Multiscale Noise Estimation and Removal for Digital Images

Miguel Colom

Universitat
de les Illes Balears

DOCTORAL THESIS
Defense date: July 7, 2014

MULTISCALE NOISE ESTIMATION AND REMOVAL FOR DIGITAL IMAGES

Miguel Colom Barco

Thesis Supervisors:
Antoni Buades Capó and Jean-Michel Morel

Jury:
Bartomeu Coll Vicens (president),
Josep Lluís Lisani Roca (secretary)
Gabriele Facciolo Furlan (committee member)
Andrés Almansa (committee member)
Enric Meinhardt Llopis (committee member)
Bernard Rougé (invited)

A mi abuela, Baba.
A todos mis amigos en París.

This version of the document was compiled on 2023/06/19 at 18:08:34.

The last updated version of this document can always be downloaded from:
`http://mcolom.info/download/thesis/thesis_miguel_colom.pdf`

Author's contact

Email: `miguel & mcolom.info` (please change the ampersand by the @ sign).

Website: `http://mcolom.info`

Acknowledgements

Starting this thesis at the UIB has been quite a great adventure, but to be honest: also much fun and plenty of special moments with so many good friends I made along this adventure.

In the UIB I have to thank Joan and Cati for their warm welcome and friendship (you'll be the next ones writing your dissertations, after me. Good luck with it!). Also, Jonathan y Thibaud whom were doing a research internship.

I spent some days at CMLA to collaborate with the team at ENS-Cachan, so it was a great pleasure to have met Ariane, Aude, Barbara,, Benjamin, Bruno, Carlo, Claire, Enric, Gabriele, Irène, Ives, José, Julie, Julien, Lara, Laurent, Loïc, Marc, Marie, Martín, Mauricio, Morgan, Nicola, Nicolas C., Nicolas L., Pauline, Rachel, Rafa, Samy, Saad, Tristan, Yohann, and Zhongwei. And also to Nick Chriss, for having improved my English with his classes, and all the secretary team: Véronique Almadovar, Micheline Brunetti, Sandra Doucet, Virginie Pauchont, and Carine Saint-Prix. And definitely, to everybody I've ever met at CMLA.

I have also to thank DxO-Labs, Centre National d'Etudes Spatiales, European Research Council, Office of Naval Research, Spanish Government, Direction Générale de l'Armement, Fondation Mathématique Jacques Hadamard, and Agence Nationale de la Recherche for funding some parts of my research.

I'm thankful to Vladimir V. Lukin, Alessandro Foi, and Benoit Vozel for their useful comments and interaction. It's been really helpful. Also, to Tomeu Coll for kindly looking at the papers I published in IPOL, suggesting changes and noticing mistakes. And to José Luis Lisani, for his thorough inspection of the papers I sent to IPOL and many useful comments, as part of the editorial process before publishing.

For sure, I really thank everybody I met at Colegio de España in Paris (Cité Universitaire) when I arrived to Paris to start a master before starting with the PhD. With not doubt, that's been one of the most fun and enriching experiences in my life! The list of friends I made there is too long to write their names in a single page.

Also, to the friends I made in Paris after I left Cité Universitaire, specially to Ceci, Jaime, Vicente, Cristina and little Luz. And all my friends and family in Palma, who understand my scientific adventures and always give me their support.

To my PhD supervisors Jean-Michel Morel and Toni Buades, for the weekly scientific discussions which gave rise to this thesis, for the projects I've been involved in, for the hours they've spent correcting all the drafts of my articles, and for being always so close. Also, to all the members of the commission that kindly accepted to evaluate this thesis.

And to Gloria, because you make every single moment the best.

Contents

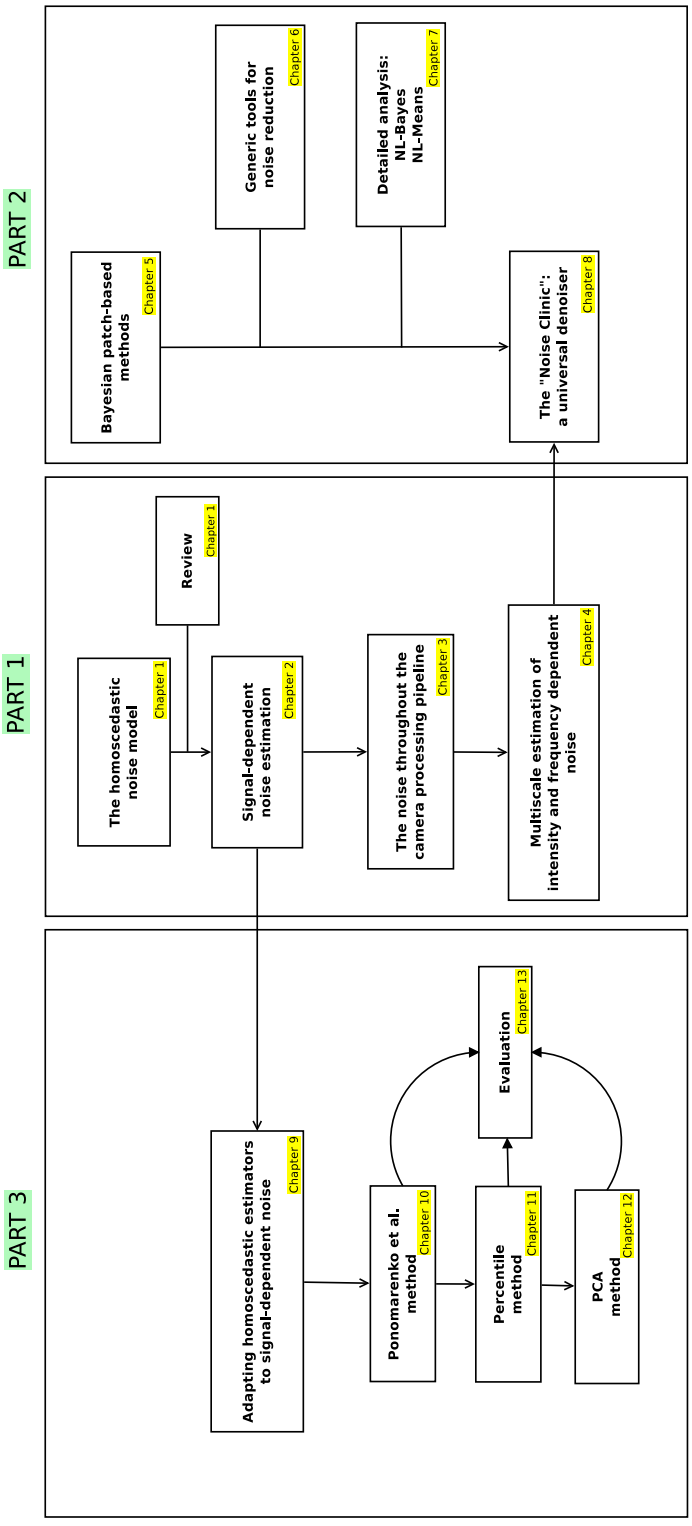


FIGURE 1. Organization of the thesis.

Summary

English

Any image, digital or analogic, contains not only information from the scene being photographed but also external interferences known as *noise*. The resulting image is the combination of the ideal image without noise with noise itself.

The "ideal image without noise" is a mathematic abstraction and it is not available in reality. Thus, it is needed methods that given only the degraded image are capable to properly characterize noise. This characterization using the noisy image is known as *blind noise estimation* since it does not use any additional information out the the noisy image.

Once noise has been properly characterized, the next step is to obtain a version of the image which is as close as possible to the ideal image. This process is known as *blind denoising*, since the ideal image is not available. Denoising methods exploit the property of *autosimilarity* of the small blocks that form the image to infer the geometry of the blocks of the ideal image. Denoising is a process guided by previous noise estimation.

Given that both noise estimation and denoising are performed *blindly*, it is important that noise characterization is as complete as possible. In this thesis several techniques for noise estimation are discussed, from the simplest which just consider homoscedastic noise, through those which consider the Poissonian model, to finally the new technique that we propose to obtain a complex noise model that depends on both intensity and frequency.

Regarding denoising, this thesis is mainly focuses on Bayesian techniques. The thesis finally reaches with the presentation of the *Noise Clinic*, the tool which we propose for automatic noise estimation and denoising. The Noise Clinic combines the automatic estimation of a complex noise model with its elimination at each of the scales of the image. This allows to restore a large typology of images, including those compressed with JPEG.

Català

Qualsevol imatge, ja sigui digital o analògica, conté no només informació de l'escena fotografiada, sinó també interferències externes conegudes com *renou*. La imatge resultant és la combinació de la imatge ideal sense renou, amb el renou mateix.

La "imatge ideal sense renou" és una abstracció matemàtica i no està disponible a la realitat. Per tant, cal utilitzar mètodes que, donada únicament la imatge deteriorada, siguin capaços de caracteritzar adequadament el renou. Aquesta caracterització a partir de la imatge amb renou es

coneix com *estimació a cegues del renou*, atés que no s'utilitza cap altra informació addicional a part de la imatge amb renou.

Un cop caracteritzat adequadament el renou, el següent pas és obtenir una versió de la imatge que sigui tan fidel com sigui possible a la imatge ideal. Aquest procés es coneix com *eliminació de renou a cegues*, ja que la imatge ideal no està disponible. Els mètodes d'eliminació de renou aprofiten la propietat d' *autosimilaritat* dels petits blocs que componen la imatge per inferir la geometria dels blocs de la imatge ideal. L'eliminació de renou és un procés guiat per l'estimació de renou prèvia.

Atès que tant l'estimació com l'eliminació de renou es realitzen *a cegues*, és important que la caracterització del renou sigui tan completa com sigui possible. En aquesta tesi es discuteixen en detall les diverses tècniques per a l'estimació de renou, des de les més simples que únicament consideren renou homoscedàstic, passant per les que consideren el model poissonià de renou, fins a finalment la nova tècnica que proposem per obtenir un model de renou complex, que depèn tant de la intensitat com de la freqüència .

Pel que fa a l'eliminació de renou, aquesta tesi se centra especialment en les tècniques basades en el model bayesià. La tesi culmina amb la presentació de la *Noise Clinic* , l'eina que proposem per a l'estimació i eliminació automàtiques del renou. La Noise Clinic combina l'estimació automàtica d'un model de renou complex amb la seva eliminació en cadascuna de les escales de la imatge. Això permet restaurar una tipologia extensa d'imatges, incloent les comprimides amb JPEG.

Castellano

Cualquier imagen, ya sea digital o analógica, contiene no solamente información de la escena fotografiada, sino también interferencias externas conocidas como *ruido*. La imagen resultante es la combinación de la imagen ideal sin ruido, con el propio ruido.

La "imagen ideal sin ruido" es una abstracción matemática y no está disponible en la realidad. Por lo tanto, es necesario utilizar métodos que, dada únicamente la imagen deteriorada, sean capaces de caracterizar adecuadamente el ruido. Esta caracterización a partir de la imagen ruidosa se conoce como *estimación a ciegas del ruido*, ya que no se utiliza ninguna otra información adicional aparte de la imagen ruidosa.

Una vez caracterizado adecuadamente el ruido, el siguiente paso es obtener una versión de la imagen que sea tan fiel como sea posible a la imagen ideal. Este proceso se conoce como *eliminación de ruido a ciegas*, ya que la imagen ideal no está disponible. Los métodos de eliminación de ruido aprovechan la propiedad de *autosimilaridad* de los pequeños bloques que componen la imagen para inferir la geometría de los bloques de la imagen ideal. La eliminación de ruido es un proceso guiado por la estimación de ruido previa.

Dado que tanto la estimación como la eliminación de ruido se realizan *a ciegas*, es importante que la caracterización del ruido sea tan completa como sea posible. En esta tesis se discuten en detalle las diversas técnicas para la estimación de ruido, desde las más simples que únicamente

consideran ruido homoscedástico, pasando por las que consideran el modelo poissoniano de ruido, hasta finalmente la nueva técnica que proponemos para obtener un modelo de ruido complejo, que depende tanto de la intensidad como de la frecuencia.

En cuanto a la eliminación de ruido, esta tesis se centra especialmente en las técnicas basadas en el modelo bayesiano. La tesis culmina con la presentación de la *Noise Clinic*, la herramienta que proponemos para la estimación y eliminación automáticas del ruido. La Noise Clinic combina la estimación automática de un modelo de ruido complejo con su eliminación en cada una de las escalas de la imagen. Esto permite restaurar una tipología extensa de imágenes, incluyendo las comprimidas con JPEG.

Preface

Digital images are matrices of regularly spaced pixels, each containing a photon count. This photon count is a stochastic process due to the physical quantum nature of light. It follows that all images are noisy. Ever since digital images exist, numerical methods have been proposed to improve the signal to noise ratio. Such "denoising" methods require a noise model and an image model. This thesis addresses the definition of noise models and their estimation from the digital image themselves. It also develops the main application which we call "blind denoising", namely the fully automatic noise detection and removal. This is done in the framework of state of the art denoising algorithms which are mostly patch-based. For this reason, the thesis also presents a synthetic theory of patch-based methods.

This thesis is divided into three parts:

(1) Noise Estimation
(2) Patch Denoising
(3) Reproducible Research Contributions

Part 1: NOISE ESTIMATION

In the first part of the thesis, Noise Estimation, we discuss several strategies to estimate the noise. The simplest experimental procedure to evaluate a noise estimation strategy just consists on simulating white Gaussian noise and adding it to a noise-free image (or to an image which is supposed to contain a very small or negligible noise). Then, the noise is estimated with some *homoscedastic*[1] noise estimator and its variance is obtained (in this context, "homoscedastic" means that the variance of the noise does not vary depending on the intensity or the frequencies in the noisy image). In Chapter 1 several homoscedastic noise methods are presented and discussed. This work by Lebrun, Colom, Buades, and Morel was published in the Acta Numerica journal with the article *Secrets of image denoising cuisine* [**1**].

Unfortunately, the simple homoscedastic noise model is not useful to estimate the noise in real digital noise images. Indeed, the very first image acquired by the camera at the focal plane (the *raw* image) exhibits a noise that depends on the intensity. This noise, called the *photon noise* can be modeled with a Poisson distribution, for which the variance is an increasing function of the expectation (the mean intensity), also called a *noise curve*. This is related to the physical quantum

[1] Here *homoscedastic* noise refers to a set of random variables with the same finite variance regardless their mean.

nature of light, for which the emission of individual photons by any body is a Poisson random process. Therefore, even in the raw image the noise is signal-dependent: the amount of noise increases with the intensity of the underlying ideal image. This invalidates for real applications the use of classic methods (discussed in Chapter 1) that only estimate a global variance of the noise for the whole image. However, most of the noise estimation methods are patch-based and in consequence they can be easily adapted to estimate signal-dependent noise. Although the exact distribution for photon noise is the Poisson distribution, when the exposure time λ is large enough ($\lambda > 1000$), the Poisson distribution of parameter λ can be approximated with small error by a Gaussian distribution with $\mu = \sigma^2 = \lambda$. Therefore, many signal-dependent noise estimation methods assume that the noise is white and Gaussian for each intensity level [**2, 3, 4, 5, 6**]. However, the assumption that states that the variance is linear with the intensity is false in general, since the saturation of the detectors at the most dark and bright pixels of the image gives a nonlinear function of the variance according to the intensity. Even if the noise function under saturation can be predicted quite accurately, as shown by Foi in the article *Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-Image Raw-Data* [**7**], the noise does not need to follow at all the linear model and might follow any other model, as demonstrated by Boie and Cox in *An Analysis of Camera Noise* [**8**]. In general, the information about the noise model that corrupted an image, the characteristics of the detector and the exact transformations it suffered until the final image was formed, are unknown. Therefore, in that situation the only solution is to obtain a noise model *directly from the noisy image*, without assuming any prefixed model or parameters (*nonparametric* estimation). In Chapter 2 we present to procedure to adapt any patch-based homoscedastic noise estimator to obtain a signal-dependent noise estimation. In continuation we give a nonparametric method that overcomes the state of the art in signal-dependent noise estimation for raw images. An extensive cross-validation procedure is described to compare this new method with state-of-the-art parametric methods and with laboratory calibration methods giving a reliable ground-truth, even for nonlinear detectors. The procedure to obtain a ground-truth is described in detail. This work was published by Colom, Buades, and Morel in the Journal of the Optical Society of America A, with the article *Nonparametric Noise Estimation Method for Raw Images* [**9**].

In Chapter 3 we study in detail the characteristics of the noise through each step of the camera processing chain, namely:

(1) The raw image acquisition at the focal plane of the camera.
(2) Demosaicing, the obtain a color image.
(3) White balance, the compensate the different gains of the detector at each channel in order to get realistic colors.
(4) Gamma correction, to increase the dynamics of the image and therefore enhance the visualization of dark pixels.
(5) JPEG compression, to reduce the size of the file that will finally contain the data of the image by lossy compression.

The aim is to understand how each of these transformations affects the noise curve obtained with a signal-dependent noise estimator. We used two different cameras (Canon EOS 30D and

FIGURE 2. Typical color spots or stains that can be observed in a JPEG image, caused mainly by two different steps of the camera processing. First, demosaicing correlates noise (thus creating low frequency noise that looks like color spots) and afterwards the gamma correction step increases the energy of the pixels, specially the darkest. As a result, the noise is converted in color spots which are clearly visible all over the image, specially at the darkest zones.

Nikon D80), two ISO speeds (1250 and 1600), and four exposure times ($1/30s$, $1/250s$, $1/400s$, $1/640s$). We identify and discuss the sources of the global perturbation that we observe as "noise" (dark noise, photon noise, readout noise, shot noise, and electronic noise), explain each of the steps in the camera processing pipeline, and for each of the step, discuss the obtained noise curves. Chapter 3 explains the origin of the kind of noise that is observed at the final JPEG image: small colors spots, especially at the darkest zones. Figure 2 shows the typical color spots or stains that can be observed in a JPEG image, caused mainly by two different steps of the camera processing. First, demosaicing correlates noise (thus creating low frequency noise that looks like color spots) and afterwards the gamma correction step increases the energy of the pixels, specially at the darkest. As a result, the noise is converted in color spots which are clearly visible all over the image, specially at the darkest zones.

Under the same ISO speed and exposure time conditions, the noise curves obtained by different cameras differ. Some cameras do not pre-process at all the data acquired at the CCD or CMOS detector and therefore it would be possible to assume a Poisson model for the noise. However, in other cameras the data at the raw image has been already altered in an unknown way, thus making it impossible to assume any model. Therefore, it is preferable to use non-parametric models that directly estimate a noise curve from the image itself with assuming a predefined model.

Figure 3 shows the noise curves obtained with a signal-dependent noise estimation along all the processing chain: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression, using a Canon EOS 30D camera. In solid lines, the temporal estimation (ground-truth) and in dashed lines the spatial estimation.

In fact, if the noise estimator assumes that the noise is signal-dependent but does not take into account that the noise depends on the frequency, the noise is strongly subestimated, as can be

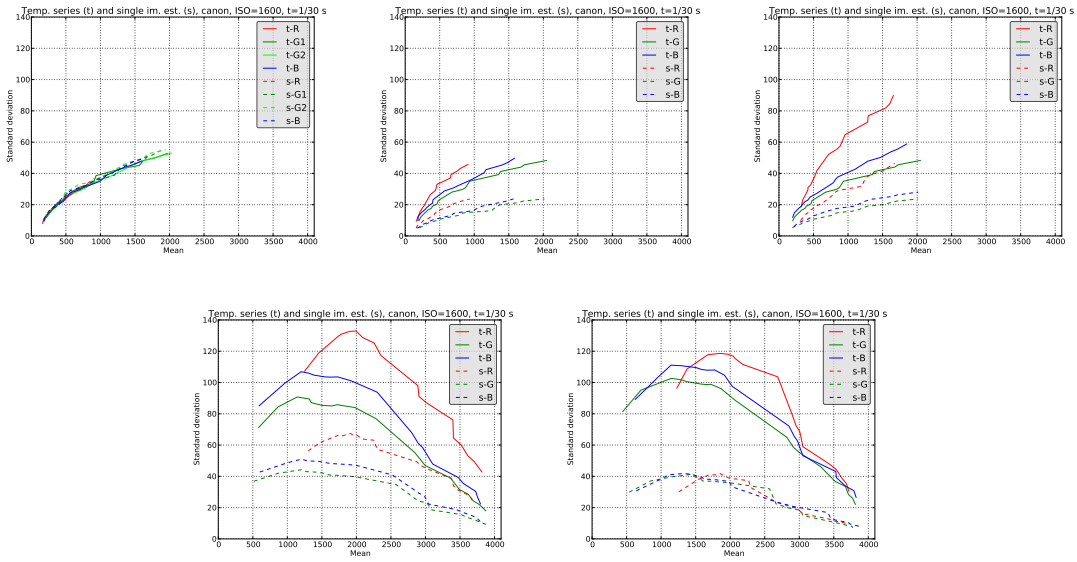


FIGURE 3. Complete pipeline for ISO 1600, t=1/30s, Canon: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression. In the first step (raw image), all four color channels share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts to decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain. In solid lines, the temporal estimation (ground-truth) and in dashed lines the spatial estimation.

observed in Figure 3, where the spatial estimation given by the algorithm (dashed lines) is really underestimating the noise (solid lines, the ground truth from the temporal estimation). In general, denoising algorithms need an accurate estimation of the noise to properly denoise an image. Since the exact transformations that have been applied to the noisy image are unknown, assuming a model is too risky and it is preferable to get a profile of the noise depending of the intensity, frequency and scale from the noisy image itself. Chapter 3 concludes that any noise estimation algorithm that is intended for real images must consider the noise not only signal-dependent, but also frequency-dependent (SFD noise). The noise must be estimated according both the intensity and the frequency.

Chapter 4 presents our proposed method to estimate SFD noise. We present a new semi-distance to measure the likeliness between two patches, called the *sparse distance* and depict a new algorithm that overcomes the current state of the art algorithm for frequency-dependent noise. The algorithm is validated using both simulations and observations of denoising results on real images. This work by Colom, Lebrun, Buades, and Morel is submitted to the IEEE Transactions On Image Processing with the article *Multiscale Estimation of Intensity and Frequency Dependent Noise* [**10**].

