mathbf{j})$ for all patches containing $\mathbf{j}$.

Denoising recipes illustrated by DCT

Noise reduction, generic recipes

- Aggregation of estimates (of patches containing a given pixel)
- Iteration and oracle filters: use first step result as oracle for second step
- Color: convert (R, G, B) into (Y, U, V).

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_oU_oV_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_o U_o V_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_o U_o V_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_o U_o V_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_o U_o V_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_oU_oV_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_oU_oV_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

Tricks improving denoising performance

Figure: Original, noisy ($\sigma = 25$), sliding DCT thresholding filter, incremental use of a $Y_o U_o V_o$ colour system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. Corresponding PSNRs : 26.85, 27.33, 30.65, 30.73, 31.25.

DCT denoising algorithm, step 1

Cancels DCT coefficients lower than 3σ. Applied independently to each $Y_o U_o V_o$ component.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, $\kappa = 8$: size of patches, $h = 3\,\sigma$: threshold parameter.

DCT denoising algorithm, step 1

Cancels DCT coefficients lower than 3σ. Applied independently to each $Y_o U_o V_o$ component.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, $\kappa = 8$: size of patches, $h = 3\,\sigma$: threshold parameter.
- **For each noisy patch:** $\tilde{P}$: Compute the DCT transform of $\tilde{P}$.

DCT denoising algorithm, step 1

Cancels DCT coefficients lower than 3σ. Applied independently to each $Y_o U_o V_o$ component.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, $\kappa = 8$: size of patches, $h = 3\,\sigma$: threshold parameter.
- **For each noisy patch:** $\tilde{P}$: Compute the DCT transform of $\tilde{P}$.
- Cancel coefficients of $\tilde{P}$ with magnitude lower than $h = 3\,\sigma$.

DCT denoising algorithm, step 1

Cancels DCT coefficients lower than 3σ. Applied independently to each $Y_oU_oV_o$ component.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, $\kappa = 8$: size of patches, $h = 3\,\sigma$: threshold parameter.
- **For each noisy patch:** $\tilde{P}$: Compute the DCT transform of $\tilde{P}$.
- Cancel coefficients of $\tilde{P}$ with magnitude lower than $h = 3\,\sigma$.
- Compute the inverse DCT transform obtaining $\hat{P}$.

DCT denoising algorithm, step 1

Cancels DCT coefficients lower than 3σ. Applied independently to each $Y_o U_o V_o$ component.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, $\kappa = 8$: size of patches, $h = 3\,\sigma$: threshold parameter.
- **For each noisy patch:** $\tilde{P}$: Compute the *DCT* transform of $\tilde{P}$.
- Cancel coefficients of $\tilde{P}$ with magnitude lower than $h = 3\,\sigma$.
- Compute the inverse DCT transform obtaining $\hat{P}$.
- Compute the aggregation weight $w_{\tilde{P}} = 1/\#\{\text{number of non-zero } DCT \text{ coefficients}\}$.
- **for each pixel i:** (aggregation) average all values at **i** of all denoised patches $\hat{Q}$ containing **i**, weighted by $w_{\tilde{Q}}$.

DCT denoising algorithm, step 2

A Wiener filter is applied in the "oracle" second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for "oracle" estimation.

DCT denoising algorithm, step 2

A Wiener filter is applied in the "oracle" second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for "oracle" estimation.
- **For each pixel i:**
- Take reference patches $\tilde{P}$ and P_1 around $\mathbf{i}$ in $\tilde{u}$ and $\hat{u}_1$.

DCT denoising algorithm, step 2

A Wiener filter is applied in the "oracle" second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for "oracle" estimation.
- **For each pixel i:**
- Take reference patches $\tilde{P}$ and P_1 around **i** in $\tilde{u}$ and $\hat{u}_1$.
- Compute the DCT transforms of $\tilde{P}$ and P_1.

DCT denoising algorithm, step 2

A Wiener filter is applied in the “oracle” second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for “oracle” estimation.
- **For each pixel i:**
- Take reference patches $\tilde{P}$ and P_1 around **i** in $\tilde{u}$ and $\hat{u}_1$.
- Compute the *DCT* transforms of $\tilde{P}$ and P_1.
- Modify DCT coefficients of $\tilde{P}$ as $\tilde{P}(i) = \tilde{P}(i)\frac{P_1(i)^2}{P_1(i)^2+\sigma^2}$.

DCT denoising algorithm, step 2

A Wiener filter is applied in the "oracle" second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for "oracle" estimation.
- **For each pixel i:**
- Take reference patches $\tilde{P}$ and P_1 around **i** in $\tilde{u}$ and $\hat{u}_1$.
- Compute the *DCT* transforms of $\tilde{P}$ and P_1.
- Modify DCT coefficients of $\tilde{P}$ as $\tilde{P}(i) = \tilde{P}(i)\frac{P_1(i)^2}{P_1(i)^2+\sigma^2}$.
- Compute the inverse DCT transform obtaining $\hat{P}$.

DCT denoising algorithm, step 2

A Wiener filter is applied in the “oracle” second step.

- **Input**: noisy image $\tilde{u}$, σ noise standard deviation, prefiltered image $\hat{u}_1$ for “oracle” estimation.
- **For each pixel i:**
- Take reference patches $\tilde{P}$ and P_1 around **i** in $\tilde{u}$ and $\hat{u}_1$.
- Compute the *DCT* transforms of $\tilde{P}$ and P_1.
- Modify DCT coefficients of $\tilde{P}$ as $\tilde{P}(i) = \tilde{P}(i)\frac{P_1(i)^2}{P_1(i)^2+\sigma^2}$.
- Compute the inverse DCT transform obtaining $\hat{P}$.
- Aggregation: $w_{\tilde{P}} = 1/\#\{\text{number of non-zero } DCT \text{ coefficients}\}$;
- Average all values at **i** of all denoised patches $\hat{Q}$ containing **i**, weighted by $w_{\tilde{Q}}$.

F. J. Anscombe.
The transformation of Poisson, binomial and negative-binomial data.
Biometrika, 35(3):246–254, 1948.

P. Milanfar.
A tour of modern image filtering.
Invited feature article to IEEE Signal Processing Magazine (preprint at http://users. soe. ucsc. edu/ milanfar/publications/), 2011.

N. N. Ponomarenko, V. V. Lukin, M. S. Zriakhov, A. Kaarna, and J. T. Astola.
An automatic approach to lossy compression of AVIRIS images.
IEEE International Geoscience and Remote Sensing Symposium, 2007.