Part 2: PATCH DENOISING

The second part of the thesis, Patch Denoising, discusses the obvious application of noise estimation: using the noise model obtained from the noisy image itself to remove the noise from it and obtain a new version of the image for which the noise has been removed (or at least, minimized). Of course, details, textures, and edges must be preserved. This process is called *denoising*.

Chapter 5 discusses the Bayesian patch-based method, which gives an optimal formulation under the assumption that the patches similar to a given image patch follow a stochastic model. Given a noiseless patch P of u with dimension $\kappa \times \kappa$, and $\tilde{P}$ an observed noisy version of P, the model gives by assuming a Gaussian model and the independence of noise pixel values

$$\mathbb{P}(\tilde{P}|P) = c \cdot \exp\left(\frac{-\|\tilde{P} - P\|^2}{2\sigma^2}\right),$$

where P and $\tilde{P}$ are considered as vectors with κ^2 components, $||P||$ denotes the Euclidean norm of P, σ^2 the variance of the Gaussian noise, and c is the normalizing constant. Knowing $\tilde{P}$, the goal is to deduce P by maximizing $\mathbb{P}(P|\tilde{P})$. Using Bayes' rule, the last conditional probability can be written as

$$\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}.$$

$\tilde{P}$ being observed, this formula can in principle be used to deduce the patch P maximizing the right term, viewed as a function of P. If we assume that patches Q similar to a given patch P also follow the Gaussian model with empirical sample covariance matrix $\mathbf{C}_P$ and empirical sample mean $\overline{P}$, then

$$\mathbb{P}(Q) = c \cdot \exp\left(\frac{-(Q - \overline{P})^t \mathbf{C}_P^{-1}(Q - \overline{P})}{2}\right).$$

Using the classical Bayesian equivalence of problems,

$$\max_P \ \mathbb{P}(P|\tilde{P}) \Leftrightarrow \max_P \ \mathbb{P}(\tilde{P}|P)\mathbb{P}(P)$$

and after some calculus, Chapter 5 concludes that a restored patch $\hat{P}_1$ can be obtained from the observed patch $\tilde{P}$ by the one step estimation

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

Chapter 6 discusses and gives also the detailed algorithmic descriptions of three generic tools used to denoise an image:

- Aggregation
- "Oracle"
- Color transform

The idea behind aggregation techniques is to combine for any pixel a set of m possible estimates. If these estimates were independent and had equal variance, then a uniform average would reduce this estimator variance by a factor m. For most denoising methods the variance of the

estimators is high near image edges. When applied without aggregation, the denoising methods leave visible "halos" of residual noise near edges (for example, in the sliding window DCT method).

Aggregation techniques aim at a superior noise reduction by increasing the number of values being averaged for obtaining the final estimate or selecting those estimates with lower variance. Another type of aggregation technique, like in the works of Raphan, Van De Ville, and Deledalle, considers the risk estimate rather than the variance to locally attribute more weight to the estimators with small risk [**11, 12, 13**].

The use of an "oracle" is another technique used to improve the denoising results. Iterative strategies to remove residual noise would drift from the initial image. Instead, a first step denoised image can be used to improve the reapplication of the denoising method to the initial noisy image. In a second step application of a denoising principle, the denoised DCT coefficients, or the patch distances, can be computed in the first step denoised image. They are an approximation to the true measurements that would be obtained from the noise-free image. Thus, the first step denoised image is used as an "oracle" for the second step.

A color transform is also a useful tool to avoid artifacts introduced by denoising. Indeed, most denoising algorithms treat each image channel independently, which may introduce color artifacts easily noticeable by the eye. Denoising the image in a different color space avoids this problem. For example, passing from the RGB to the YUV colorspace. Finally, Chapter 6 shows an example (Figure 1) where it can be clearly identified the effect of each of these tools when performing denoising.

Chapter 7 presents and discusses in detail two patch-based denoising methods, the classic NL-means method which uses similar patches to denoise by aggregation and the NL-Bayes method (Chapter 7), which uses an oracle step to obtain a list of similar patches for each patch it denoises. A cleaned version of the patch is obtained from the improved sample covariance matrix of the similar patches thanks to the oracle. Detailed algorithmic descriptions for both methods are given in Algorithm 18 and Algorithm 19, namely.

Chapter 8 develops the project which we named *Noise Clinic*, that involves applying the multiscale intensity-frequency estimation method in Chapter 4 to obtain a noise model from the patches of the noisy image itself (nonparametric estimation) to denoise it without any prior information (blind denoising) using a Bayesian patch-based method. The method uses a *multiscale* strategy that allows to estimate accurately strongly correlated low-frequency noise, as long as the image is large enough.

Indeed, in most images handled by the public and even by scientists, the noise model is imperfectly known or unknown. End users only dispose of the result of a complex image processing chain effectuated by uncontrolled hardware and software (and sometimes by chemical means). For such images, recent progress in noise estimation permits to estimate from a single image a noise model which is simultaneously signal and frequency dependent. Chapter 8 proposes a multiscale denoising algorithm adapted to this broad noise model. This leads to a blind denoising algorithm which we demonstrate on real JPEG images and on scans of old photographs for which

the formation model is unknown. The consistency of this algorithm is also verified on simulated distorted images. This algorithm is finally compared to the unique state of the art previous blind denoising method [**14**] by Portilla.

Figure 4 shows a 7 February 1966 picture of the Honolulu Conference, with General Earle Wheeler, Secretary of State Dean Rusk, and President Lyndon B. Johnson depicted in Camp Smith, Hawaii[2] (a), a detail in the noisy image (b), and the same detail in the denoised image (c). It can be observed that the noise has been removed, while keeping the details of the image. Figure 5 shows the noise curves obtained when denoising the image using two scales with the Noise Clinic. On the left, the mean of the standard deviations of the high (a) and low (b) frequencies at the first scale. On the right, the mean of the standard deviations of the high (b) and low (d) frequencies at the second scale. It can be observed that the noise at the first scale is lower than the noise at the second scale, and that within any of the scales the noise at the low frequencies is higher than the noise at the high frequencies. This is because the noise has mainly energy in its low frequencies and is highly correlated. Figure 6 shows a detail of difference between the noisy image and the image denoised at each scale. In scale #1 (b), the noise has a particular spatial structure, since low-frequency noise is detected at the second scale. On scale #0 (a), the noise is less correlated, but still mainly low-frequency noise. To improve the visualization of the images, the histogram of the difference image has been equalized.

Part 3: REPRODUCIBLE RESEARCH CONTRIBUTIONS

The third part of the thesis presents three of the reproducible research contributions of this dissertation:

- how to adapt block-based homoscedastic noise estimators to measure intensity-dependent noise;
- how to filter the obtained noise curves to cancel undue oscillations;
- how saturated pixels distort the shape of noise curve and how to avoid them.

The articles presented in this third part were published in the Image Processing On Line (IPOL) journal. It publishes image processing and image analysis algorithms, described in accurate literary form, coupled with code. This way, scientists are allowed to check directly the published algorithms online with any uploaded image. It also promotes reproducible research, and the establishment of a state of the art verifiable by all, and on any image.

Section 1.1 of Chapter 9 discusses how to adapt most of the patch-based noise estimation methods to measure intensity-dependent noise. For a signal-dependent noise, a "noise curve" must be established. This noise curve associates with each image value $\mathbf{U}(x, y)$ an estimation of the standard deviation of the noise associated with this value. Thus, for each block in the image, its mean must be computed and will give an estimation of a value in $\mathbf{U}$. The measurement of the

[2]From the National Archives and Records Administration. The National Archives and Records Administration provides images depicting American and global history which are public domain or licensed under a free license.

(a)

(b)

(c)

FIGURE 4. A 7 February 1966 picture of the Honolulu Conference, with General Earle Wheeler, Secretary of State Dean Rusk, and President Lyndon B. Johnson depicted in Camp Smith, Hawaii (a), a detail in the noisy image (b), and the same detail in the denoised image (c). It can be observed that the noise has been removed, while keeping the details of the image.

variation of the block (for example, its variance) will also be stored. The means are classified into a disjoint union of variable intervals or bins, in such a way that each interval contains a large enough number of elements. These measurements allow for the construction of a list of block variances whose corresponding means belong to the given bin. The procedure to obtain the noise curve is discussed and the description is given in Algorithm 21.

The noise curve obtained may present peaks when some given gray level interval contains mostly means of blocks belonging to a highly textured region. In this case, the measured block variance would be caused by the signal itself and not by the noise and the noise variance would

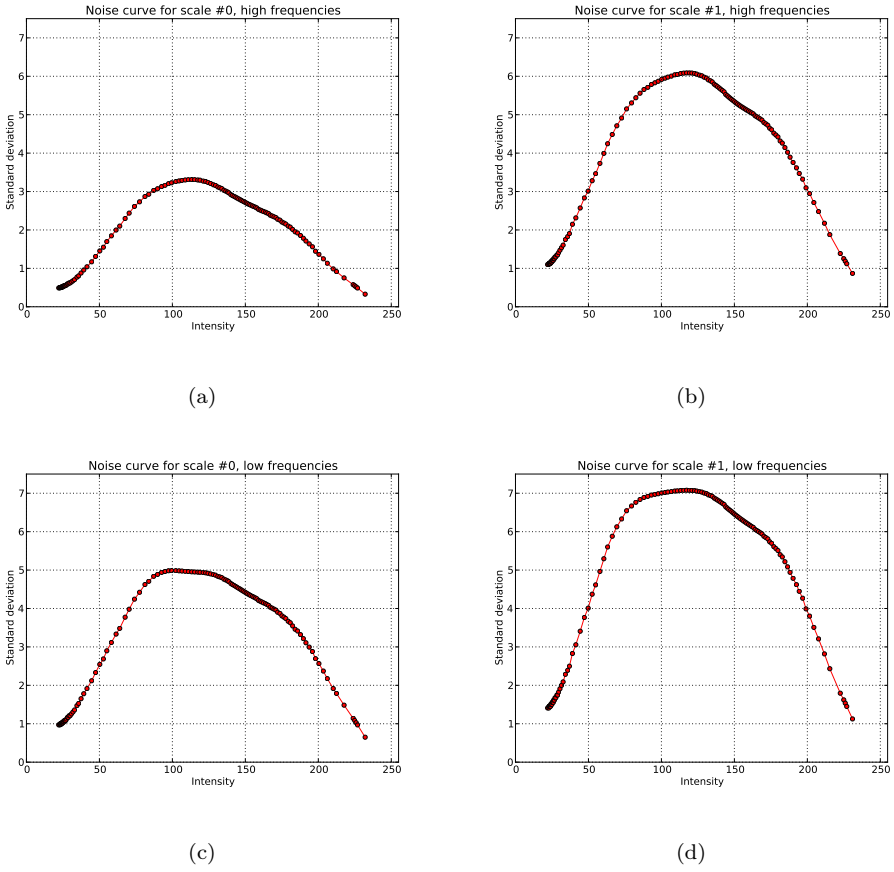


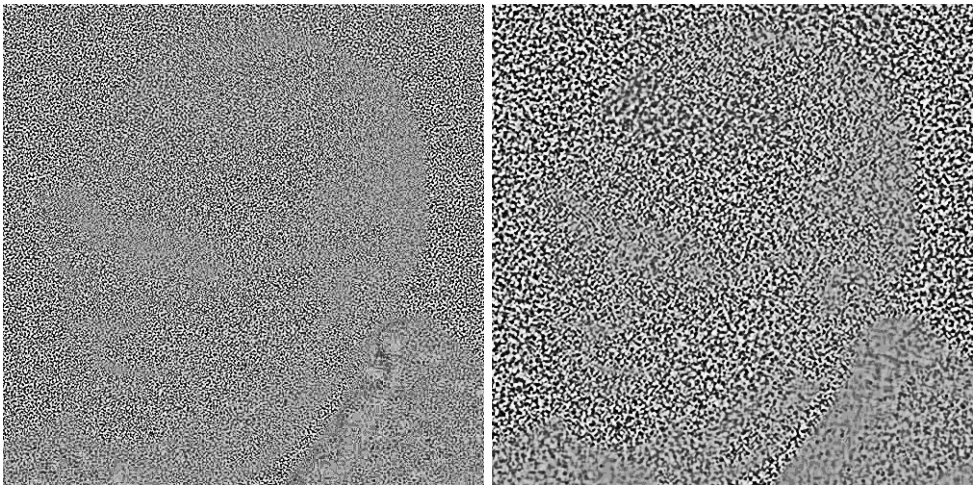
FIGURE 5. Noise curves obtained when denoising the image with two scales. On the left, the mean of the standard deviations of the high (a) and low (b) frequencies at the first scale. On the right, the mean of the standard deviations of the high (b) and low (d) frequencies at the second scale.

be overestimated. To solve it, the noise curve obtained can be filtered. The pseudocode of the filtering is detailed in Algorithm 24.

When the number of photons measured by the CCD or CMOS detector during the exposure time is too high, its output may get saturated, and therefore underestimated. When the signal saturates the output of the CCD or CMOS detector, the measured variance in the saturated areas of the image is zero. If saturated pixels are taken into account when measuring the noise, the noise curve is no more reliable. Section 1.3 of Chapter 9 presents the strategy proposed to discard saturated pixels, that consists on rejecting the blocks that contain a group of four connected exactly equal pixels, in any of the channels. The pseudocode can be found in Algorithm 25.

These three general tools (adaptation to signal-dependent noise, noise curve filtering, and avoiding saturated pixels in the estimation) were applied to the

- Ponomarenko et al. method;



(a) (b)

FIGURE 6. Detail of difference between the noisy image and the image denoised at each scale. In scale #1 (b), the noise has a particular spatial structure, since low-frequency noise is detected at the second scale. On scale #0 (a), the noise is less correlated, but still mainly low-frequency noise. To improve the visualization of the images, the histogram of the difference image has been equalized.

- Percentile method;
- PCA method.

Chapter 10 discusses and analyzes in deep detail the Ponomarenko et al. method inside this new framework. In the article *An Automatic Approach to Lossy Compression of AVIRIS Images* **[15, 16]** N. N. Ponomarenko, V. V. Lukin, M. S. Zriakhov, A. Kaarna, and J. T. Astola propose a new method to specifically compress AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) images. As part of the compression algorithm, a noise estimation is performed with a proposed new algorithm based on the computation of the variance of overlapping 8×8 blocks. The noise is estimated on the high-frequency orthonormal DCT-II coefficients of the blocks. To avoid the effect of edges and textures, the blocks are sorted according to their energy measured on a set of low-frequency coefficients. The final noise estimation is obtained by computing the median of the variances measured on the high-frequency part of the spectrum of the blocks using only those whose energy (measured on the low-frequencies) is low. A small percentile of the total set of blocks (typically the 0.5%) is used to select those blocks with the lower energy at the low-frequencies.

Chapter 11 discusses and analyzes in depth the Percentile method **[17]** using the presented framework. Given a white Gaussian noise signal $\mathbf{N}_\sigma$ on a sampling grid, its variance σ^2 can be estimated from a small $w \times w$ pixels sample. However, in natural images we observe $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{N}_\sigma$, the combination of the geometry of the scene that is photographed and the added noise. In this case, estimating directly the standard deviation of the noise from $w \times w$ samples of $\tilde{\mathbf{U}}$ is not reliable since the measured standard deviation is not explained just by the noise but also from

the geometry of $\mathbf{U}$. The Percentile method tries to estimate the standard deviation σ from $w \times w$ blocks of a high-passed version of $\tilde{\mathbf{U}}$ by a small p-percentile of these standard deviations. The idea behind is that edges and textures in a block of the image increase the observed standard deviation but they never make it decrease. Therefore, a small percentile (0.5%, for example) in the list of standard deviations of the blocks is less likely to be affected by the edges and textures than a higher percentile (50%, for example). The 0.5%-percentile is empirically proven to be adequate for most natural, medical and microscopy images.

Chapter 12 discusses and analyzes in deep detail the PCA method from the article *Image Noise Level Estimation by Principal Component Analysis* **[18, 3]**, where S. Pyatykh, J. Hesser, and L. Zheng propose a new method to estimate the variance of the noise in an image from the eigenvalues of the covariance matrix of the overlapping blocks of the noisy image. Instead of using all patches of the noisy image, the authors propose an iterative strategy to adaptively chose the optimal set containing the patches with lowest variance. The method is analyzed inside the presented framework.

For these three methods, the following tests were performed in Chapter 13 for the Ponomarenko et al., Percentile and PCA methods:

- Tests on simulated white Gaussian noise using the noise-free images. In this case we took seven bins to classify the blocks according to their means (see Section 1.1 of Chapter 9).
- Tests on a set of real raw images obtained by a Canon EOS 30D camera (see Figure 2). The procedure explained in Section 1.1 of Chapter 9 was used to get a noise curve. The results were compared to the ground-truth noise curve of the camera.
- Test on multiscale coherence. The standard deviation of a Gaussian white noise is divided by two when the image is down-scaled. By *down-scaling* the image we mean a sub-sampling of the image where each block of four pixels is substituted by their mean. This test checks if the measured noise is divided by two at each image down-scaling.

We show the results obtained with the set of images for the Ponomarenko et al., Percentile and PCA methods.

Of course, estimating homoscedastic noise is not enough to characterize the signal-dependent noise in real digital images. In order to evaluate the accuracy of the three methods with signal-dependent noise, we obtained a ground-truth noise curve of the Canon EOS 30D camera with the procedure explain in Chapter 2. Then, we obtained the noise curves for each of the raw images in 2 and compared them with the ground-truth noise curve of the camera. Figure 7 shows the ground-truth noise curve for raw images for the Canon EOS 30D camera, with ISO speed 1600.

To evaluate the accuracy of each method, we compared the obtained noise curve obtained by each method with the ground-truth.

After evaluating all three methods with the three tests (simulated homoscedastic noise, intensity-dependent noise, and multiscale coherence), the conclusion is that the strategy followed by the modified Ponomarenko et al. method is the best and can be considered as the state of the art in intensity-dependent noise estimation. The Percentile method gives slightly worse results than

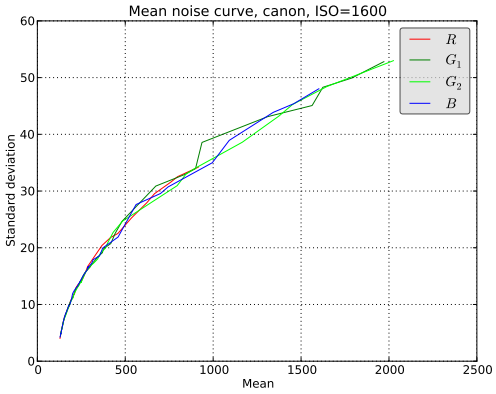


FIGURE 7. Ground-truth noise curve for the Canon EOS 30D camera, for ISO speed 1600.

Ponomarenko et al. method and PCA is in general worse and requires at least the double of samples than Ponomarenko and Percentile to achieve similar results.

Section 5 of Chapter 13 discusses about the algorithmic complexity the operations needed to adapt homoscedastic noise estimation methods into intensity-dependent, the noise filtering, and the detection of blocks with saturated pixels. These operations can be considered a common framework where almost all noise estimation algorithms can be put inside and estimate intensity-dependent noise. The algorithmic complexity analysis of the Ponomarenko et al., Percentile, and PCA methods within the common framework is also given.

Chapter 6 presents the conclusions of the research presented in this thesis.

Bibliography of the thesis

The following research articles were published or submitted during this thesis:

Published

M. Colom, A. Buades, and J.M. Morel, "Nonparametric noise estimation method for raw images", *Journal of the Optical Society of America A*, vol. 31, 2014. doi: 10.1364/JOSAA.31.000863

M. Colom and A. Buades, "Analysis and Extension of the Ponomarenko et al. Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, vol. 3, pp. 173–197, 2013. doi: 10.5201/ipol.2013.45

M. Colom and A. Buades, "Analysis and Extension of the Percentile Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, vol. 2013, pp. 322–349, 2013. doi: 10.5201/ipol.2013.90

M. Lebrun, M. Colom, A. Buades, and J.M. Morel, "Secrets of image denoising cuisine", *Acta Numerica*, vol. 21, pp. 475–576, 2012, doi: 10.1017/S0962492912000062

To appear

M. Colom, G. Facciolo, M. Lebrun, N. Pierazzo, M. Rais, Y. Wang, and J.M. Morel, "A mathematical perspective of image denoising", *International Congress of Mathematicians*, 2014, To appear

Submitted

M. Colom, M. Lebrun, A. Buades, and J.M. Morel, "A non-parametric approach for the estimation of intensity-frequency dependent noise", *IEEE International Conference on Image Processing*, 2014, Submitted

M. Colom, M. Lebrun, A. Buades, and J.M. Morel, "Multiscale estimation of intensity and frequency dependent noise", *IEEE Transactions On Image Processing*, 2014, Submitted

M. Colom and A. Buades, "Analysis and Extension of the PCA Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, 2014, Submitted

M. Lebrun, M. Colom, and J.M. Morel, "The Noise Clinic: a universal blind denoising algorithm", *IEEE International Conference on Image Processing*, 2014, Submitted

M. Lebrun, M. Colom, and J.M. Morel, "Multiscale image blind denoising", *IEEE Transactions On Image Processing*, 2014, Submitted

Part 1

NOISE ESTIMATION

In this first part of the thesis, we address the problem of the different noise models and how adequate they are to estimate noise in different scenarios. From the simplest models for white Gaussian noise to the most complex noise model which are able to estimate noise in JPEG-compressed images. All the noise estimation models described in this thesis (and thus, all denoising methods), are block-based.

Chapter 1 discusses the homoscedastic white Gaussian noise model, the simplest model. It is not directly applicable to real images, where the physics of light make the variance of the noise depend on the intensity. However, it is possible to adapt the noise estimator under this model to measure signal-dependent noise, as explained in Chapter 2 and hence the interest of considering this model. In this chapter we evaluate several homoscedastic noise estimation methods and also introduce the objective of noise estimation which is, obviously, denoising. The classic NL-means denoiser is shown as an example, and three generic tools for denoising are discussed.

In Chapter 2 we discuss the adaptation of block-based noise estimators to measure signal-dependent noise, a model that is consistent with the photon noise (modeled with the Poisson distribution). The concept of "noise curve" is introduced and a non-parametric estimating the noise curve directly from a single raw image is described. We also detail the procedure that permits to obtain a ground-truth noise curve.

In order to check the validity of the signal-dependent noise model, Chapter 3 shows the noise curves at each step of the camera processing pipeline (from the initial raw to the final JPEG image) and compares them with the corresponding ground-truth curve. The signal-dependent noise model is valid for raw images, but is not enough for correlated noise, as shown in Chapter 3 This noise curves along the step of the camera pipeline show that, indeed, the signal-dependent model is not sufficient after demosaicing.

Chapter 4 discusses a new estimation method able to measure the noise even in JPEG-encoded images, where the noise model is complex, with low-frequency noise, and highly correlated.

CHAPTER 1

The homoscedastic noise model

In this chapter, our main goal is to review not less than 13 blind homoscedastic white noise estimation methods using a single image. These methods will be discussed within the context of block-based denoising. Indeed, the main idea behind denoising techniques is to exploit the self-similarity property of natural images, to find similar patches whose aggregation gives a denoised version of them. The measure of the distance between patches depends on the variance of the noise and therefore noise estimation is a needed step for blind denoising. We will show this using a classic denoiser (NL-means) as an example. The performance of all methods depends on three generic tools: color transform, aggregation, and an "oracle" step, which will be presented in this chapter.

In this chapter we will only focus on homoscedastic noise estimation, that is, assuming that the noise variance is fixed and does not depend on the intensity or the frequency of the underlying noise-free image. Since adapting block-based noise estimators to signal and frequency dependent noise is relatively easy, it is justified to analyze first homoscedastic noise estimators. Later in Chapter 2 we discuss the adaptation of block-based noise estimator to signal-dependent noise and in Chapter 4 the adaptation to signal and frequency dependent noise.

1. Introduction

Most digital images and movies are currently obtained by a CCD or CMOS detector. The value $\tilde{u}(\mathbf{i})$ observed by a sensor at each pixel $\mathbf{i}$ is a Poisson random variable whose mean $u(\mathbf{i})$ would be the ideal image. The difference between the observed image and the ideal image $\tilde{u}(\mathbf{i}) - u(\mathbf{i}) = n(\mathbf{i})$ is called "shot noise". On a motionless scene with constant lighting, $u(\mathbf{i})$ can be approached by simply accumulating photons for a long exposure time, and by taking the temporal average of this photon count, as illustrated in Figure 1.

Accumulating photon impacts on a surface is therefore the essence of photography. The first Nicéphore Niépce photograph in 1826 **[23]** was obtained after an eight hours exposure. The problem of a long exposure is the variation of the scene due to changes in light, camera motion, and incidental motions of parts of the scene. The more these variations can be compensated, the longer the exposure can be, and the more the noise can be reduced. If a camera is set to a long exposure time, the photograph risks motion blur. If it is taken with short exposure, the image is dark, and enhancing it reveals the noise.

A possible solution is to take a burst of images, each with short-exposure time, and to average them after registration. This technique, illustrated in Figure 1, was evaluated recently in a paper

that proposes fusing bursts of images taken by cameras [**24**]. This paper shows that the noise reduction by this method is almost perfect: fusing m images reduces the noise by a $\sqrt{m}$ factor.

It is not always possible to accumulate photons. There are obstacles to this accumulation in astronomy, biological imaging and medical imaging. In day to day images, the scene is moving, which limits the exposure time. The main limitations to any imaging system are therefore the noise and the blur. In this review, experiments will be conducted on photographs of scenes taken by normal cameras. Nevertheless, the image denoising problem is a common denominator of all imaging systems.

A naive view of the denoising problem would be: how to estimate the ideal image, namely the mean $u(\mathbf{i})$, given only one sample $\tilde{u}(\mathbf{i})$ of the Poisson variable? The best estimate of this mean is of course this unique sample $\tilde{u}(\mathbf{i})$. Getting back a better estimate of $u(\mathbf{i})$ by observing only $\tilde{u}(\mathbf{i})$ is impossible. Getting a better estimate by using also the rest of the image is obviously an ill-posed problem. Indeed, each pixel receives photons coming from different sources.

Nevertheless, a glimpse of a solution comes from image formation theory. A well-sampled image u is band-limited [**25**]. Thus, it seems possible to restore the band-limited image u from its degraded samples $\tilde{u}$, as was proposed in 1966 in [**26**]. This classic Wiener-Fourier method consists in multiplying the Fourier transform by optimal coefficients to attenuate the noise. It results in a convolution of the image with a low-pass kernel.

From a stochastic viewpoint, the band-limitedness of u also implies that values $\tilde{u}(\mathbf{j})$ at neighboring pixels $\mathbf{j}$ of a pixel $\mathbf{i}$ are positively correlated with $\tilde{u}(\mathbf{i})$. Thus, these values can be taken into account to obtain a better estimate of $u(\mathbf{i})$. These values being nondeterministic, Bayesian approaches are relevant and have been proposed as early as 1972 in [**27**].

In short, there are two complementary early approaches to denoising, the Fourier method, and the Bayesian estimation. The Fourier method has been extended in the past thirty years to other linear space-frequency transforms such as the windowed DCT [**28**] or the many wavelet transforms [**29**].

Being first parametric and limited to rather restrictive Markov random field models [**30**], the Bayesian method are becoming non-parametric. The idea for the recent non parametric Markovian estimation methods is a now famous algorithm to synthesize textures from examples [**31**]. The underlying Markovian assumption is that, in a textured image, the stochastic model for a given pixel $\mathbf{i}$ can be predicted from a local image neighborhood P of $\mathbf{i}$, which we shall call "patch".

The assumption for recreating new textures from samples is that there are enough pixels $\mathbf{j}$ similar to $\mathbf{i}$ in a texture image $\tilde{u}$ to recreate a new but similar texture u. The construction of u is done by nonparametric sampling, amounting to an iterative copy-paste process. Let us assume that we already know the values of u on a patch P surrounding partially an unknown pixel $\mathbf{i}$. The Efros-Leung [**31**] algorithm looks for the patches $\tilde{P}$ in $\tilde{u}$ with the same shape as P and resembling P. Then a value $u(\mathbf{i})$ is sorted among the values predicted by $\tilde{u}$ at the pixels resembling $\mathbf{j}$. Indeed, these values form a histogram approximating the law of $u(\mathbf{i})$. This algorithm goes back

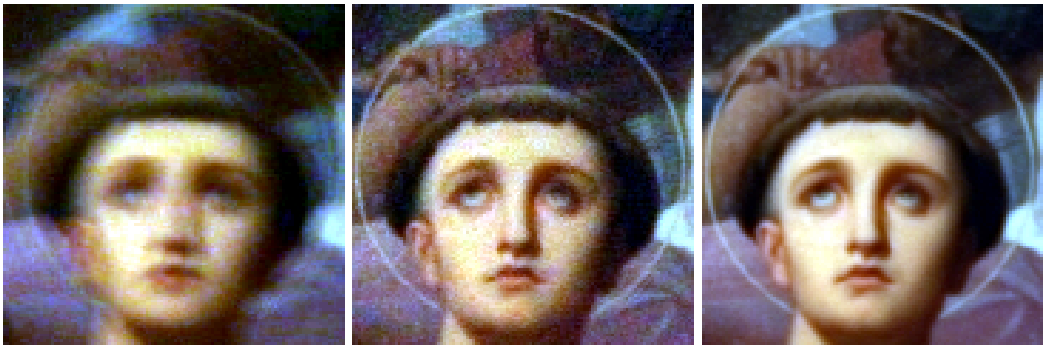


FIGURE 1. From left to right: (a) one long-exposure image (time=0.4 s, ISO=100), one of 16 short-exposure images (time=1/40 s, ISO=1600) and their average after registration. The long exposure image is blurry due to camera motion. (b) The middle short-exposure image is noisy. (c) The third image is about **four times** less noisy, being the result of averaging 16 short-exposure images. From **[24]**.

to Shannon's theory of communication **[25]**, where it was used for the first time to synthesize a probabilistically correct text from a sample.

As was proposed in **[32]**, an adaptation of the above synthesis principle yields an image denoising algorithm. The observed image is the noisy image $\tilde{u}$. The reconstructed image is the denoised image $\hat{u}$. The patch is a square centered at $\mathbf{i}$, and the sorting yielding $u(\mathbf{i})$ is replaced by a weighted average of values at all pixels $\tilde{u}(\mathbf{j})$ similar to $\mathbf{i}$. This simple change leads to the "non-local means" algorithm, which can therefore be sketched in a few rows.

Algorithm 1 Non-local means algorithm

1: **Input**: noisy image $\tilde{u}$, σ noise standard deviation. **Output**: denoised image $\hat{u}$.
2: Set parameter $\kappa \times \kappa$: dimension of patches.
3: Set parameter $\lambda \times \lambda$: dimension of research zone in which similar patches are searched.
4: Set parameter C.
5: **for** each pixel $\mathbf{i}$ **do**
6: Select a square reference sub-image (or "patch") $\tilde{P}$ around $\mathbf{i}$, of size $\kappa \times \kappa$.
7: Call $\hat{P}$ the denoised version of $\tilde{P}$ obtained as a weighted average of the patches $\tilde{Q}$ in a square neighborhood of $\mathbf{i}$ of size $\lambda \times \lambda$. The weights in the average are proportional to $w(\tilde{P}, \tilde{Q}) = e^{-\frac{d^2(\tilde{P},\tilde{Q})}{C\sigma^2}}$ where $d(\tilde{P}, \tilde{Q})$ is the Euclidean distance between patches $\tilde{P}$ and $\tilde{Q}$.
8: **end for**
9: Aggregation: recover a final denoised value $\hat{u}(\mathbf{i})$ at each pixel $\mathbf{i}$ by averaging all values at $\mathbf{i}$ of all denoised patches $\hat{Q}$ containing $\mathbf{i}$

It was also proved in **[32]** that the algorithm gave the best possible mean square estimation if the image was modeled as an infinite stationary ergodic spatial process (see Chapter 7 for an exact statement). The algorithm was called "non-local" because it uses patches $\tilde{Q}$ that are far away from $\tilde{P}$, and *even patches taken from other images*. NL-means was not the state of the art denoising method when it was proposed. As we shall see in the comparison Section 4 of

Chapter 4, the 2003 Portilla et al. [**33**] algorithm has a better PSNR performance. But quality criteria show that NL-means creates less artifacts than wavelet based methods. This may explain why patch-based denoising methods have flourished ever since. By now, 1500 papers have been published on nonlocal image processing. Patch-based methods seem to achieve the best results in denoising. Furthermore, the quality of denoised images has become excellent for moderate noise levels. Patch-based image restoration methods are used in many commercial software.

An exciting recent paper in this exploration of nonlocal methods raises the following claim [**34**]: *For natural images, the recent patch-based denoising methods might well be close to optimality.* The authors use a set of 20000 images containing about 10^{10} patches. This paper provides a second answer to the question of absolute limits raised in [**35**], "Is denoising dead?". The Cramer-Rao type lower bounds on the attainable RMSE performance given in [**35**] are actually more optimistic: they allow for the possibility of a significant increase in denoising performance. The two types of performance bounds considered in [**34**] and [**35**] address roughly the same class of patch-based algorithms. It is interesting to see that these same authors propose denoising methods that actually approach these bounds, as we shall see in Chapter 7.

The denoising method proposed in [**34**] is actually based on NL-means (algorithm 1), with the adequate parameter C to account for a Bayesian linear minimum mean square estimation (LMMSE) estimation of the noisy patch given a database of known patches. The only and important difference is that the similar patches Q are found on a database of 10^{10} patches, instead of on the image itself. Furthermore, by a simple mathematical argument and intensive simulations on the patch space, the authors are able to approach *the best average estimation error* which will *ever be attained by any patch-based denoising algorithm* [**34**].

These optimal bounds are nonetheless obtained on a somewhat restrictive definition of patch-based methods. A patch-based algorithm is understood as an algorithm that denoises each pixel by using the knowledge of: a) the patch surrounding it, and b) the probability density of all existing patches in the world. It turns out that state of the art patch-based denoising algorithms use more information taken in the image than just the patch. For example, most algorithms use the obvious but powerful trick to denoise all patches, and then to *aggregate* the estimation of all patches containing a given pixel to denoise it better. Conversely, these algorithms generally use much less information than a universal empirical law for patches. Nevertheless, the observation that at least one algorithm, BM3D [**36**] might be arguably very close to the best predicted estimation error is enlightening. Furthermore, doubling the size of the patch used in the [**34**] paper would be enough to cover the aggregation step. The difficulty is to get a faithful empirical law for 16×16 patches.

The "convergence" of all algorithms to optimality will be corroborated here by the thorough comparison of nine recent algorithms (Section 4 of Chapter 4). These state of the art algorithms seem to attain a very similar qualitative and quantitative performance. Although they initially seem to rely on different principles, our final discussion will argue that these methods are equivalent.

Image restoration theory cannot be reduced to an axiomatic system, as the statistics of images are still a widely unexplored continent. Therefore, a complete theory, or a single final algorithm

closing the problem are not possible. The problem is not fully formalized because there is no rigorous image model. Notwithstanding this limitation, rational recipes shared by all methods can be given, and the methods can be shown to rely on only very few principles. More precisely, this chapter will present the following recipes, and compare them whenever possible:

- three techniques that improve every denoising method (Chapter 6);
- a complete review on several families of homoscedastic noise estimation techniques (Section 3);

Nevertheless, this convergence of results and techniques leaves several crucial issues unsolved. (This is fortunate, as no researcher likes finished problems.) With one exception, (the BLS-GSM algorithm **[37, 14, 38]**), state of the art denoising algorithms are not multiscale. High noises and small noises also remain unexplored.

In a broader perspective, the success of image denoising marks the discovery and exploration of one of the first densely sampled high dimensional probability laws ever (numerically) accessible to mankind: the "patch space". For 8×88 patches, by applying a local PCA to the patches surrounding a given patch, one can deduce that this space has a dozen significant dimensions (the others being very thin). Exploring its structure, as was initiated in **[39]**, seems to be the first step toward the statistical exploration of images. But, as we shall see, this local analysis of the *patch space* already enables state of the art image denoising.

Most denoising and noise estimation algorithms commented here are be available at the *Image Processing on Line* (IPOL) journal, `http://www.ipol.im/`. Each algorithm is given a complete description, the corresponding source code, and can be run online on arbitrary images.

2. Noise models

The main source of noise in digital images is the so-called *shot noise*, which is is inherent to photon counting. The value $\tilde{u}(\mathbf{i})$ observed by a sensor at each pixel $\mathbf{i}$ is a Poisson random variable whose mean would be the ideal image. The standard deviation of this Poisson distribution is equal to the square root of the number of incoming photons $\tilde{u}(\mathbf{i})$ in the pixel captor $\mathbf{i}$ during the exposure time. This noise adds up to a thermal noise and to an electronic noise which are approximately additive and white.

For sufficiently large values of $\tilde{u}(\mathbf{i})$, $(\tilde{u}(\mathbf{i}) > 1000)$, the normal distribution $\mathcal{N}(\tilde{u}(\mathbf{i}), \sqrt{\tilde{u}(\mathbf{i})})$ with mean $\tilde{u}(\mathbf{i})$ and standard deviation $\sqrt{\tilde{u}(\mathbf{i})}$ is an excellent approximation to the Poisson distribution. If $\tilde{u}(\mathbf{i})$ is larger than 10, then the normal distribution still is a good approximation if an appropriate continuity correction is performed, namely $\mathbb{P}(\tilde{u}(\mathbf{i}) \leq a) \simeq \mathbb{P}(\tilde{u}(\mathbf{i}) \leq a + 0.5)$, where a is any non-negative integer.

As a rule of thumb, the noise model is relatively easy to estimate when the raw image comes directly from the imaging system, in which case the noise model is known and only a few parameters must be estimated. For this, efficient methods are described by Foi et al. **[40, 7]** for Poisson and Gaussian noise.

Nevertheless, the pixel value is *signal dependent*, since its mean and variance depend on $\tilde{u}(\mathbf{i})$. To get back to the classic "white additive Gaussian noise" used in most researches on image denoising, a *variance-stabilizing transformation* can be applied: when a variable is Poisson distributed with parameter $\tilde{u}(\mathbf{i})$, its square root is approximately normally distributed with expected value of about $\sqrt{\tilde{u}(\mathbf{i})}$ and variance of about $1/4$. Under this transformation, the convergence to normality is faster than for the untransformed variable [**41**]. The most classic VST is the Anscombe transform [**42**] which has the form $f(u_0) = b\sqrt{u_0 + c}$.

The denoising procedure with the standard variance stabilizing transformation (VST) procedure follows three steps:

(1) apply VST to approximate homoscedasticity;
(2) denoise the transformed data;
(3) apply an inverse VST.

Note that the inverse VST is not just an algebraic inverse of the VST, and must be optimized to avoid bias [**43**].

Consider any additive signal dependent noisy image, obtained for example by the Gaussian approximation of a Poisson variable explained above. Under this approximation, the noisy image satisfies $\tilde{u} \simeq \tilde{u} + g(\tilde{u})n$ where $n \simeq \mathcal{N}(0, 1)$. We can search for a function f such that $f(\tilde{u})$ has uniform standard deviation,

$$f(\tilde{u}) \simeq f(\tilde{u}) + f'(\tilde{u})g(\tilde{u})n.$$

Forcing the noise term to be constant, $f'(\tilde{u})g(\tilde{u}) = c$, we get

$$f'(\tilde{u}) = \frac{c}{g(\tilde{u})},$$

and integrating

$$f(\tilde{u}) = \int_0^{\tilde{u}} \frac{c\,dt}{g(t)}.$$

When a linear variance noise model is taken, this transformation gives back an Anscombe transform. Most classic denoising algorithms can also be adapted to signal dependent noise. This requires varying the denoising parameters at each pixel, depending on the observed value $\tilde{u}(\mathbf{i})$. Several denoising methods indeed deal directly with the Poisson noise. Wavelet-based denoising methods [**44**] and [**45**] propose to adapt the transform threshold to the local noise level of the Poisson process. Lefkimmiatis et al. [**46**] have explored a Bayesian approach without applying a VST. Deledalle et al., [**47**] argue that for high noise level it is better to adapt NL-means than to apply a VST. These authors proposed to replace the Euclidean distance between patches by a likelihood estimation taking into account the noise model. This distance can be adapted to each noise model such as the Poisson, the Laplace or the Gamma noise [**48**], and to more complex (speckle) noise occurring in radar (SAR) imagery [**49**].

Nonetheless, dealing with a white uniform Gaussian noise makes the discussion on denoising algorithms far easier. The recent papers on the Anscombe transform [**43**] (for low count Poisson noise) and [**50**] (for Rician noise) argue that, when combined with suitable forward and inverse VST transformations, algorithms designed for homoscedastic Gaussian noise work just as well as

ad-hoc algorithms signal-dependent noise models. This explains why in the rest of this chapter the noise is assumed uniform, white and Gaussian, having previously applied, if necessary, a VST to the noisy image. This also implies that we deal with *raw* images, namely images as close as possible to the direct camera output before processing. Most reflex cameras, and many compact cameras nowadays give access to this raw image.

But there is definitely a need to denoise current image formats, which have undergone unknown alterations. For example, the JPEG-encoded images given by a camera contain a noise that has been altered by a complex chain of algorithms, ending with lossy compression. Noise in such images cannot be removed by the current state of the art denoising algorithms without a specific adaptation. The key is to have a decent noise model. For this reason, it is important to be able to estimate the noise from the noisy image itself, without assuming any noise model and without relying or trusting any prior information, such as the model of the captor or the kinds of transformations that the image might have undergone.

Compared to the denoising literature, research on noise estimation is a poor cousin. Few papers are dedicated to this topic. Among the recent papers one can mention [**51**], which argues that images are scale invariant and therefore noise can be estimated by a deviation from this assumption. Unfortunately this method is not easily extended to estimate scale dependent or signal dependent noise, like the one observed in most digital images in compressed format. As a rule of thumb, the noise model is relatively easy to estimate when the raw image comes directly from the imaging system, in which case the noise model is known and only a few parameters must be estimated. For this, efficient methods are described in [**7**], [**40**] for Poisson and Gaussian noise.

In this chapter we will focus on methods that allow for local, signal and scale dependent noise. Later on Chapter 3 it will be shown why only considering signal and scale dependent noise is not enough and Chapter 4 will present our proposed algorithm for signal, frequency, and scale dependent noise.

One cannot denoise an image without knowing its noise model. It might be argued that the noise model comes with the knowledge of the imaging device. Nevertheless, the majority of images dealt with by the public or by scientists have lost this information. This loss is caused by format changes of all kinds, which may include resampling, denoising, contrast changes and compression. All of these operations change the noise model and make it *signal and scale dependent*.

The question that arises is why so many researchers are working so hard on denoising models, if their corpus of noisy images is so ill-informed. It is common practice among image processing researchers to add the noise themselves to noise-free images to demonstrate the performance of a method. This proceeding permits to reliably evaluate the denoising performance, based on a controlled ground truth. Nevertheless the denoising performance may, after all, critically depend on how well we are able to estimate the noise. Most world images are actually encoded with lossy JPEG formats. Thus, noise is partly removed by the compression itself. Furthermore, this removal is scale dependent. For example, the JPEG 1985 format divides the image into a disjoint set of 8×8 pixels blocks, computes their DCT, quantizes the coefficients and the small ones are replaced

by zero. This implies that JPEG performs a frequency dependent threshold, equivalent to a basic Wiener filter. The same is true for JPEG 2000 (based on the wavelet transform).

In addition, the Poisson noise of a raw image is signal dependent. The typical image processing operations, demosaicking, white balance and tone curve (contrast change) alter this signal-dependency in a way which depends on the image itself.

In short:

- the noise model is different for each image;
- the noise is signal dependent;
- the noise is scale dependent;
- the knowledge of each dependence is crucial to denoise properly any given image which is not raw, and for which the camera model is available.

Thus, estimating JPEG noise is a complex and risky procedure, as well explained in [**52**] and [**53**]. It is argued in [**54**] that noise can be estimated by involving a denoising algorithm. Again, this procedure is probably too risky for noise and scale dependent signal.

A review and comparison of classic homoscedastic noise estimation methods is presented in Section 3.

3. Review of homoscedastic block-based noise estimators

This chapter presents a comparison and discussion of classic block-based noise estimation methods, most of them from [**55**]. To evaluate them, ten noise-free images are used in the tests. Noiseless images are obtained by taking snapshots with a reflex camera of scenes under good lighting conditions and with a low ISO level. To reduce further the noise level, the average of each block of 5×5 pixels is computed, reducing therefore the noise by a 5 factor. Since the images are RGB, the mean of the three channels can computed, reducing the noise by a further $\sqrt{3}$ factor. The noise is therefore reduced by a $5\sqrt{3} \simeq 8.66$ factor, and the images can be considered noise-free. The set of images can be seen in Figure 3. Some of them present large flat regions with little texture, while others do not contain any flat regions and are highly textured. The size of each noise-free test image is 704×469 pixels.

In this chapter we only evaluate the performance of the estimator using simulated homoscedastic (fixed variance) noise. The noise is added to the noise-free images and then the estimation given by each noise estimator is compared to the value of the simulated noise. Seven noise levels were applied to these noise-free images: $\sigma \in \{1, 2, 5, 10, 20, 50, 80\}$. Figure 2 shows the result of adding white homoscedastic Gaussian noise with $\sigma \in \{1, 2, 5, 10, 20, 50, 80\}$ to the noise-free image *traffic*.

These images, from left to right and from top to bottom will be referred to as *bag*, *building1*, *computer*, *dice*, *flowers2*, *hose*, *leaves*, *lawn*, *stairs*, and *traffic*. For the uniform-noise tests, five noise levels were applied to these noise-free images: $\sigma \in \{1, 2, 5, 10, 20, 50, 80\}$. Figure 2 shows the result of adding white Gaussian noise with $\sigma \in \{1, 2, 5, 10, 20, 50, 80\}$ to the noise-free image *traffic*.

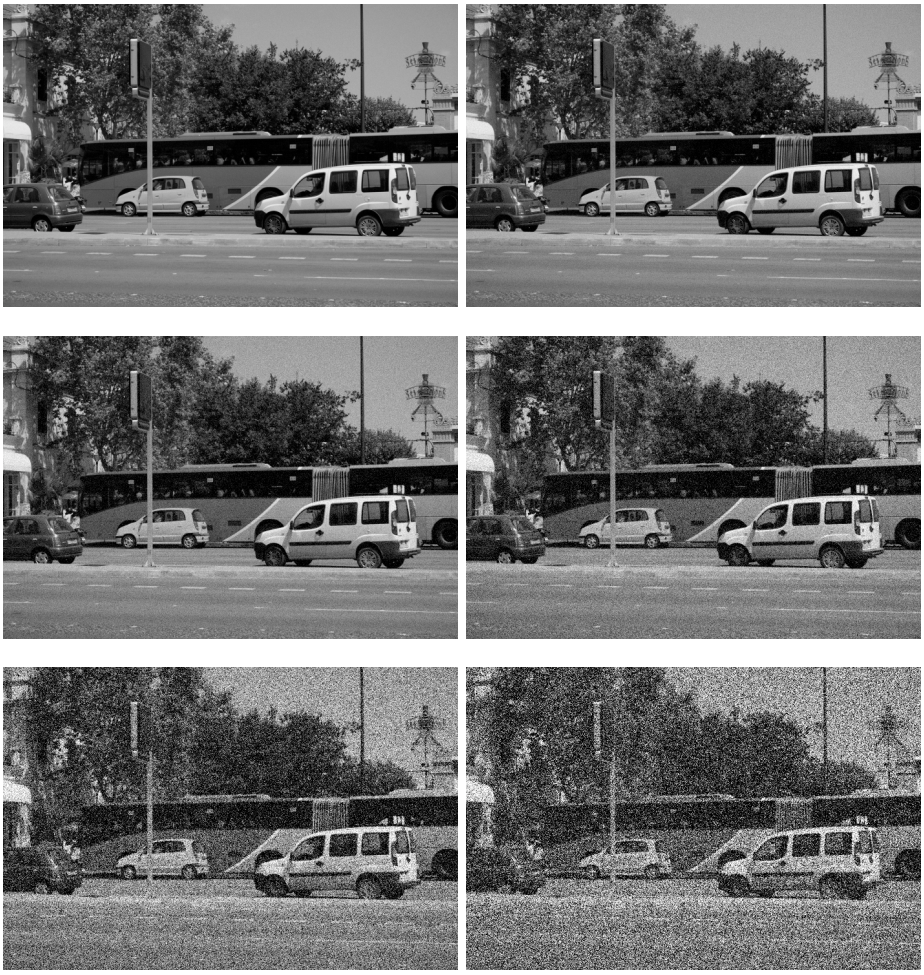


FIGURE 2. Result of adding white homoscedastic Gaussian noise with $\sigma \in \{2, 5, 10, 20, 50, 80\}$ to the noise-free image *traffic*. It may need a zoom in to perceive the noise for $\sigma = 2$ and $\sigma = 5$.

FIGURE 3. Noise-free images used in the tests. From left to right and from top to bottom: *bag*, *building1*, *computer*, *dice*, *flowers2*, *hose*, *leaves*, *lawn*, *stairs* and *traffic*.

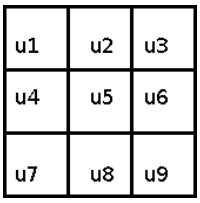


FIGURE 4. 3×3 window centered at pixel u_5.

3.1. The Average method. The "Average method" **[55]** tries to minimize the effect of edges and textures in the image by subtracting a low-pass filtered version of the image from the noisy image: $U = I - f(I)$. In the Average method, function f is the mean of the pixels centered at some given pixel with a $w \times w$ pixels window ($w = 3$). Since the blocks only contain 9 pixels, the sample variance is biased and must be corrected. Indeed, if pixel u_5 is in the center of the block (Figure 4), the computed sample variance is $S^2_{9-1}\left(u_5 - \sum_{i=1}^{9} u_i\right) = S^2_8\left(u_5 - \frac{1}{9}u_5 - \frac{1}{9}\sum_{i=1,i\neq 5}^{9} u_i\right) = S^2_8\left(\frac{8}{9}u_5\right) + \frac{1}{81}S^2_8\left(\sum_{i=1,i\neq 5}^{9} u_i\right) = \frac{64}{81}\sigma^2 + \frac{8}{81}\sigma^2 = \frac{8}{9}\sigma^2$. Here, S^2_8 is the sample variance. That is, $S^2_8(B) = \frac{1}{8}\sum_{i=1}^{9}(B_i - \bar{B})$, where $\bar{B} = \frac{1}{9}\sum_{i=1}^{9} B_i$ and B_i are pixel intensities of the block B. To avoid that the remaining edges and textures affect the noise estimation, blocks where the magnitude of the intensity gradient is above a certain threshold are discarded. The gradient is computed at the central pixel of each block. For example, if u_5 is the central pixel (Figure 4), the magnitude of the gradient would be calculated as $\sqrt{(u_6 - u_5)^2 + (u_8 - u_5)^2}$. The threshold is obtained from the normalized cumulative histogram of the magnitude of the intensity gradients. The threshold is the value in the cumulative histogram corresponding to the p percent of the blocks, with $p = 1\%$. Because of the fact that only blocks with smallest variance are taken into account, the estimate is biased and has to be corrected by a factor learned on noise in order to get the final $\tilde{\sigma}$ estimate. With $p = 1\%$, the empirical correction factor is $k = 1.1609$. The standard deviation of all blocks whose intensity gradient magnitude is below the threshold is computed. The mean of these standard deviations, multiplied by the correction factor, is estimated as $\tilde{\sigma}$. The pseudo-code for the Average method is in Algorithm 2. The results for white Gaussian noise are given in Table 1.

Algorithm 2 Average noise estimation algorithm.

AVERAGE - Return the standard deviation of the image noise. **Input U**: input image. **Input** p: percent of pixels to compute (1% typically). **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$w = 3$.

$k = 1.1608823968593502$.

$g[x, y] = \sqrt{(U[x + 1, y] - U[x, y])^2 + U[x, y + 1] - U[x, y])}, \forall (x, y) \in \text{supp}(I)$. ▷ Compute gradient magnitude

$h_c, h_l = HISTOGRAM(g)$. ▷ Compute the histogram of g. Here, h_c is a list containing the number of elements in each bin and h_c a list containing the limits h_l of each bin.

$\text{acc} = \text{ACC}(h_c, h_l)$. ▷ Accumulated histogram of g

$i = i : \text{acc}[i] = \frac{p}{100}$. ▷ Index of threshold bin

$t = \frac{h_l[i]+h_l[i+1]}{2}$. ▷ Get threshold

$B \leftarrow$ all $w \times w$ overlapped blocks in U.

$F \leftarrow \text{mean}(B)$ ▷ Mean of all overlapped blocks in U

$D = C(B) - F$. ▷ Filter image: the mean of the block is subtracted from the central pixel $C(B)$ in the block B.

$S = 0$. ▷ List of standard deviations

for all $w \times w$ blocks $B_D \in$ D, where (x, y) is the center of the block **do**

  if $g[x, y] < t$ **then**

    $S \leftarrow \sqrt{\frac{9}{8} S^2_{w^2-1}(B_D)}$

  end if

end for

return $\tilde{\sigma} = k \times \text{mean}(S)$. ▷ k is the correction factor

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	6.00	6.75	9.18	13.56	22.45	51.51	80.79
building1	1.99	3.25	6.77	12.23	22.12	51.62	81.17
computer	3.51	4.40	7.68	12.79	22.58	51.96	81.56
dice	1.30	2.27	5.26	10.27	20.27	50.35	80.61
flowers2	1.69	2.71	5.74	10.63	20.57	50.67	80.37
hose	2.46	3.31	6.05	10.80	20.60	50.25	80.37
leaves	4.91	5.58	7.98	12.28	21.58	50.89	80.48
lawn	5.13	5.79	8.05	12.30	21.58	50.96	80.78
stairs	2.37	3.33	6.08	10.75	20.48	50.39	80.39
traffic	4.04	5.26	8.63	13.74	23.37	52.30	81.44
constant	1.00	2.01	5.00	10.01	20.04	50.06	80.52

TABLE 1. Average method results with simulated homoscedastic white Gaussian noise.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	6.39	7.18	9.67	14.03	22.92	52.01	81.46
building1	2.03	3.33	6.92	12.51	22.46	52.11	81.80
computer	3.63	4.56	8.00	13.22	23.06	52.61	82.34
dice	1.33	2.31	5.34	10.38	20.42	50.68	81.23
flowers2	1.78	2.85	5.97	10.90	20.87	51.12	80.89
hose	2.56	3.42	6.18	10.94	20.82	50.68	81.00
leaves	5.13	5.83	8.23	12.53	21.86	51.34	81.10
lawn	5.37	6.04	8.28	12.55	21.90	51.39	81.53
stairs	2.46	3.45	6.22	10.90	20.70	50.79	81.05
traffic	4.21	5.58	9.14	14.26	23.98	53.04	82.22
constant	1.01	2.02	5.03	10.07	20.17	50.36	81.04

TABLE 2. Median method results with simulated homoscedastic white Gaussian noise.

3.2. Median method. The Median method **[55]** is a variant of the Average (Section 3.1). In this method, function f to filter the image is the median. The same correction factor used for the Average method is used here. The white Gaussian noise estimates are given in Table 2. The only difference with the Average method in the algorithm description is that the mean of the block is replaced by the median of the block.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	3.51	4.20	7.07	12.36	22.10	51.87	81.00
building1	1.23	2.20	5.28	10.50	20.58	51.04	81.02
computer	1.42	2.42	5.54	10.62	20.82	51.24	81.63
dice	1.20	2.16	5.11	10.09	20.13	50.21	80.04
flowers2	1.23	2.23	5.34	10.47	20.53	50.52	80.26
hose	1.61	2.59	5.61	10.58	20.45	50.40	79.97
leaves	2.96	3.65	6.54	11.53	21.33	50.75	80.81
lawn	2.43	3.29	6.35	11.48	21.14	50.55	80.75
stairs	1.37	2.44	5.51	10.59	20.39	50.31	79.76
traffic	1.32	2.33	5.55	10.77	21.23	51.34	81.31
constant	1.00	1.99	5.01	9.93	20.00	49.90	80.11

TABLE 3. Block method results with simulated homoscedastic white Gaussian noise.

3.3. Block method. The Block method [**56, 57**] measures the noise from the set of 7×7 pixels blocks with minimal variance. The size of this set of blocks is chosen as a fixed small percentage p of the total number of pixels in the image. The direct estimation $\hat{\sigma}^2$ is the mean of the set of sample variances under the p-percentile. Of course, this produces a biased result, because only the blocks with the minimal variance are chosen. Therefore, to obtain $\tilde{\sigma}^2$, the direct sampled variance estimation $\hat{\sigma}^2$ has to be multiplied by a correction factor. This factor is $k = 1.8350$ for $p = 1\%$. The pseudo-code for the Block method is in Algorithm 3 and its results with white Gaussian noise in Table 3. Two different block sized where tests: $w = 11$ (Table 4) and $w = 15$ (Table 5), but increasing the block size does not give, in general, better results. The computation time, however, increases significantly with the block size. The correction factors for $w = 11$ is$k = 1.4429$ and for $w = 15$ it is $k = 1.2994$.

Algorithm 3 Block algorithm.

1: **BLOCK** - Return the standard deviation of the image noise. **Input U**: input image. **Input** p: percentage of blocks to compute (1% typically). **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

2: $w = 7$.
3: $k = 1.8350297625014702$.
4: $B \leftarrow$ all $w \times w$ blocks in U.
5: $V = 0$ ▷ List of sample variances
6: **for** all $w \times w$ blocks $B_D \in B$ **do**
7: $\quad V \leftarrow S^2_{w^2-1}(B_D)$. ▷ Add sample variances to list V
8: **end for**
9: $\tilde{\sigma} = k\sqrt{\text{Mean}\left(V\left[0 \ldots |V|\frac{p}{100}\right]\right)}$. ▷ k is the correction factor

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	4.58	5.01	7.31	12.46	22.19	52.03	81.33
building1	1.23	2.17	5.22	10.35	20.43	50.94	80.86
computer	1.42	2.41	5.48	10.59	20.80	50.94	81.62
dice	1.21	2.18	5.12	10.12	20.14	50.22	80.20
flowers2	1.20	2.19	5.29	10.42	20.60	50.59	80.35
hose	1.70	2.60	5.63	10.57	20.54	50.46	80.23
leaves	3.37	3.91	6.54	11.50	21.33	50.97	80.88
lawn	2.77	3.46	6.25	11.36	21.13	50.78	80.72
stairs	1.31	2.37	5.43	10.54	20.43	50.47	79.82
traffic	1.31	2.28	5.42	10.61	21.05	51.37	81.17
constant	1.00	2.00	5.01	9.93	20.01	49.96	80.03

TABLE 4. Block method results with simulated homoscedastic white Gaussian noise ($w = 11$).

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	5.67	5.98	7.88	12.65	22.30	52.20	81.43
building1	1.24	2.16	5.20	10.33	20.36	50.90	81.13
computer	1.43	2.40	5.45	10.52	20.70	50.95	81.40
dice	1.22	2.19	5.14	10.14	20.12	50.21	80.23
flowers2	1.19	2.18	5.26	10.34	20.55	50.66	80.50
hose	1.79	2.64	5.61	10.56	20.59	50.35	80.21
leaves	3.79	4.26	6.66	11.46	21.23	50.99	80.90
lawn	2.99	3.58	6.20	11.26	21.18	50.83	80.52
stairs	1.28	2.33	5.38	10.53	20.48	50.50	79.91
traffic	1.34	2.28	5.36	10.54	20.94	51.32	81.27
constant	1.00	2.00	5.01	9.97	19.97	49.99	80.08

TABLE 5. Block method results with simulated homoscedastic white Gaussian noise ($w = 15$).

3.4. Gradient method. In the Gradient method [**58, 59**], the magnitude of the intensity gradient is computed using a 3×3 least squares fit for all possible pixels of the image (excluding those pixels in the edges of the image, where the gradient can not be computed). For the 3×3 least squares fit, both the Sobel and Prewitt operators were tested. The Sobel operator uses the following matrices for the horizontal and vertical gradient components:

$$S_x = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix}, \; S_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix}$$

The Prewitt operator uses these matrices:

$$P_x = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix}, \; P_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix}$$

The same results were obtained computing the gradient with the Sobel or the Prewitt operators. Because for a white Gaussian noise the horizontal and vertical magnitudes in the intensity gradient are normally distributed (with mean $\mu = 0$ and variance σ^2), the magnitude of the noise intensity gradient is Rayleigh distributed. Indeed, if X and Y are normal random variables with mean $\mu = 0$ and variance σ_R^2, then $R = \sqrt{X^2 + Y^2}$ is Rayleigh distributed, with parameter σ_R. Note that σ_R is not the variance of the noise, but the variance of the directional gradient components.

It is important that blocks do not overlap, because the random variables X and Y must be independent in order to obtain a Rayleigh distribution R with σ parameter. A histogram of the magnitude of the intensity gradient is made. As explained in Section 1 of Chapter 11, edges and textures affect the right part of the histogram mainly. If the image is not dominated by edges and textures, the σ_R parameter can be estimated by finding the mode (the value at which the distribution is maximal) of the Rayleigh distribution. However, this is only possible if edges and textures are moderate, and therefore not affecting the part of the histogram containing the mode. Before finding the mode of the distribution, the histogram is filtered at least four times. To low-pass filter the histogram, the same procedure used to filter the noise curves explained in Chapter 1.2 was applied.

Once the mode σ_R is found in the filtered histogram, the noise standard deviation $\tilde{\sigma}$ can be obtained by multiplying the mode by a correction factor $f = 0.7607$. Unfortunately, trying to fit a Rayleigh distribution to the normalized histogram is a difficult task, because edges and textures affect the histogram, changing the shape of the distribution. When normalizing the histogram, these undesired values in the tail change the shape of the normalized distribution. Therefore, even if a Rayleigh if fitted onto the steep rising portion of the affected histogram, it would not be fitting the noise image distribution. Figure 5 illustrates this problem. In green, the ground-truth distribution (pure noise). In blue, the distribution obtained with a noisy image with several edges and textures. In red, the Rayleigh distribution fitted to the steep rising part (20%) of the histogram. On the left, fitting a Rayleigh distribution to the steep rising portion histogram of the

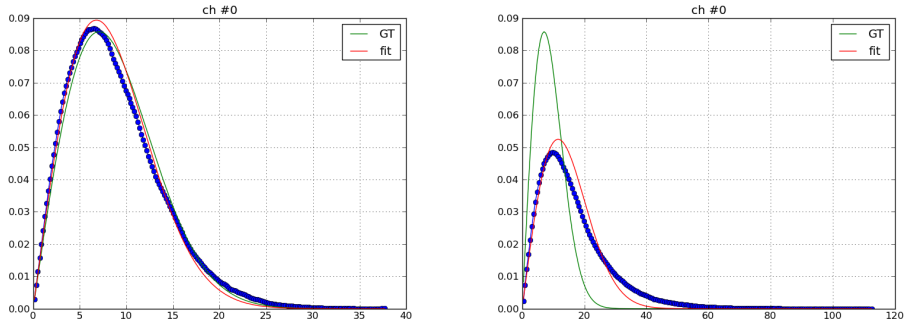


FIGURE 5. Left: fitting a Rayleigh distribution to the steep rising portion (20%) histogram of the variance of pure-noise ($\sigma = 5$) patches. The fit (red) is very close to the ground truth (green) and therefore the obtained $\hat{\sigma}$ is accurate. Right: trying to fit a Rayleigh distribution to the steep rising portion (20%) histogram of the variance of the *leaves* image (many edges and textures). The obtained parameter σ does not correspond to that of the noise (ground truth distribution).

variance of pure-noise ($\sigma = 5$) patches. The fit (red) is computed over the ground truth (green). Thus the obtained $\hat{\sigma}$ is accurate. On the right, trying to fit a Rayleigh distribution to the steep rising portion histogram of the variance of the *leaves* image (many edges and textures). The obtained parameter σ does not correspond to that of the noise (ground truth distribution). The pseudo-code for the algorithm is presented in Algorithm 4 and the results for white Gaussian noise in Table 6.

Algorithm 4 Gradient noise estimation algorithm.

GRADIENT - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$f = 0.76071735110441019$. ▷ Correction factor

for $[x, y] \in [1, N_x - 2] \times [1, N_y - 2]$ **do**

$g_x[x, y] = (U[x+1, y-1]+2U[x+1, y]+U[x+1, y+1]-U[x-1, y-1]-2U[x-1, y]-U[x-1, y+1])/8$

$g_y[x, y] = (U[x-1, y-1]+2U[x, y-1]+U[x+1, y-1]-U[x-1, y+1]-2U[x, y+1]-U[x+1, y+1])/8.$

$g[x, y] = \sqrt{g_x^2 + g_y^2}$ ▷ Compute 3×3 least squares fit gradient magnitude

end for

$H = HISTOGRAM(g)$. ▷ Compute the histogram of g. Here, H is a list with the number of elements in each bin.

$k = \text{argmax}_k(H[k])$.

mode = median(bin_k). ▷ Median of the elements in bin $\#k$

$\tilde{\sigma} = f \times \text{mode}$.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	5.04	5.42	7.47	12.58	20.64	47.99	74.67
building1	0.76	2.78	6.20	11.14	20.91	51.57	77.98
computer	1.02	1.09	6.75	11.67	20.52	49.16	80.61
dice	0.92	2.10	4.97	9.71	19.94	51.17	80.50
flowers2	0.74	2.73	6.35	9.83	19.43	51.44	83.17
hose	2.14	3.12	5.81	10.68	20.39	48.95	75.72
leaves	4.26	5.35	6.77	12.09	19.19	51.54	82.97
lawn	4.06	5.05	7.07	11.81	21.49	49.06	76.83
stairs	2.21	3.19	5.81	10.52	20.37	50.67	83.89
traffic	0.96	1.06	6.62	11.33	23.15	49.29	80.19
constant	0.98	2.00	5.31	9.43	17.72	50.89	80.32

TABLE 6. Gradient method results with simulated homoscedastic white Gaussian noise.

Rule #	Condition	Computation of $\tilde{\sigma}$
1	$q[l,4] < 1$	$\tilde{\sigma} = 0$
2	$l_0 = n$	$\tilde{\sigma} = \sqrt{v[n]}$
3, 4	$l_0 = 3$ or $l_0 = 4$	WARNING: CAN NOT ESTIMATE
5.1	$\rho_0 < \rho \leq \rho_1$	$\tilde{\sigma} = \sqrt{v[3]}$
5.2	$\rho_1 < \rho \leq \rho_2$	$\tilde{\sigma} = \sqrt{\delta v[3] + (1-\delta)v[4]}$
5.3	$\rho_2 < \rho \leq \rho_3$	$\tilde{\sigma} = \sqrt{\delta v[4] + (1-\delta)v[5]}$
5.4	$\rho_3 < \rho \leq \rho_4$	$\tilde{\sigma} = \sqrt{\delta v[5] + (1-\delta)v[6]}$
6.1	$T < \alpha[l_0 - 1] \leq 0$	$\tilde{\sigma} = \sqrt{\delta v[l_0 - 2] + (1-\delta)v[l_0 - 1]}$
6.2	$-1 < \alpha[l_0] \leq -0.5$	$\tilde{\sigma} = \sqrt{\delta v[l_0 - 2] + (1-\delta)v[l_0 - 1]}$
6.3	$-0.5 < \alpha[l_0] \leq T$	$\tilde{\sigma} = \sqrt{((1+\delta)v[l_0 + 1] + (1-\delta)v[l_0])/2}$

TABLE 7. Summary of the Pyramid method rules.

3.5. Pyramid method. The Pyramid method **[6]** considers n levels in the input image, $l = 2, 3, \ldots, n$, where $N = 2^n$ is the size of the image. At each level, the variance of all non-overlapping blocks of size $2^l \times 2^l$ is computed. To compute this variance, the unbiased estimator for the population variance is used: $S^2_{B-1}(X) = \frac{1}{B-1} \sum_{i=1}^{B} (X - \bar{X})$, where $\bar{X} = \frac{1}{B} \sum_{i=1}^{B} x_i$ and B the number of elements in X. For $l = 2, 3, \ldots, n-1$, the four smallest values for each level l are stored. A slippage test in levels $l = 2, 3, \ldots, n-2$ is done and the set $v(l)$ is defined as the mean of the smallest values (between one and four values are used, depending on the case). Finally, the contribution to $v(l)$ (signal) and that of the noise is obtained by analyzing the shape of $\alpha(l) = \frac{v(l-1)}{v(l)} - \beta(l)$. The function β is experimentally determined by the authors to be $\beta(l) = 1 - \frac{2^{6-l}}{10}$. The method proposes several conditional rules that, if held, give an estimate of the noise. It can happen that the method determines that the noise is poorly separated from the signal and no result is obtained. However, the method can detect this situation and prevents giving a bad result. The results for the Pyramid method for white Gaussian noise are given in Table 9. Code *NE* means that the method considered that noise and signal were not well separated and therefore it did not gave any estimation. Rule 6.3 was applied when no rule held.

Function α is computed at each level l as $\alpha[l] = \frac{v[l-1]}{v[l]} - \beta[l]$. l_u is the minimal argument l that gets $\alpha[l] < 0$ for $l = 3, \ldots, n$ and l_0 the minimal argument l that gets $\sum_{i=l_u}^{l} \alpha[l] < T$ with $l = 3, \ldots, n-2$, where $T = -0.1$. The pyramid method proposes several rules as conditions that if held, provide a way to get the noise standard deviation estimation. This rules are summarized in Table 7. Some of the rules interpolate to get the value of δ. The interpolation function is: $\text{INTERPOLATION}(a, \rho, b) = \frac{\rho - b}{a - b}$. Rules 5.2, 5.3, 5.4, 6.1, 6.2 and 6.3 use interpolation. The interpolation parameters to get δ according to the rule are given in Table 8. Rules 5.1, 5.2, 5.3 and 5.4 use the following variables: $\rho = \alpha[5] + \alpha[6]$, $\rho_0 = -2$, $\rho_1 = -1.5$, $\rho_2 = -1$, $\rho_3 = -0.5$, $\rho_4 = T$, where $T = -0.1$.

Algorithm 5 Pyramid noise estimation algorithm.

PYRAMID - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$n = \frac{\log(N)}{\log(2)}$. ▷ Image is $N \times N$ and $N = 2^n$.
$\beta[l] = 1 - 0.1 \times 2^{6-l}$ for $l = 1 \dots n$.
$Q = \emptyset$. i-order statistic $q[l, i], i \in \{1, 2, 3, 4\}$
$r = \emptyset$. ▷ Thresholds $r[l, i], l \in \{1, 2, \dots, n\}, i \in \{1, 2, 3\}$
$v = \emptyset$. ▷ Variances $v[l], l \in \{1, 2, \dots, n\}$
for $l = 1$ to n **do**
    $K = 4^{n-l}$, $c = 2^l$, $V = \emptyset$. ▷ List of variances in level l
    for $x = 0$ to N step c **do**
        for $y = 0$ to N step c **do**
            $v = \text{Var}(u[x \dots, x + c - 1, y \dots, y + c - 1])$. ▷ Sample variances
            $V \leftarrow v$. ▷ Store sample variance
        end for
    end for
    $q[l, 1], q[l, 2], q[l, 3], q[l, 4] = \text{Min}(V)$. ▷ Find 1, 2, 3 and 4 order statistics of V
    $r[l, 1] = (q[l, 2] - q[l, 1])/(q[l, 4] - q[l, 1])$
    $r[l, 2] = (q[l, 3] - q[l, 2])/(q[l, 4] - q[l, 2])$
    $r[l, 3] = (q[l, 4] - q[l, 3])/(q[l, 4] - q[l, 2])$
    $v[l] = \text{Mean}(q[l, :])$ **if** $r[l, 1] \leq 0.5$.
    $v[l] = (q[l, 2] + q[l, 3] + q[l, 4])/3$ **if** $r[l, 2] \leq 0.7$.
    $v[l] = (q[l, 3] + q[l, 4])/2$ **if** $r[l, 3] <= 0.7$.
    $v[l] = q[l, 4]$ **otherwise**.
end for
Compute α and l_u
PYRAMID_APPLY_RULES.

Rule #	Interpolation
5.2	$\delta = \text{INTERPOLATION}(\rho_1, \rho, \rho_2)$
5.3	$\delta = \text{INTERPOLATION}(\rho_2, \rho, \rho_3)$
5.4	$\delta = \text{INTERPOLATION}(\rho_3, \rho, \rho_4)$
6.1	$\delta = \text{INTERPOLATION}(T, \alpha[l_0 - 1], 0)$
6.2	$\delta = \text{INTERPOLATION}(-1, \alpha[l_0], -0.5)$
6.3	$\delta = \text{INTERPOLATION}(-0.5, \alpha[l_0], T)$,

TABLE 8. Pyramid method interpolation parameters.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	NE	NE	NE	19.40	26.37	49.03	73.02
building1	0.00	1.96	4.44	8.92	21.12	62.97	83.91
computer	0.00	1.82	4.38	10.08	20.63	69.31	75.54
dice	0.00	0.00	4.32	10.88	29.48	43.42	75.34
flowers2	0.00	1.58	5.28	9.80	23.04	64.19	84.65
hose	0.00	NE	6.30	12.37	19.46	46.34	79.86
leaves	3.25	3.63	6.72	13.28	28.53	50.15	83.29
lawn	3.13	4.39	6.66	10.18	27.44	49.68	74.18
stairs	0.00	1.87	5.37	13.28	19.25	48.51	79.78
traffic	0.00	NE	5.01	10.37	25.69	52.96	67.94
constant	0.00	0.00	4.61	7.27	19.39	36.29	77.57

TABLE 9. Pyramid method results with simulated homoscedastic white Gaussian noise.

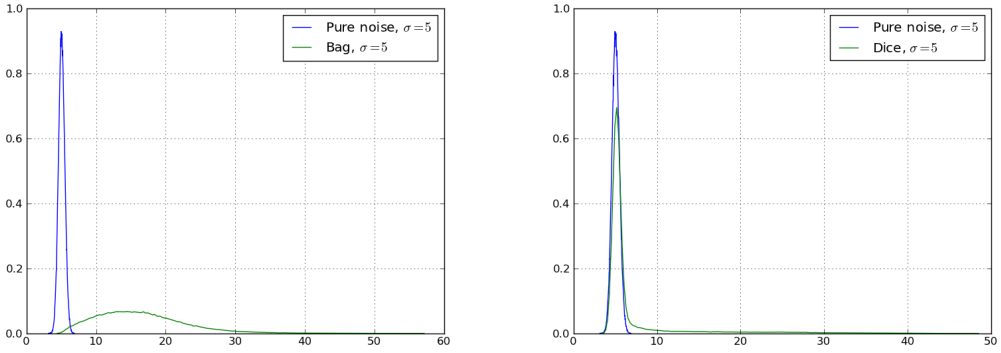


FIGURE 6. Problem of the *Scatter* method when trying to figure out the standard deviation of the noise from the peak of the histogram in images dominated by textures and edges. On the left, the normalized histograms of standard deviations for pure noise and *bag* images. On the right, the normalized standard deviation histograms for pure noise and *dice* images. The noise standard deviation is $\sigma = 5$ for both images.

3.6. The Scatter method. The Scatter method **[60]** is intended not only for additive noise, but also for multiplicative noise. It uses the Hough transform to have a plot of the averages M and variances V of all 8×8 blocks in the image. Then, it fits a straight line to the main cluster, using the Hough transform. The slope and intercept of this line are the multiplicative and additive components of the noise. In the case of signal independent white Gaussian noise, the Hough transform boils down to the largest peak of the standard deviations histogram. Therefore the standard deviation is figured out by the largest peak, which is enough to compare this method with the other ones.

In the white Gaussian noise test, it was used the largest peak in the histogram to figure out $\tilde{\sigma}$ (Algorithm 7). When the peak of the histogram technique is used, $\hat{\sigma}$ must be corrected by a factor $k = 1.0229$ to get the final estimation $\tilde{\sigma}$. If the image is dominated by edges and textures, the Scatter method fails to get a reliable estimation because the large peak can be produced by them and not by the noise.

Figure 6 illustrates this problem. On the left, it shows the normalized histogram of standard deviations corresponding to pure noise and the one corresponding to the *bag* image, both with $\sigma = 5$. On the right, the normalized histograms of a pure noise and of the *dice* image. The image *bag* has many edges and textures, while the *dice* image has not. The estimate in *bag* is wrong, because the variance retrieved from the blocks corresponds to the variations of the signal, rather than noise. The estimate in *dice* is accurate, because this image contains large flat regions and little texture (although a small bump can be observed on the right part of the histogram). When the peak of the histogram is used to get $\hat{\sigma}$, a correction factor $k = 1.0229$ must be used to get $\tilde{\sigma}$. The pseudo-code for the Scatter method for signal-dependent noise is Algorithm 6. However, here

the method is evaluated for white Gaussian noise (Algorithm 7) and its results are shown in Table 10.

Algorithm 6 Scatter signal-dependent noise estimation algorithm.

SCATTER - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$w = 8$.
$B \leftarrow$ all $w \times w$ block in U.
$M \leftarrow \text{Mean}(B)$. ▷ Obtain the mean of each $w \times w$ block.
$V \leftarrow \text{Var}(B)$. ▷ Obtain the variance of each $w \times w$ block.
Construct *scatter plot* of V vs. M^2.
Fit a straight line fitted to the main cluster, using the Hough transform.
Model the fitted line as $Y = mU + a$. ▷ m is the multiplicative and a the additive noise component.

Algorithm 7 Scatter white Gaussian noise estimation algorithm.

SCATTER_PEAK - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$w = 8$.
$k = 1.0229490535653729$.
$B \leftarrow$ all $w \times w$ block in U.
$M \leftarrow \text{Mean}(B)$. ▷ Obtain the mean of each $w \times w$ block
$V \leftarrow \text{Var}(B)$. ▷ Obtain the variance of each $w \times w$ block
$H = HISTOGRAM(V)$. ▷ Compute the histogram of V. Here, H is a list with the number of elements in each bin.
$m = \text{argmax}_i(H[i])$. ▷ Look for the bin with most elements (mode of the histogram).
$\hat{\sigma} = \text{Mean}[H(m)]$.
return $\tilde{\sigma} = k \times \hat{\sigma}$. ▷ Corrected value

These experiments show clearly that block variances are not at all reliable to estimate low noises. See how all images having textures lead to a strong over estimation of the noise. They therefore amply justify the use high pass filters to clean up the blocks from slow variations due to the signal and give prominence to noise components.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	13.52	13.50	14.43	16.87	23.75	52.54	82.71
building1	1.34	2.05	5.32	10.47	20.53	51.57	81.38
computer	1.54	2.32	5.31	10.65	20.63	51.58	83.29
dice	1.32	2.20	5.18	10.19	20.38	50.86	80.96
flowers2	1.29	2.30	5.26	10.51	20.53	51.18	80.14
hose	2.88	2.98	5.63	10.84	21.15	51.03	80.12
leaves	6.85	6.59	8.01	12.05	21.59	51.67	82.07
lawn	6.52	6.71	8.07	11.96	21.47	51.42	80.72
stairs	1.62	2.49	5.61	10.68	20.76	50.71	80.56
traffic	1.18	2.20	5.28	10.59	20.97	51.93	83.44
constant	1.01	2.00	4.97	9.94	20.05	50.83	80.36

TABLE 10. Scatter method results with simulated homoscedastic white Gaussian noise.

3.7. The mean of DCT high-frequency coefficients method. It is well known that the energy in high-frequency coefficients of a DCT image block corresponds mainly to the noise and edges, while the medium an low frequency coefficients contribute to the geometry of the image. This method computes the variance of the noise from the high-frequency coefficients.

Since the orthonormal DCT-II is a isometry, the variance can be computed both at the spatial domain or the frequential domain. However, computing it at the frequential domain has the advantage that it decorrelates the signal and compacts its energy on a few coefficients, thus making it possible to estimate the noise using only high-frequency DCT-II coefficients.

A correction factor $k = 1.0061$ is needed to correct the obtained standard deviation.

The estimated $\tilde{\sigma}^2$ is obtained as the mean of the variance of coefficient at frequencies $[6, 7]$, $[7, 6]$, and $[7, 7]$. To compute the variance, the unbiased estimator for the population variance is used: $S^2_{N-1}(X) = \frac{1}{N-1}\sum_{i=1}^{N}(X - \bar{X})$, where $\bar{X} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and N the number of elements in X. The pseudo-code for the DCT method is given in Algorithm 8 and the results for white Gaussian noise in Table 11.

Algorithm 8 DCT noise estimation algorithm.

DCT - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$w = 8$.
$k = 1.0061426171829178$. ▷ Correction factor
$B \leftarrow$ all $w \times w$ block in U.
$N =$ number of blocks in B.
$C = \text{2D-DCT}(B)$. ▷ Compute the 2D orthonormal DCT of each block in B
$C_1 \leftarrow C[6, 7]$. ▷ Obtain all coefficients at position $[6, 7]$
$C_2 \leftarrow C[7, 6]$. ▷ Obtain all coefficients at position $[7, 6]$
$C_3 \leftarrow C[7, 7]$. ▷ Obtain all coefficients at position $[7, 7]$
$\hat{\sigma} = \sqrt{\text{Mean}\left[\text{Var}(C_1), \text{Var}(C_2), \text{Var}(C_3)\right]}$.
return $\tilde{\sigma} = k \times \hat{\sigma}$.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	2.59	3.10	5.58	10.31	20.30	50.24	80.66
building1	3.78	4.18	6.24	10.71	20.40	50.40	80.56
computer	3.51	3.94	6.06	10.57	20.47	50.62	80.52
dice	1.24	2.14	5.10	10.03	20.25	50.23	80.53
flowers2	1.47	2.28	5.16	10.10	20.05	50.27	79.97
hose	1.42	2.23	5.15	10.12	20.12	50.29	81.18
leaves	2.80	3.27	5.64	10.40	20.32	50.04	81.12
lawn	3.06	3.53	5.79	10.44	20.37	50.35	80.24
stairs	1.46	2.28	5.15	10.15	20.11	50.18	80.90
traffic	4.84	5.13	6.90	11.01	20.54	50.63	80.93
constant	1.00	2.02	5.04	10.04	20.15	49.84	80.36

TABLE 11. The Mean of DCT high-frequency coefficients method results with uniform noise.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	1.90	2.71	5.47	10.23	20.20	50.10	80.30
building1	1.71	2.77	5.75	10.59	20.23	50.13	80.29
computer	1.72	2.68	5.60	10.38	20.39	50.44	80.15
dice	1.16	2.10	5.08	9.98	20.17	49.92	80.28
flowers2	1.19	2.14	5.12	10.04	19.98	50.08	79.74
hose	1.32	2.19	5.11	10.06	20.03	50.07	81.21
leaves	2.35	3.00	5.57	10.36	20.26	49.98	81.14
lawn	2.60	3.23	5.70	10.41	20.30	50.22	79.87
stairs	1.39	2.25	5.13	10.11	19.94	50.07	80.46
traffic	2.25	3.23	6.06	10.74	20.43	50.63	80.48
constant	1.00	2.01	5.00	10.05	19.95	49.49	79.58

TABLE 12. Mean of DCT high-frequency coefficients method with MAD estimator results with white Gaussian noise.

3.8. The Mean of DCT high-frequency coefficients method with MAD estimator. We know of no particular reference for this method, except that in view of the other ones, it is simply necessary to test it. The Median of Absolute Deviations (MAD) estimator **[61]** is a robust estimator of the standard deviation that can be used when the data present outliers. The MAD estimator of X is defined as $\text{MAD}(X) = \text{median}\left(|X - \text{median}(X)|\right)$. If X follows a normal distribution, MAD is a biased estimator of the standard deviation and must be multiplied by $k = 1.4865$ to get back $\tilde{\sigma}$. This method is essentially the same as in Section 3.7, but using the MAD estimator instead of directly computing the sampled variance. The pseudo-code for the DCT-MAD method is given in Algorithm 9 and the results for white Gaussian noise in Table 12.

Algorithm 9 DCT-MAD noise estimation algorithm.

DCT - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$w = 8$.
$k = 1.486495980422939$.
$B \leftarrow$ all $w \times w$ block in U.
$C = \text{2D-DCT}(B)$. ▷ Compute the 2D orthonormal DCT of each block in B
$C_1 \leftarrow C[6, 7]$. ▷ Obtain all coefficients at position $[6, 7]$
$C_2 \leftarrow C[7, 6]$. ▷ Obtain all coefficients at position $[7, 6]$
$C_3 \leftarrow C[6, 7]$. ▷ Obtain all coefficients at position $[7, 7]$
$\hat{\sigma} = \text{Mean}\left[\text{MAD}(C_1), \text{MAD}(C_2), \text{MAD}(C_3)\right]$.
return $\tilde{\sigma} = k \times \hat{\sigma}$.

3.9. Deconvolution method (E.I.N.V.). The Estimation of Image Noise Variance method [**5**] (E.I.N.V) first computes the finite differences derivative of the image, then makes a histogram of the local standard deviations of these high pass filtered values, and finally it evaluates the histogram iteratively in order to converge to the noise standard deviation estimate.

Step 1: suppression of the original image x

Given an image $y(m,n)$ corrupted with Gaussian noise $w(m,n)$, $y(m,n) = x(m,n)+w(m,n)\ \forall (m,n) \in$ the domain of the ideal image x:

First, the horizontal derivative is computed:

$$y_1(m,n) = \tfrac{1}{\sqrt{2}}\left[y(m+1,n) - y(m,n)\right]$$

Then, the vertical derivative of the horizontal derivative y_1:

$$y_2(m,n) = \tfrac{1}{\sqrt{2}}\left[y_1(m,n+1) - y_1(m,n)\right].$$

Step 2: computing the histogram of local standard deviations Using a window of size $L \times L$ pixels, with $L = 2K+1,\ K \in \mathbb{N}$, the variance is estimated by the classic unbiased estimator

$$\hat{\sigma}^2(m,n) = \tfrac{1}{N_L - 1}\left(\left[\sum_{i=-K}^{K}\sum_{j=-K}^{K} y_2^2(m+i,n+j)\right] - N_L\hat{\mu}^2(m,n)\right).$$

N_L is the number of pixels in the window (L^2).

The local mean is computed as:

$$\hat{\mu}(m,n) = \tfrac{1}{N_L}\sum_{i=-K}^{K}\sum_{j=-K}^{K} y_2(m+i,n+j).$$

The authors explain that they tested several different window sizes and found that the best results are achieved with the minimal possible window size, $L = 3$.

The histogram $h(k)$ with entries $\hat{\sigma}(m,n)$ is defined as:

$$h(k) = \begin{cases} \#\{(m,n) : k - \frac{1}{2} \le \alpha\hat{\sigma}(m,n) < k + \frac{1}{2}\},\ \text{if } k = 1,\dots,k_{\max}; \\ 2\#\{(m,n) : 0 \le \alpha\hat{\sigma}(m,n) < \frac{1}{2}\}|,\ \text{if } k = 0, \end{cases}$$

where "#" means the cardinal of the set.

The value $\alpha = 1$ was considered adequate by the authors for most of the images.

Step 3: Evaluation of the histogram

An averaging of the values $\hat{\sigma}^2(m,n)$ can be performed by computing the mean square value of the histogram,

$$s_l^2 = \frac{\sum_{k=0}^{k_{max}} k^2 h(k)}{\sum_{k=0}^{k_{max}} h(k)}.$$

The value s_l^2/α^2 is an initial global estimate for the noise variance σ_w^2.

The estimation can be improved iteratively thanks to the prior knowledge of the shape which the histogram should have. The histogram should exhibit a rapid descent for large values of k, but for a natural image, the descent is significantly slower. The reason they give for this fact is that the histogram is the convolution of a noisy uniform image with the histogram of the ideal image.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	3.20	3.82	6.45	11.12	20.65	50.35	80.59
building1	1.58	2.53	5.79	11.14	20.92	50.57	80.46
computer	2.02	2.87	5.99	11.09	21.05	50.76	80.68
dice	1.07	2.12	5.11	10.08	20.12	50.25	80.63
flowers2	1.12	2.15	5.16	10.15	20.17	50.36	80.33
hose	1.68	2.56	5.38	10.24	20.25	50.08	80.63
leaves	3.60	4.12	6.45	11.01	20.64	50.26	80.65
lawn	3.51	4.03	6.39	10.95	20.60	50.33	80.51
stairs	1.84	2.67	5.42	10.29	20.14	50.36	80.48
traffic	2.14	2.98	6.14	11.41	21.21	50.79	80.64
constant	1.00	2.00	5.04	10.00	20.03	50.28	80.40

TABLE 13. Method for the Estimation of Image Noise Variance (deconvolution method) results with white Gaussian noise.

Therefore, it is a deconvolution problem. The method fades-out the histogram by a descending weighting function $g(k)$. The following function is used:

$$g_l(k) = \begin{cases} 1 \text{ if } k \leq s_l \\ \frac{1}{2}\left[1 - \cos\left(\frac{\beta - \frac{k}{s_l}}{1-\beta}\right)\right] \text{ if } s_l < k < \beta s_l \\ 0 \text{ if } k \geq \beta s_l. \end{cases}$$

The value of β must be obtained experimentally. The authors of the method found that $\beta = 2.12$ is adequate. This value was used in all tests for this method. An improved value of the square mean s^2 can be obtained iteratively by

$$s^2_{l+1} = \frac{\sum_{k=0}^{k_{max}} k^2 g_l(k) h(k)}{\sum_{k=0}^{k_{max}} g_l(k) h(k)}.$$

A fixed value $l_{max} = 4$ is enough. The initial value s_l if taken from $s^2_l = \frac{\sum_{k=0}^{k_{max}} k^2 h(k)}{\sum_{k=0}^{k_{max}} h(k)}$. Finally, the estimate $\tilde{\sigma}^2 = \frac{s_{l_{max}}}{\alpha^2}$. It was found that a correction factor $F = 1.0141$ is needed to get an unbiased estimation. The pseudo-code for this method is given in Algorithm 10 and the results for white Gaussian noise in Table 13.

This is comparable to the other ones, in that it computes the average of a weighted quantile of the variance histogram. The chosen differential operator, the second cross derivative, is judicious. The results are bad for low noise. They show that either the high pass filter has not a large enough order, or the quantile is not low enough.

Algorithm 10 Estimation of Image Noise Variance (deconvolution method) algorithm.

EINV - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$F = 1.0140974819332935$. ▷ Correction factor

$y_1(m,n) = \frac{1}{\sqrt{2}}\left[y(m+1,n) - y(m,n)\right]$.

$y_2(m,n) = \frac{1}{\sqrt{2}}\left[y_1(m,n+1) - y_1(m,n)\right]$.

$\hat{\sigma}^2(m,n) = \frac{1}{N_L - 1}\left(\left[\sum_{i=-K}^{K}\sum_{j=-K}^{K} y_2^2(m+i,n+j)\right] - N_L\hat{\mu}^2(m,n)\right)$.

Create histogram $h(k)$. ▷ View step 2

$Q = \text{ones}(k_{max}+1)$. ▷ Array of k_{max} ones

for $i = 1 \ldots 4$ **do**

  N $= 0$. ▷ Numerator

  D $= 0$. ▷ Denominator

  for $k = 0, ...k_{max}$ **do**

    $N = N + k^2 Q(k)h(k)$

    $D = D + Q(k)h(k)$

  end for

  $s_l = \sqrt{N/D}$

  for $k = 0, \ldots, k_{max}$ **do**

    if $k < s_l$ **then**

      $g_l(k) = 1$

    end if

    if $s_l < k$ and $k < \beta s_l$ **then**

      $g_l(k) = 0.5\left[1 - \cos\left(\frac{\pi(\beta - k/s_l)}{1-\beta}\right)\right]$

    end if

    if $k \geq \beta s_l$ **then**

      $g_l(k) = 0$

    end if

  end for

  $Q = g_l$. ▷ Copy array

  $\tilde{\sigma} = (s_l/\alpha)F$

end for

return $\tilde{\sigma}$

$$L_1 = \begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array} \qquad L_2 = \frac{1}{2}\begin{array}{|c|c|c|}\hline 1 & 0 & 1 \\ \hline 0 & -4 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array}$$

FIGURE 7. Laplace-like operators.

$$N = 2(L_2 - L_1) = \begin{array}{|c|c|c|}\hline 1 & -2 & 1 \\ \hline -2 & 4 & -2 \\ \hline 1 & -2 & 1 \\ \hline \end{array}$$

FIGURE 8. Difference of Laplacian operators to get the noise estimation operator in the article *Fast Noise Variance Estimation* [**4**].

3.10. Method for Fast Noise Variance Estimation. The Fast Noise Variance Estimation method [**4**] tries to avoid the interference of image structures (edges and textures) present on the image when estimating the noise. To do it, it detects these structures using an operator based on the Laplacian, and cancels them. It considers two 3×3 Laplacian stencils L_1 and L_2 (Figure 7) and makes their difference to obtain the noise estimation operator $L = 2(L_2 - L_1)$ (Figure 8).

Operator L cancels edges and structures in the image, but it is a biased estimator of the variance. The correction factor to get an unbiased estimation of the variance is $\frac{1}{36}$. The pseudo-code for this method is given in Algorithm 11 and the results for white Gaussian noise in Table 14.

Algorithm 11 Fast Noise Variance Estimation algorithm.

FNVE - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

$L = [[1, -2, 1], [-2, 4, -2], [1, -2, 1]]$.
$V = 0$.
for $[x, y] \in [1, N_x - 2] \times [1, N_y - 2]$ **do**
    $A = 0$.
    for $(i, j) \in (-1, \ldots 1, -1 \ldots 1)$ **do**
        $A = A + U(x + i, y + i)L(1 + i, 1 + j)$.
    end for
    $V = V + A^2$.
end for
return $\tilde{\sigma} = \sqrt{\frac{V}{36(N_x-1)(N_y-1)}}$.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	3.61	4.00	6.09	10.60	20.29	49.94	80.14
building1	5.00	5.30	7.01	11.14	20.57	50.15	80.11
computer	5.21	5.50	7.17	11.21	20.66	50.42	80.10
dice	1.34	2.19	5.09	10.01	20.05	49.94	80.25
flowers2	1.79	2.49	5.21	10.09	20.00	50.11	79.81
hose	1.76	2.47	5.21	10.11	20.09	49.92	80.33
leaves	3.65	4.03	6.10	10.61	20.36	49.84	80.36
lawn	3.78	4.16	6.18	10.63	20.34	50.03	80.06
stairs	1.81	2.51	5.22	10.13	20.03	50.05	80.25
traffic	6.02	6.25	7.76	11.59	20.80	50.41	80.24
constant	1.00	2.00	5.02	9.96	19.92	50.04	80.22

TABLE 14. Fast Noise Variance Estimation method results with simulated homoscedastic white Gaussian noise.

Image / $\tilde{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	2.37	3.12	5.78	10.48	20.03	49.81	79.58
building1	1.77	2.85	5.91	10.71	20.39	49.85	79.52
computer	1.99	2.97	5.92	10.79	20.28	49.74	79.90
dice	1.11	2.08	5.03	9.96	19.80	49.81	78.97
flowers2	1.16	2.11	5.05	9.98	20.07	49.33	79.08
hose	1.47	2.33	5.17	10.06	19.85	49.67	79.27
leaves	2.89	3.50	5.86	10.44	20.08	49.64	79.76
lawn	3.06	3.68	6.07	10.61	20.19	49.64	79.43
stairs	1.77	2.57	5.31	10.16	20.03	49.13	79.68
traffic	2.25	3.29	6.24	11.01	20.67	50.22	79.66
Flat image	0.99	1.96	4.92	9.92	19.63	48.85	78.59

TABLE 15. Wavelet DB3 MAD results with white Gaussian noise.

3.11. Wavelet MAD. This method consists on applying the MAD estimator to the detail (HH) coefficients of the Discrete Wavelet Transform (DWT) of the image.

The algorithmic description for this method can be found in Algorithm 12 and its results for white Gaussian noise in Table 15. The DB3 Daubechies wavelet were used.

Algorithm 12 Wavelet-MAD noise estimation algorithm.

1: **Wavelet-MAD** - Return the standard deviation of the image noise. **Input U**: input image. **Output** $\tilde{\sigma}$: estimated standard deviation of the noise in the image.

2: $w = 8$.

3: Obtain multiresolution wavelet coefficients LL, HL, LH, HH of U.

4: $M = \text{median(HH)}$.

return $\tilde{\sigma} = \frac{\text{median}(|\text{HH} - M|)}{0.6745}$

3.12. Conclusion. In this chapter we presented a review of classic homoscedastic noise estimation methods.

Table 16 gives the RMSE values obtained by evaluating these methods with white Gaussian noise, with $\sigma \in \{1, 2, 5, 10, 20, 50, 80\}$ along all the images. This table is the final result of this study of homoscedastic noise estimation methods, and is particularly decisive for the low noise values. Giving the RMSE depending on σ for white Gaussian noise is perhaps not quite intuitive, and requires some attention. For example, a RMSE of 1.0 is an excellent estimate for $\sigma = 80$, but is quite bad if $\sigma = 2$. Thus this last table shows that the results are very bad for all methods but the Ponomarenko et al. and the Percentile methods. Still the error is about 8% for low level noise $\sigma = 2, 5$, but it only doubles for much larger noises, which means that it is ridiculously low for high noise. These precisions on the noise estimation are more than enough for applying denoising algorithms.

Method	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
PCA [**3**]	**0.31**	**0.21**	**0.13**	**0.25**	0.37	1.17	2.57
DCT-MAD	0.59	0.85	0.48	0.29	0.12	0.38	0.71
DCT	0.99	2.21	0.71	0.31	0.19	0.46	0.74
Ponomarenko et al. [**15, 16**]	0.56	0.39	0.28	0.34	0.49	1.38	1.60
Wavelet MAD	0.91	0.91	0.73	0.43	**0.05**	**0.25**	0.83
F.N.V.E. [**4**]	1.64	3.12	1.40	0.45	0.17	0.26	**0.25**
Percentile [**17**]	0.58	0.52	0.51	0.47	0.67	1.22	2.07
E.I.N.V. [**5**]	1.36	0.72	1.01	0.67	0.39	0.56	0.59
Block [**56, 57**]	1.47	0.41	0.94	1.05	1.27	1.00	1.05
Gradient [**58, 59**]	2.44	0.96	1.45	1.30	0.54	1.17	1.76
Scatter [**60**]	7.24	0.58	1.75	1.39	2.23	1.56	2.26
Average [**55**]	3.02	2.15	2.31	1.64	1.43	1.63	0.99
Median [**55**]	3.26	2.35	2.54	1.83	1.81	2.17	1.61

TABLE 16. Comparison of the averaged RMSE along all test images in Figure 3 using simulated homoscedastic white Gaussian noise.

In table 16 they appear three additional methods: the Ponomarenko et al., the Percentile, and the PCA methods. These methods have an special interest because their performance is clearly superior to that of the classic method (the case of the Ponomarenko et al. and the Percentile methods), or because they are relatively recent (the PCA method). These methods have been studied with great detail in the third part of the thesis. The Ponomarenko et al. method is detailed in Chapter 10, the Percentile method in Chapter 11, and the PCA method in Chapter 12, and evaluated in Chapter 13.

The method Estimation of Image Noise Variance is referred to as *E.I.N.V.* [**5**] and the method Fast Noise Variance Estimation as *F.N.V.E* [**4**]. The Pyramid method [**6**] is not evaluated here, because it was not able to give a estimate for all σ levels. The Ponomarenko et al. method was configured with $p = 0.005$ and $w = 8$.

It can be concluded that for simulated homoscedastic white Gaussian noise, the PCA methods gives better results than its competitors, for low and moderated noises ($\sigma < 10$). For $\sigma > 10$, simpler methods, as F.N.V.E and the Wavelet MAD methods give the best results.

It is important to note that this table only shows the preliminary results of the study presented in this thesis, where the noise is homoscedastic, white, and simulated. As it will be explained in Chapter 2, simulating this kind is simple noise is not realistic at all, since real images are (at least) signal-dependent, and thus the variance of the noise increases with the intensity. Adapting the method to signal-dependent noise implies changes that make this classification change significantly. In fact, the classic method give inaccurate results when adapted, whereas the Ponomarenko et al. method gives the best results, followed by Percentile and PCA.

In Chapter 3 it will be shown than the signal-dependent noise is not enough (for example, it does not apply to digital images after demosaicing the raw image at the focal plane). In Chapter 13 it will be shown that after adapting the methods to signal-dependent noise, the PCA method is not reliable when the image contains high noise, and needs much more samples/bin than the Ponomarenko et. al and Percentile methods, which perform much accurately.

CHAPTER 2

Signal-dependent noise estimation

Optimal denoising works at best on raw images (the image formed at the output of the focal plane, at the CCD or CMOS detector), which display a white signal-dependent noise. The noise model of the raw image is characterized by a function that given the intensity of a pixel in the noisy image returns the corresponding STD; the plot of this function is the noise curve. *This chapter develops a non-parametric approach estimating the noise curve directly from a single raw image. An extensive cross-validation procedure is described to compare this new method with state-of-the-art parametric methods and with laboratory calibration methods giving a reliable ground-truth, even for nonlinear detectors.*

The signal-dependent noise model is valid for raw images, but when the noise is correlated and thus frequency-dependent (for example, after demosaicing the raw image), the noise model presented here is not enough, as will be shown in Chapter 3. Chapter 4 will discuss a new model able to measure the noise even in JPEG-encoded images.

1. Introduction

Most denoising methods assume that the noise in the image is additive, homoscedastic, white, and Gaussian. *Homoscedastic* means that the variance of the Gaussian noise is fixed and does not depend on the pixel position or value. By "white" noise, we mean that the noise pixel values are independent (look at Section 3 of Chapter 1 for a review of classic homoscedastic white noise estimators). We shall retain this terminology throughout.

The homoscedasticity assumption is not realistic. The photon emission by a body follows a Poisson distribution which can be approximated by a Gaussian distribution when the number of photons is large enough. But the variance of this Gaussian is signal dependent. In the Poisson model **[62, 63, 64, 65, 66, 67, 68]**, an image value $\tilde{\mathbf{U}}(x, y)$ at pixel (x, y) is a Poisson variable with variance and mean equal to $\mathbf{U}(x, y)$, where $\mathbf{U}$ is the ideal noise-free image. The Poisson noise has therefore a standard deviation (STD) equal to $[\mathbf{U}(x, y)]^{1/2}$. Thus, an ideal raw image is a white Poisson noise whose mean at each pixel is the noiseless value. Note that this is related to the quantum nature of light and the probability of emitting a photon, independently of the technology used at the CFA (CCD, CMOS). This Poisson noise adds up to a thermal noise and to an electronic noise which are approximately additive and white, making the final noise model not necessarily Poisson distributed, but still white and signal dependent.

Noise estimation is a necessary preliminary step for most image processing and computer vision algorithms **[1]**. Nevertheless, several other denoising methods propose to deal directly with

Poisson noise. Wavelet-based denoising methods [**45, 44**] propose to adapt the transform threshold to the local noise level of the Poisson process. Lefkimmiatis et al. [**46**] have explored a Bayesian approach and Deledalle et al. [**47**] have adapted the Non-local means algorithm [**32**] to Poisson noise. These papers assume that no variance stabilizing transform (VST) transforming the signal dependent noise into a nearly homoscedastic noise is accurate enough to transform the Poisson noise into homoscedastic noise. The advantage of VSTs is that they permit the application of a classic denoising algorithm. The VST associated with Poisson noise is often called Anscombe transform [**42**], but one can attach a VST to any signal dependent noise model [**1**]. As a matter of fact, papers on the Anscombe transform [**43**] (for low count Poisson noise) and [**50**] (for Rician noise) argue that, when combined with suitable forward and inverse variance stabilizing transformations, algorithms designed for signal independent Gaussian noise work just as well as ad-hoc algorithms for Poisson noise models. These considerations confirm the importance of estimating as accurately as possible the noise curves of raw images, since their accurate knowledge is required to compute the VST. In most CCDs and CMOS detectors, the variance of the noise at a pixel is approximated (assuming that all detectors at the CFA are equivalent and thus neglecting the fixed pattern noise) by a simple linear model $\sigma^2 = A + B\mathbf{U}$, where $\mathbf{U}$ is the expectation of the intensity of this pixel in the noisy image. This model is valid under the assumption mentioned above of a combination of a Poisson with a thermal noise. Yet, this assumption holds only if the signal is not saturated and the photon count large enough. At the darkest pixels, the Poisson distribution of the noise cannot be approximated by a Gaussian and it becomes a *shot* noise. In short, the noise variance does not necessarily follow the linear model in the darkest and brightest image regions. An accurate estimation of the noise at the darkest zones is crucial since subsequent processes in the camera chain (specially, the gamma correction step) are designed to increase the dynamics in the dark zones. If the noise is not removed at the raw image stage, it might end up really augmented at the final stage.

Parametric noise estimation methods try to obtain the parameters that control a noise model (for example, the A and B parameters of the linear model). Yet, to get a realistic estimation, they have to take into account the effect of the saturation in the darkest and brightest pixels in the final noise curve. To validate the estimation of a noise estimation method, its noise curve must be compared to a ground-truth curve. Such a ground-truth for a particular camera and settings can be obtained by taking a series of photographs of a pattern, that is mostly flat and contains a wide range of gray levels. The temporal variation of the gray level at a given pixel gives an estimate of the noise STD associated with this gray level. However, the series of photographs must be taken under controlled conditions, to ensure that any variation of the intensity of a pixel can be only explained by the noise. In short, it is a heavy procedure (that is, it requires constant lighting, a camera stabilizer to fix its position, and isolation from any kind of electromagnetic source that may introduce electronic noise into the camera) which also needs access to the camera that took the photographs. It also requires the *a priori* knowledge of the form of the camera noise model, which is not granted. This explains why the establishment of a method able to estimate automatically the noise model from a single snapshot is a valid question. Furthermore, if the method can be shown

to be reliable even without any *a priori* model guess, its credibility will be somewhat augmented. In this chapter, we show that it is indeed possible to use a non-parametric estimator to get an accurate noise curve from the noisy image itself, by measuring the variance locally with patch-based methods [**69, 70, 71, 3, 56, 72**]. This eliminates the need for lab calibration procedure. Indeed, the procedure described uses one or several photographs taken in arbitrary environment and yields a non-parametric noise model as good (for those images) as the one obtained by the heavier ground truth procedure (laboratory calibration). We also examine the question of whether it is better to use a parametric or a non parametric model when dealing with a single or a few photographs. Our conclusion is that the non parametric method gives results comparable to the parametric method, but is somewhat less risky as it does not propagate local estimation errors caused by the presence of texture in the image.

Our plan follows from the above discussion. Since noise estimation is a well-known procedure for white homoscedastic noise, Section 2 will review the literature on white homoscedastic noise estimation and will point out competitive algorithms. Section 3 explains the procedure that should be followed to get a reliable non-parametric noise curve from a series of images, under controlled conditions. Section 4 discusses how homoscedastic white noise estimation algorithms can be adapted to estimate an arbitrary signal-dependent noise curve. Section 5 compares the Root Mean Squared Errors (RMSE) between the non-parametric ground-truth, the STDs from the series of images and two state of the art parametric methods. Finally, Section 6 presents the conclusions, that validate our proposed nonparametric method, but also the use of two state-of-the-art parametric methods.

2. State-of-the-art in white noise estimation

Many noise estimation methods share the following features, which can be summarized in two sentences:

- estimate noise in high frequencies, where noise dominates over signal;
- estimate noise in image regions with the least variation, typically the blocks with the smallest STDs.

Thus, these numerous methods [**4, 5, 6, 16, 56, 72, 55, 57, 60, 58, 59, 73, 74**] proceed roughly as follows:

- they start by applying some high-pass filter, which concentrates the image energy on its edges, while the noise remains spatially homogeneous;
- they compute the energy of many blocks extracted from this high-passed image;
- they estimate the STDs of the blocks;
- to avoid blocks whose STD is mostly explained by the underlying ideal image, a statistic robust to (many) outliers must be applied. The methods therefore prefer the flattest blocks, which belong to a (low) percentile of the STDs of all the blocks.

Note that the power spectral density of a natural image is not homogeneous. Most of the energy corresponding to its geometry is located at the low and medium frequencies, whereas high

frequency coefficients bring little visual information (with the exception of the edges). Conversely, an image can be considered “highly textured" if the energy at the high-frequency coefficients is as high as the energy observed at edges. Thus, high-passing the image before estimating the noise spatially (or equivalently, estimating the noise only at the high-frequency DCT coefficients) is an initial step for many noise estimation algorithms. This enhances the contribution of the noise. Yet, avoiding edges and textures in the estimation remains necessary.

We shall limit ourselves to discuss the method acknowledged as the best estimator for homoscedastic noise in the review [**1**], the Ponomarenko et al. method [**16**], along with the two of the most competitive parametric methods for noise estimation in raw images [**7, 75**]. We briefly describe these competitors in the next paragraphs. For a complete review on noise estimation methods, we refer the reader to [**1**] and [**55**].

2.0.0.1. The Ponomarenko et al. approach. The Ponomarenko et al. [**16**] method is an extension of the previous method [**72**], based on the analysis of the DCT coefficients. In short, the Ponomarenko et al. method computes the variance of the high-frequency coefficients of a set of blocks whose variance measured at the low frequencies is minimal. We refer the reader to Chapter 10, where this method is analyzed in deep detail.

We now discuss two parametric methods that will be compared here.

2.0.0.2. Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-image Raw-data. Foi et al. proposed a simple parametric noise model [**7**] that takes into account the non-linear response of the CCD due to the saturation of the signal and noise at the darkest and brightest pixels of the image. The model assumes the well-known normal approximation, for which the Poisson distribution $\mathcal{P}(\lambda)$ of the noise can in practice be approximated by the normal distribution: $\mathcal{P}(\lambda) \to \mathcal{N}(\mu = \lambda, \sigma = \lambda)$. The method has two stages. In the first step it estimates several pairs intensity/STD that form a scatter plot. In the second step, the observed pairs are used to fit a global parametric model. Before applying these two steps, the image is preprocessed. First, the 2D-wavelet transform of the image is computed and the wavelet detail coefficients stored. The 1D Daubechies wavelet and scaling functions are used to create the 2D kernels of the transform. The STD of the noise is obtained from the detail coefficients of the transformed signal. In order to be robust against edges, the image is segmented into level sets according to the intensity. Since the image to be segmented is noisy, the segmentation is done in a low-pass filtered version. With the selected regions of the image, the intensity of each pair is obtained as the sample mean of the approximation wavelet coefficients and the estimated variance with the unbiased sample variance estimator. The last step of the method is to fit the A and B parameters of the linear model of the variance, for which a maximum-likelihood (ML) fitting is performed. However, since saturation makes the response of the CCD or CMOS detector non-linear, the method needs to modify the expectation and variance estimators to take saturation into account. The authors calculated the new estimators from the distribution of the non-saturated signal and gave the explicit expression for the expectation and variance estimators under saturation. Finally, these new pairs are incorporated into the ML fitting in order to get the A and B parameters of the linear model despite the presence of saturation. The model is able to predict the non-linear response of the CCD or CMOS detector

under saturation, giving explicitly the variance of the clipped noise for any intensity. Therefore, this method will be used as an example where parametric and nonparametric methods are cross-validated (Section 5).

2.0.0.3. Image Informative Maps for Component-wise Estimating Parameters of Signal-dependent Noise. In the paper [**75**] Uss et al. propose to adapt the use of disjoint informative maps [**76**] to estimate a parametric signal-dependent Poisson-like noise model. It discriminates between two kinds of non-overlapping blocks (SW – scanning window): those which belong to textures (TI – texture informative) and those that are suitable for noise estimation (NI – noise informative). To describe the textures of a given SW in the image, the 2D fractal Brownian motion (fBm) model is used, since the model is able to characterize a texture with few parameters. The roughness of the texture is obtained from the Hurst exponent in the fBm model. The estimation of the noise is obtained from a limited set of high-frequency coefficients of the DCT transform of the SWs that belong to the NI map. This idea was introduced in the Ponomarenko et al. method [**16**] and stated as the state of the art technique for noise estimation [**1**]. The Cramér-Rao Lower Bound (CRLB) is used to decide if a SW belongs to the TI or NI maps, on the texture parameters and the noise STD of the SW. All the SWs in the image are sorted according to increasing CRLB and then compared to a threshold. The SW below the threshold have the lowest CRLB and therefore belong to the NI map. The rest are assumed to be textures and assigned to the TI map. Since the criterion based on the CRLB relies on the (unknown) texture and noise parameters, the method begins with an initial guess for the NI and TI maps by fixing a noise STD and texture level to have an initial and rough CRLB criterion. Then, with the available CRLB criterion better STD and texture levels are computed, allowing for an even better CRLB criterion. The refining loop is iterated until convergence is reached. To estimate signal-dependent noise, the set of SW is partitioned into disjoint intensity sets according to their mean intensity and the method is applied separately to each set in order to get an (intensity–STD) pair. Therefore, this method is coherent with the claim we make in Section 4, which states that any block-based homoscedastic noise estimation method can be easily adapted to deal with signal-dependent noise, just by splitting the whole set of image blocks of the image into disjoint in intensity sets to apply then the homoscedastic version of the method to each of these sets. For example, if the input image has size $N_x \times N_y$, there are $M = (N_x - w + 1)(N_x - w - 1)$ overlapping blocks, that may be distributed into a set of M/k bins, where each bin contains k image blocks/bin whose mean intensity is a part of complete intensity range of the image.

3. Non-parametric noise ground-truth curve

Parametric methods fix an *a priori* model for the noise. For example, at the output of the CCD or CMOS detector a good approximation of the Poisson noise is to use the normal distribution approximation, at least when the number of incoming photons is large enough. Therefore, the variance of the noise is equal to the expectation. The noise at the output of the CCD or CMOS detector is Poissonian and therefore its variance is linear with the intensity. Also, thermal and electronic noise are added, and the noisy signal is amplified afterwards. Thus, the variance of the

noise can be modeled as a function of the intensity of the ideal (noise-free) image: $\sigma^2(\mathbf{U}) = A+B\mathbf{U}$. However, since the dynamics of the digital output from the CCD or CMOS detector is limited, the darkest and brightest pixels of the image can get saturated because of the noise, that becomes clipped noise. Because of the saturation, the probability distribution of the noise is no longer a symmetric normal distribution, but a truncated version with different statistics. The variance of the truncated distribution does not coincide with that of the normal distribution. Therefore, any realistic parametric estimation method must take into account that under saturation the simpler linear model is no longer valid. Some methods [**7**] adapt their expectation and variance estimators in order to take into account the effect of the saturation before fitting the linear function, while others [**75**] try to fit with polynomials of higher order or transform the image in such a way that the linear model holds.

In any case, the parametric model has to be validated in order to ensure that the curve they provide is indeed a function that accurately relates the intensity of the ideal image with the STD of the added noise. To do it, the estimations of the parametric method must be compared with a ground-truth noise curve. For the construction of the ground-truth the constraint of using just a single image is not needed. Indeed, it can be built from a series of snapshots of a calibration pattern taken from a camera in fixed position. The series must be taken under controlled laboratory conditions that ensure that the temperature and lighting remain constant. Ideally, any two images of the series should be exactly equal in absence of noise. Therefore, any variation between the images is only explained by stochastic light fluctuations (photon noise and shot noise) and the noise generated by the camera itself (dark noise, readout noise and electronic noise). In Section 1.2 of Chapter 3 it is explained in detail how to build the ground-truth curve for a particular camera and ISO speed. The ground-truth noise curves of the Canon EOS 30D and Nikon D80 for ISO speeds 1250 and 1600 are shown.

If $\tilde{\mathbf{U}}_i(x,y)$ is a pixel of the noisy image i at position (x,y), the intensity of the ideal image can be approximated by its empirical expectation $\hat{\mu}(x,y) = \mathbf{E}\left[\{\tilde{\mathbf{U}}_i(x,y)\}\right]$ for $i = 1,\ldots,N$, where N is the number of snapshots in the series. The empirical variance associated to intensity $\hat{\mu}(x,y)$ is $\hat{\sigma}^2(x,y) = \mathrm{Var}\left[\{\tilde{\mathbf{U}}_i(x,y)\}\right]$.

The calibration pattern must be mostly flat and represent a wide range of gray levels. Since the noise curve mainly depends on the ISO sensitivity, a different noise curve is estimated for each ISO level. Series of different exposure times were taken for each ISO in order to get representative information in the whole gray level range. The noise curves for different times of exposure were combined to obtain a single curve. In order to get a ground-truth noise curve, for each exposure time $(1/30s, 1/250s, 1/400s, 1/640s)$ about two hundred pictures of the calibration pattern were taken. Since each 2×2 block of the CFA (Color Filter Array) contains one sample of the red channel, two samples of the green channel and one sample of the blue channel, the raw image was resampled as an image with four different color channels of half width and height. Thus, four different noise ground-truth curves were obtained from the series, each corresponding to one of the four channels of the CFA. By splitting the color range into *bins* (disjoint in intensity intervals) and

computing the median value of the STDs at each bin, a ground-truth is obtained for the camera noise curve given the ISO and exposure times.

Figure 1 shows the noise curves obtained with a Nikon D80 camera with fixed ISO sensitivity of 1250 and 1600 and four exposure times, $t \in \{1/30s, 1/250s, 1/400s, 1/640s\}$. The obtained curves overlap perfectly. Each one treats a different color interval, thus permitting to fuse them into a single noise curve. This fused curve can be observed in the same figure. For each color value, the fused noise estimation is obtained by the median of the available estimations obtained for the different exposure time. The value for each curve is linearly interpolated using the two closest neighbors. Since the noise curve does not depend on the exposure time, these curves overlap (hence the double values). However, this overlap is not perfect because the STD is computed with a finite number of samples and therefore the estimation has some variance that causes a small error centered at the theoretical value. Curve (b) is the mean of all four curves at different exposure times, which cancels their variation around the theoretical value and therefore it can be used finally as a ground-truth for evaluating noise estimation algorithms. Figure 5 displays the approximation of the computed ground-truth values by a linear model with the Nikon D80 camera. Because of the saturation at the darkest zones, the estimated noise in the dark gray level does not follow a linear model. However, using the partial linear model splitting the curve into three parts might be useful to model this kind of curves is the noise model is known in advance. The ground-truth curve obtained with the procedure presented here describes accurately the characteristics of the noise without depending only on the minimal assumption that the noise depends only on the intensity. Parametric methods assume priors for a particular noise model and therefore their accuracy depends on how realistic these assumptions are. Section 4 shows that it is possible to get a reliable noise curve that matches with negligible error the non-parametric ground-truth and Section 5 shows that indeed it is possible to validate parametric methods with the non-parametric ground-truth curve.

4. Non-parametric signal-dependent noise estimation

Parametric models are accurate under the condition of prior knowledge about the noise model. For example, the Foi et al. [**7**] method assumes the linear model $\sigma^2 = A + B\mathbf{U}$ for the variance, but with a saturation effect. On the other hand, Uss et al. showed that the measured noise variance cannot always be fitted with a linear function, but with a polynomial of at least second order [**75**]. However, we were unable to fit a 2nd, 3rd or 4th order polynomial to the saturated noise curve in Figure 5. In order to use a linear function, these authors modify the intensity of the pixels at each SW of the image by a function that nullifies quadratic and higher terms of the noise variance model. After this transformation, the estimation is accurate.

Parametric methods require a validation, by a comparison to ground truth noise curves. The data in the ground-truth must be empirical, in the sense that it does not assume any prior (with the exception that the variance of the noise is a function of the expectation) and simply measures the variance of the noise *as-is*. As discussed in Section 3, the major problem of the comparison

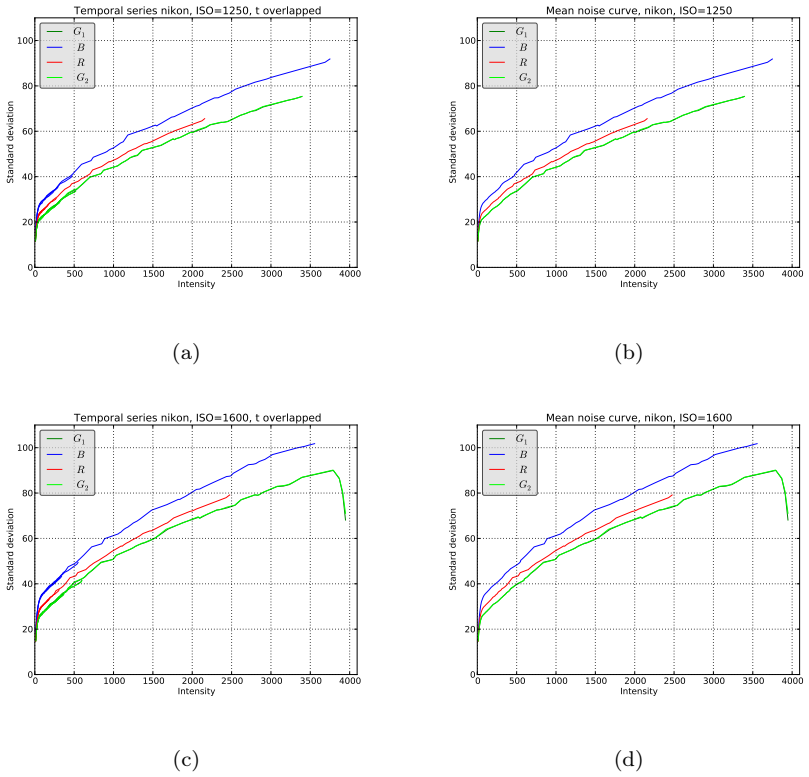


FIGURE 1. Noise ground-truth curves obtained for a Nikon D80 camera with fixed ISOs of 1250 (a) and 1600 (c) and four exposure times, $t \in \{1/30s, 1/250s, 1/400s, 1/640s\}$ using laboratory calibration. Channels G_1 and G_2 give the same STD. The obtained curves overlap perfectly. Since they cover different color intervals, their fusion yields a complete noise curve (b), (d).

against the ground-truth is that it is different for each camera model and it must be obtained under controlled laboratory conditions.

Our goal here is precisely to show that the laboratory calibration method used to obtain the ground truth can be replaced by a non-parametric method, estimating directly on the image the signal-dependent noise. We adapted the Ponomarenko et al. method [**16**], since it is scored as the best method in a previous review [**1**]. Other non-parametric estimation methods could be used as well. For example, in the paper of Liu et al. [**52**] a flexible eigenfunction representation of the noise level curves was proposed, but it needs a prior segmentation of the noisy image.

We extended the Ponomarenko et al. method [**16**] to be able to estimate signal-dependent noise (see the details in Chapter 10). To this aim, the means of the blocks are classified into a disjoint union of variable intervals (bins), in such a way that each interval contains a fixed and large enough number of elements. Thus, these intervals are automatically adapted to the image itself instead of prefixed, since the intensity range of each bin depends on the mean intensity of the $w \times w$ blocks in the image. We found that each bin should contain at least 42000 samples,

which seems to be the lowest number permitting a reliable estimation. The value of p for the p-quantile of block variances must be small to avoid blocks with large variance, corresponding to edges and textures. In general, if a bin is made of blocks that belong to a flat or smooth zone, we found experimentally that 210 per bin are enough to estimate the variance (using $p = 0.5$, the median). However, with a smaller $p = 0.005$ percentile value, we can discard 99.5% of the blocks with a higher variance and therefore in general all of the blocks affected by edges and textures. By choosing a large value for the bin cardinality, namely 42000, we ensure that the blocks below this low quantile are still numerous enough: $42000 \times 0.005 = 210$, so that they ensure a reliable estimate of the noise variance.

To each bin a list of image blocks is associated, each of them being endowed with a list of STDs. Notice that a bin does not correspond to a spatial region of the image, but only to a set of blocks with similar means.

Another modification with respect to the original method is the procedure to find the best p-quantile. The values in the list $\{\mathbf{V}^H(i,j)\}$ depend on the choice of p-quantile. If p is small, the method becomes more robust to the influence of textures and geometry of the image; but the accuracy of the estimation also decreases with p. Our assumption is that the variance measured using $p = 0.5$ (the median) should not be significantly different to the variance obtained with lower values of p, unless the image is composed mainly of textures. The proposed iteration to get a robust estimation of the variance, adapting p, is as follows. At each bin,

(1) Initially, $p = 0.5$ (the median) and $\Delta p = 0.005$.
(2) Set $S = \{\mathbf{V}^H(i,j) \mid i + j \geq T\}$.
(3) Set V_s to the median of the values of S under the Δp-quantile.
(4) Set V_p to the median of the values of S under the p-quantile.
(5) If $p \geq \Delta p$ and $V_p \leq V_s$ then [Set $p = p - \Delta p$ and go to step 3] else END.

This procedure decreases the initial p from the median to a lower value that makes the estimation robust to textures and geometry, if needed.

About the $w \times w$ size of the scanning window, we use the same value that the authors of the original algorithm proposed, and we found that indeed the best results are obtained with $w = 8$ in most natural images. Since the optimal size of the window depends on the density of edges and level of texturization of the noisy image, if it is *a priori* known that the image is mainly composed by large flat or smooth areas, it is better to use a larger window (up to 21×21) and to choose a smaller size in the opposite case (but at least 3×3) to obtain a reliable variance noise estimation. A larger window estimates the noise more accurately (since the sample variance estimator has itself a variance that depends on the number of available samples), but is less likely to contain only data from flat or smooth zones, and more likely to capture edges and geometry. Nevertheless, the proposed method is "blind", in the sense that no prior information about the characteristics of the image or the noise is available. Adapting the window size is beyond the scope of this chapter and it is left as future work.

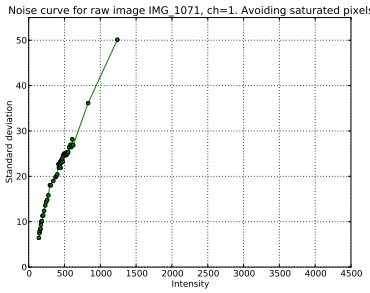
To avoid outliers in the estimation, we systematically discard completely saturated blocks. Indeed, when the number of photons counted by the CCD or CMOS detector during the exposure time is too high, its output may get saturated, and therefore underestimated. When the signal saturates the output of the CCD or CMOS detector, the measured variance in the saturated areas of the image is zero. Indeed, the effect of the saturation must be measured and given by the non-parametric method in the produced noise curve, but the completely saturated pixels have outlier intensities. Figure 2 shows a noise curve obtained by using or avoiding the saturated blocks, where the modified Ponomarenko et al. [**16**] algorithm was performed with 49 bins. Since the intensity of the saturated pixels is much higher (outlier) than the intensity of the rest of the points, the noise curve is linearly interpolated along the gap in between. This natural image is a normal scene that is useful to illustrate the problem of the saturation. The bike is not illuminated directly by any source of light (only ambient light) and therefore it does not reflect much light, with the exception of a few points at the handlebar that reflect light with enough power to saturate the detectors. The STD equal to zero (measured near intensity 4000) is indeed correct, but all the interpolated points in between are definitely not. The strategy we adopted was to discard the blocks that contain a sub-group of 2×2 pixels sharing the same intensity, in any of the channels. It must be noted that this only removes the blocks containing pixels that are completely saturated, but keeps the rest of the blocks, including those where the noise distribution is truncated, but not absolutely saturated. This permits to measure and observe the saturation in the curve, as shown in Figure 5.

5. Cross-validation of several methods. Discussion

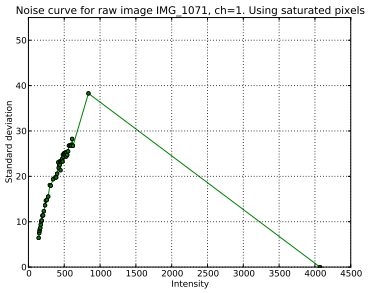
In order to compare ground-truth, parametric methods and our non-parametric method, we used a dataset of 20 images obtained with a Nikon D80 camera using ISO 1250 and exposure time $1/640s$. In these images the darkest pixels are saturated and therefore the noise curve does not follow the linear variance model. Our dataset contains some views of a room with objects over a table, images of corridors, bookshelves, (Figure 5) stairs, and classrooms inside a building, with different lighting levels. Also, two outdoor images of highly textured images (Figure 4). For each test image, we computed the RMSE between the STDs given by the method and by the ground-truth. The control points are given by the method and the STD of the ground-truth corresponding to that intensity is obtained by linear interpolation between the two nearest intensity control points of the ground-truth curve.

Figure 3 shows the obtained results. In general, the RMSE of the modified Ponomarenko et al. (red curve) method is close to zero, which means that it could be used to establish a (non-parametric) ground truth. The estimations given by Foi et al. (green curve) and Uss et al. (blue curve) are really close to the non-parametric ground-truth, and are therefore also validated by our approach. The Foi et al. method failed to measure the noise correctly when the images are composed mainly by textures (images #19 and #20, see Figure 4), whereas the Uss et al. and the proposed method gave good results in that case. Note that textures cause a localized error in the non-parametric curve (middle), whereas they cause a global error in the parametric curves. Fig 5 shows three examples of images in our dataset (images #8 and #12) where all algorithms

(a)



(b)

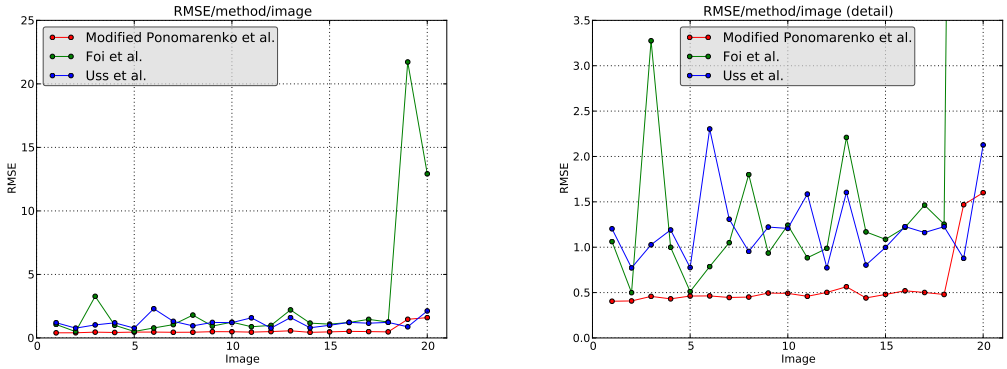


(c)

FIGURE 2. Noise curve obtained when the saturated pixels are avoided in the noise estimation (b) and when they are taken into account (c) by using the Ponomarenko et al. method **[16]** with 49 bins in an image with saturated pixels (a).

estimated the noise correctly. Foi et al. method (red curve) matches accurately the ground-truth curves (green and blue), since it is designed to predict the shape of the curve under saturation conditions, whereas Uss et al. estimation is overall correct, except in the saturation zone, as expected. As explained in Section 2, the original Ponomarenko method is only able to estimate homoscedastic noise, that is, a value of STD that does not depend on the intensity. However, in Figs. 4 and 5 we show noise curves, that correspond to the modified Ponomarenko method: added bins to get control points in the curve for different intensities and avoid using completely saturated points before the estimation. Of course, the over-estimation caused by a bin where all samples belong to textures can be avoided if more than a single image is available, by estimating the noise in the mosaic made of several different input images.

As shown in Figure 5, the linear model does not hold when the image is saturated. Uss et al. tried to use a second-order polynomial to fit the saturated noise curve. However, we found that a second-order polynomial was not general enough to fit the saturated curves. Foi et al. assumed the linear model, but taking into account the effect of the saturation. However, both methods assume that the noise can be modeled with a linear function when there is no saturation. This is true for most CCDs or CMOS detectors, but the output recorded in the raw file given by the camera might not be a linear function of the intensity **[77]**. This makes clear the necessity of validating parametric methods, which assume an *a priori* model for the noise. In contrast, the



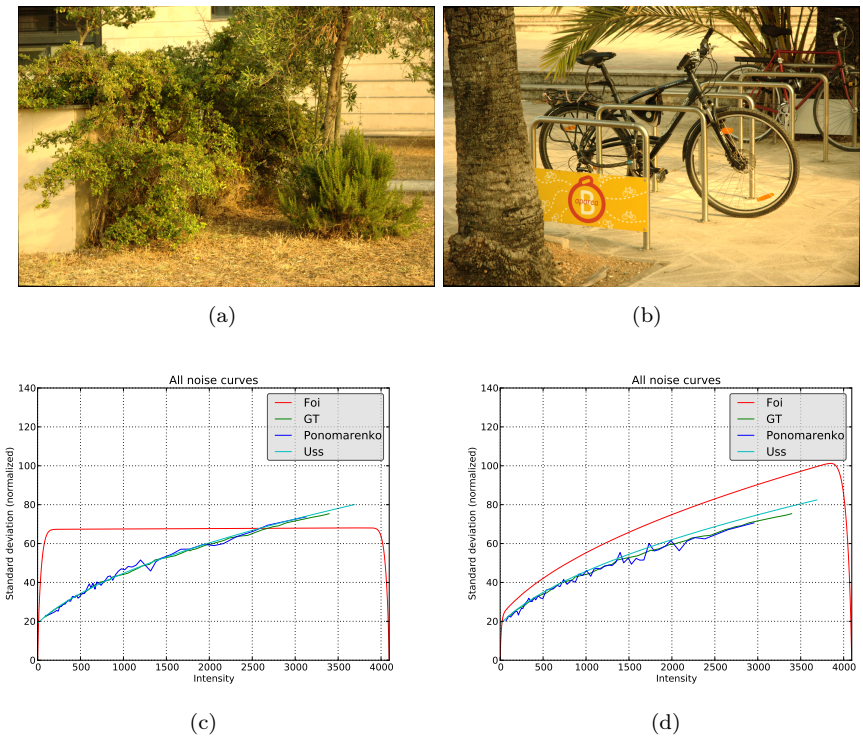
(a) (b)

FIGURE 3. RMSE between the methods and the ground-truth for all the 20 images in our dataset. In general, the RMSE of the modified Ponomarenko et al. (red curve) method is close to zero, which means that indeed it can be considered to be a non-parametric ground-truth curve. The estimations given by Foi et al. (green curve) and Uss et al. (blue curve) are really close to the non-parametric ground-truth, and therefore they are also validated by our approach. (a): obtained RMSEs/image, (b): detailed view.

estimates of a non-parametric method rely on the minimal assumption that the signal is a function of the expectation. In general, the best results are obtained with the modified Ponomarenko et al. method.

To decide if a method is valid or not, its RMSE with respect to the non-parametric ground-truth has to be compared to a threshold. We consider that a method is valid if the RMSE between the measured STD and the ground-truth is less or equal to $\Delta\hat{\sigma}_8 = 0.15$ (assuming that the images are encoded with 8 bits. This value was chosen to be as low as possible and, at the same time, consistent with the accuracy of state-of-the-art noise estimation methods. Since the raw images are encoded with 12 bits, the threshold is $\gamma = \Delta\hat{\sigma}_8 \times 16 = 2.4$. The estimation of the Foi et. al method is considered valid in 17 of 20 images whereas the Uss et al. method is validated with all the images in our dataset.

5.1. Complexity. The Uss et al. method follows four steps: (1) initialization of the TI map and the polynomial function for the variance, (2) estimate texture and noise variance for each TI and NI SWs and label the SW into NI or TI, (3) update the CRLB and finally, (4) apply the noise estimator to the samples associated to each bin and update the variance polynomial. Steps 2, 3 and 4 are iterated until convergence is reached. The complexity of the Uss and the modified Ponomarenko method is similar and their complexity is linear with the number of pixels in the image. Both imply an estimation of the noise variance at the DCT coefficients in small patches of the image after classifying the them according to their intensity. For its part, the Foi et al. method



(a) (b)

(c) (d)

FIGURE 4. highly textured images that caused small oscillations in the noise curves with the proposed method and wrong results with Foi et al: images #19 (a) and #20 (b) (see the obtained RMSEs in Figure 3). (c), (d): the ground-truth obtained with the series (green), the non-parametric ground-truth (darker blue), the Uss et al. method (brighter blue) and the Foi et al. method (red).

follows these steps in order to obtain the final parametric model: compute the detail wavelet coefficients of the image, segment the image to find homogeneous zones, estimate locally pairs of intensity/variance, and finally the maximum-likelihood fitting of the global parametric model. All steps can be computed quickly, but unlike Uss. and the modified Ponomarenko method, it requires a previous segmentation of the image.

5.2. Denoising results. We used the noise curves obtained with the Uss, Foi, and the modified Ponomarenko methods as the input of the NL-Bayes **[78]** denoising algorithm, after applying a VST to the noisy image. Only the green channel was used. Note that according to our threshold criterion, both the Uss and Foi methods are validated and therefore their denoising results in almost all images in the dataset are very similar. Figure 6 (a) shows details of the results obtained for the image #3 of our dataset, where the Foi et al. method failed to estimate correctly the noise. While Uss and the modified Ponomarenko methods denoise the image properly, the noise at the dark zone (the bag over the table) remains visible. The image (e) is the test image #20 of our dataset, where the Uss and the modified Ponomarenko method give an valid estimation, whereas the Foi method overestimates. All methods gave an increased RMSE for that particular image,

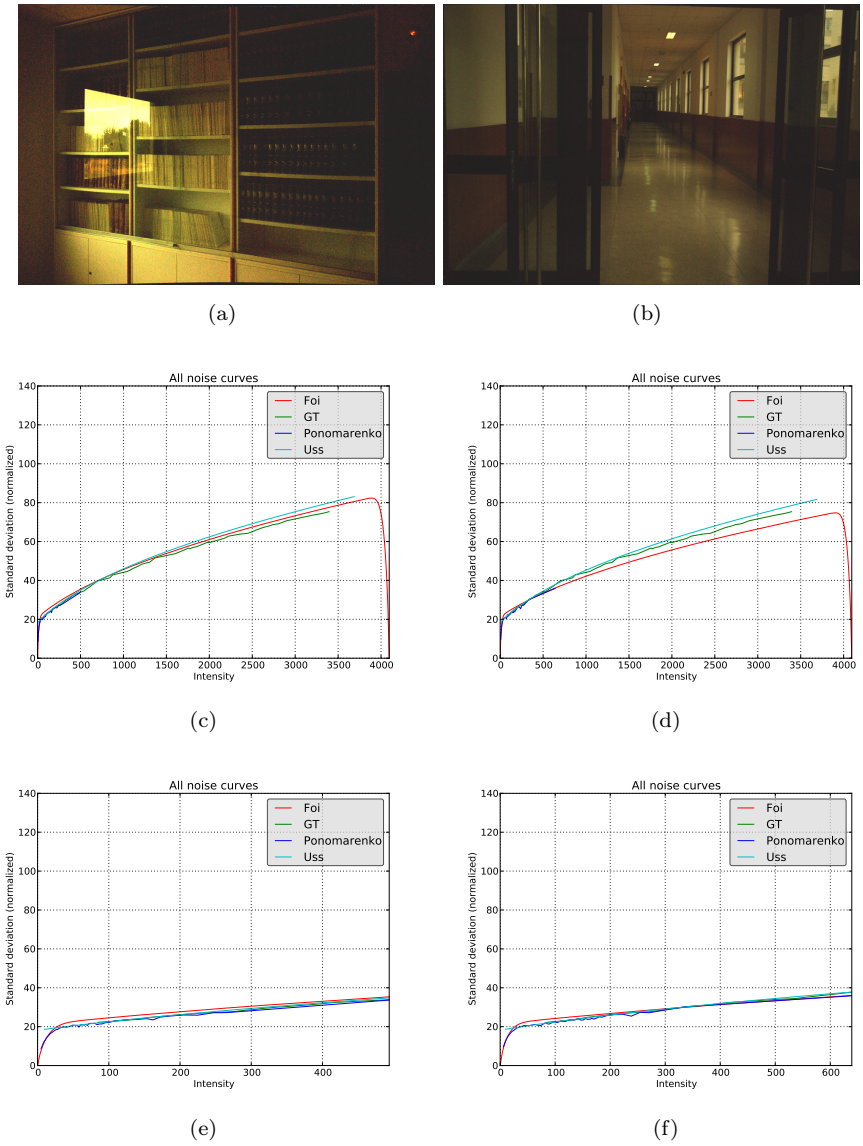
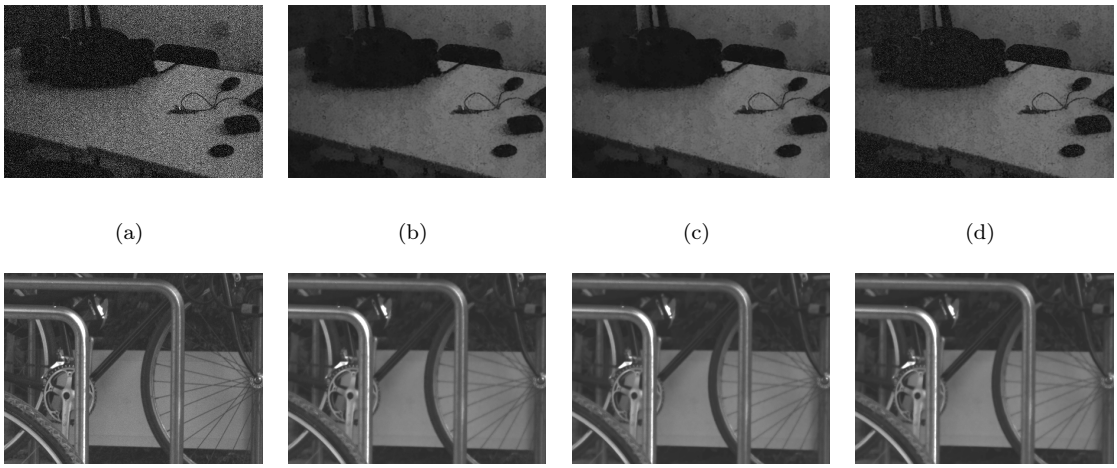


FIGURE 5. Examples of images (#8 (a) and #12 (b) in our dataset) where all algorithms estimated the noise correctly. Their noise curves along all the intensity range (c), (d). Detail of the noise curves only within the range of the estimation given by the modified Ponomarenko et al. method (non-parametric ground-truth, green curve), (e), (f). Note that the Foi et al. method (red curve) matches accurately the ground-truth curves (green and blue), since it is designed to predict the shape of the curve under saturation conditions, whereas the Uss et al. estimation is overall correct, except in the saturation zone.

which causes blurred denoised images and loss of fine details. Both the Uss and the modified Ponomarenko methods give similar visual results, whereas the overestimation in the method blurs the image even more.



(a) (b) (c) (d)

(e) (f) (g) (h)

FIGURE 6. Details of the denoising results with the NL-Bayes algorithm using the noise curves obtained from the noisy images (a, e) with the modified Ponomarenko (b, f), Uss et al. (c, g), and Foi et al. (d, h) methods. Image (a) is the test image #3 of our dataset (it is very dark, so we increased brightness for visualization purposes), where the Foi method was unable to give a reliable estimation and thus the noise is not removed at the dark zones and remains visible (the bag over the table). Image (e) is a detail from the test image #20 of our dataset, where the Uss and the modified Ponomarenko method give an valid estimation, whereas the Foi method overestimates.

6. Conclusion

We showed that estimating an accurate noise curve from a single raw image is possible and can be done by an adaptation of a non-parametric noise estimator **[16]**. The only minimal assumption is that the noise STD is a function of the expected signal. Being able to apply a noise estimator (with relatively low complexity compared to denoising algorithms) to each raw image frees the users of a tedious and sometimes impossible camera calibration task. Indeed, noise curves obtained in an optical lab require measurements for each ISO and each optical setup, a heavy and costly procedure. By estimating the noise directly on the raw image, there is no risk of model error or accuracy loss caused by a noise parameter estimation on another camera.

According to the provided RMSE results (see Figure 3), the non-parametric method proposed here exhibits a very stable error (close to RMSE=0.5) when the image is not composed mainly of

textures. However, even if the image is highly textured (see images #19 and #20 in Figure 4), the error is small and similar to the RMSE obtained with the compared state-of-the-art methods.

In general, the estimation given by the proposed method is as reliable as the actual ground-truth obtained from the temporal series of the series of images of a calibration pattern in the laboratory, and matches the best parametric methods.

Signal-dependent noise is a sufficient model when estimating noise in raw images, but this model is not enough, in general. Noise is transformed at each step of the camera processing chain (raw image, demosaicing, white balance, gamma correction, and JPEG encoding). After demosaicing, the noise becomes correlated (and thus, frequency-dependent), as explained in Chapter 3. Chapter 4 will propose an algorithm for estimating both intensity and frequency dependent noise, which is valid even for JPEG-encoded images.

CHAPTER 3

The noise throughout the camera processing pipeline

Noise in raw images follow a signal-dependent Poisson model and can be estimated by adapting the state of the art homoscedastic noise estimation algorithms **[9, 16]**. *However, the JPEG standard* **[79]** *is preferred in general, since it reduces drastically the size of the file by using lossy and lossless compression. In the raw image the noise follows a Poisson distribution, it is white, and uncorrelated. After JPEG compression, the noise is frequency-dependent, correlated and no longer white. We will show how the noise is affected at each step of the camera processing chain. The noise curves obtained with the Ponomarenko et al. method* **[15, 16]** *along the complete camera processing chain (raw image, demosaicing, white balance, gamma correction, and JPEG-encoding) are shown (along the corresponding autocorrelation matrices of the noise) and compared to the temporal estimation (nonparametric ground-truth curve, see Section 2 of Chapter 3).*

1. The noise curves at each step of the camera pipeline

In this chapter, the effects of the camera processing chain (from the raw to the final JPEG image) on the noise will be discussed, showing detailed noise curves at each step, using different cameras, ISO speeds, and exposure times. Finally, it will be shown that it is possible to obtain a unique noise curve for raw images depending only on the camera model and the ISO that was configured when the picture was taken. For all the experiments in this chapter, two different cameras (Canon EOS 30D and Nikon D80) were used, combining two ISO speeds (ISO 1250, ISO 1600) and four exposure times (1/30s, 1/250s, 1/400s, 1/640s).

1.1. Noise at the Color Filter Array. Each cell of the mosaic acquired by the CCD[1] or CMOS[2] detectors of most digital cameras presents a value at each pixel that can be modeled as a Poisson variable whose expectation is the actual “true" color **[8]**. The random fluctuations of this value around its mean can be considered independent and can be estimated accurately with state-of-the-art homoscedastic noise estimation algorithms **[16, 1, 3]**. More precisely, the noise measured at the CCD or CMOS detector is the combination of several sources:

- **Dark noise**, caused by a small electronic current known as the *dark current*, created at the depletion zone of the semiconductors of the CCD or CMOS detector because of the high intensity electric fields. The dark current (and therefore the dark noise) increases with the temperature. The dark current at the same temperature is a function of each captor at the CCD or CMOS detector, and therefore the additional current does not

[1]Charge-Coupled Device.

[2]Complementary Metal-Oxide Semiconductor.

vary between snapshots. This pattern observed by the camera when the CCD or CMOS detector is not exposed to light is known as the *dark frame*. Since the dark frame pattern is always the same and nothing more than a constant bias, it can be subtracted after taking the photograph of a scene.

- **Photon noise**, due to the physical nature of light. The photons are emitted as quantum of energy with a rate that has some variance. As we mentioned, this process can be modeled with a Poisson distribution.
- **Readout noise**. Since the charge accumulated at the CCD is really small, it must be processed by an analogic amplifier which adds noise to the measurement.
- **Shot noise**. The noise of the Poisson photon acquisition can be modeled in good light as an additive Gaussian noise with standard deviation equal to the square root of the expected intensity. Nevertheless, this approximation is no longer true for very low photon income, in which case the Poissonian image is called "shot noise". Indeed, in a dark scene most of the photosensitive surface receives a few photons whereas the rest of the surface does not. Therefore, after the amplification of the measured charge, it causes that isolated bright dots appear in the output image.
- **Electronic noise** caused by the absorption of electromagnetic energy by the semiconductors of the camera circuits and the crosstalk phenomenon, among others.

1.2. The camera pipeline. Most of the noise evaluation methods assume or require the noise to be uncorrelated and thus they are adequate to estimate directly the noise in the image formed at the CFA[3], the raw image. Unfortunately, the first transformation applied to the CFA to obtain a color image (by demosaicing the Bayer pattern) correlates the noise. In addition, the lossy compression performed by JPEG encoding makes the noise frequency-dependent. Many methods [**16, 1**] assume that the variance of the noise can be estimated by measuring the variance only at high-frequency coefficients, supposing that noise is frequency-independent. Yet, JPEG compression quantizes the values of the coefficients depending of the frequency. This energy loss of certain coefficients causes an underestimation of the noise estimation in these methods.

The following steps and transformations are performed in the camera processing pipeline in order to obtain a final JPEG image from the raw:

(1) Acquire the image data from the CFA (**raw image**);
(2) interpolate the data from the CFA to obtain a color image from the grayscale mosaic (**demosaicing**);
(3) adjust the weight of each color channel in order that the picture has the same colors as the photographed scene (**white balance**);
(4) modify the dynamic range of the image in order to give more importance to the intensities that human observers are more adapted to detect. This is done a non-linear function to

[3]Color Filter Array.

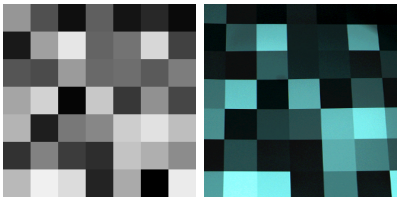


FIGURE 1. Left: calibration pattern used to obtain the ground-truth of the cameras. Right: photograph of the calibration pattern taken by the Canon EOS 30D camera, raw image. The image is not grayscale because no white balance has been applied at this first step.

each pixel, called **gamma correction**. For some cameras this function cannot be formulated with a simple expression and it is given as a table that maps the input intensities into different values (**tone curve**);

(5) apply lossy compression (quantization of the DCT coefficients).
(6) apply lossless compression (Huffman encoding).

Another objective of this chapter is to build *ground-truth* (GT) noise curve for the noise at the CFA for both the Canon and Nikon cameras, depending on the ISO. A noise curve associates with each observable intensity in each color channel a value for its standard deviation. To evaluate and validate different noise estimation algorithms the GT is an absolute requirement. To build the ground-truth curves, the calibration pattern shown in Figure 1 (left) was printed with a high quality plotter and several (about 500) raw image pictures of the calibration pattern were taken with the Canon and Nikon cameras with the same lighting conditions, fixing the ISO and the exposure times. To minimize the effect of the edges in the image causing fluctuations due to sub-pixel displacements of the still camera, and not to noise, the photographed calibration pattern contains large different gray level rectangles (Figure 1, right). The image has a green tone because the Bayer pattern presents two green pixels for one red and one blue. The variance of a pixel value along different snapshots of the same still scene (Figure 2) can only be explained by the noise and therefore the noise. Thus, the noise curve obtained computing the standard deviation values of the temporal series gives the ground-truth noise curve for that camera. This noise curve obviously also depends on the ISO and on the exposure time (Figure 4). As shown in Section 2, since the standard deviation of the noise is a function of the intensity, it is possible to overlap all the noise curves from different exposure times into a single ground-truth curve combining the curves from all the exposure times.

1.2.1. *Step 1/5: raw image.* The first step to acquire the raw image is to count the number of incident photons over the CFA along the exposure time, using a CCD or CMOS detector. Because of the photoelectric effect, the CCD or CMOS detector is able to accumulate electric charge by the absorption of electromagnetic energy. The camera electronics measures the voltage produced by the accumulated charge during the exposure time, digitizes the values and finally stores them using some proprietary format (NEF for the Canon and CR2 for the Nikon cameras). For both

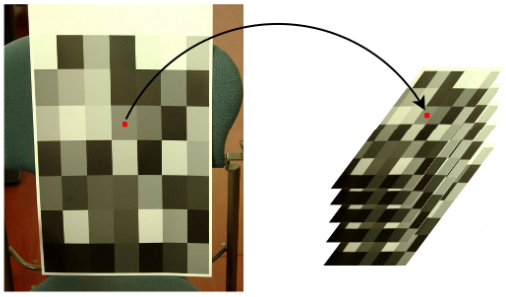


FIGURE 2. Computing the STD of the temporal series with the calibration pattern.

cameras, the intensity of a pixel in one of the four channels (R, G_1, G_2, B) is encoded with 12 bits. At this first step, the noise is uncorrelated, that is, the noise at a certain pixel is not related with the noise at any other pixel with the same signal intensity. Figure 3 shows the autocorrelation function of a rectangle in the calibration pattern. This function only presents a peak at $[0, 0]$, that is, the pixels are almost only correlated with themselves and not with any other neighbor pixel. It can also be observed, for the Canon camera (left), a pattern of some pixels at distance 2 and beyond with small autocorrelation, caused by crosstalk between captors of the same type in the CFA.

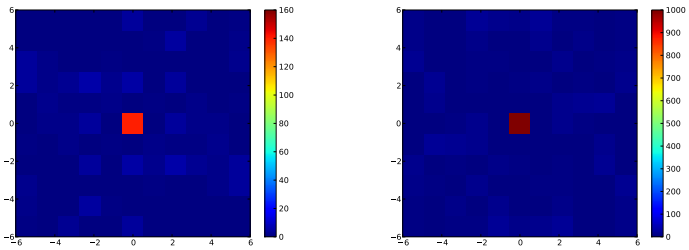


FIGURE 3. Autocorrelation function for a rectangle of the calibration pattern for the Canon (left) and Nikon (right) cameras with ISO 1600 and t=1/250s in the raw image. The noise is mostly not correlated, although it can be observed, for the Canon camera, a pattern of some pixels at distance 2 and beyond with small autocorrelation, caused by crosstalk between captors of the same type in the CFA. On the other hand, the autocorrelation at $[0, 0]$ shows that the image obtained with the Nikon camera contains more noise than the image of the Canon camera.

The noise curve obtained by the temporal series (ground-truth) and the noise curve obtained by any accurate noise estimation algorithm coincide, since the noise is Poisson distributed. Figure 4 shows how the curves match in the case of the Ponomarenko et al. method **[16]**, using ISO 1600 and t=1/250s for both the Canon and Nikon cameras.

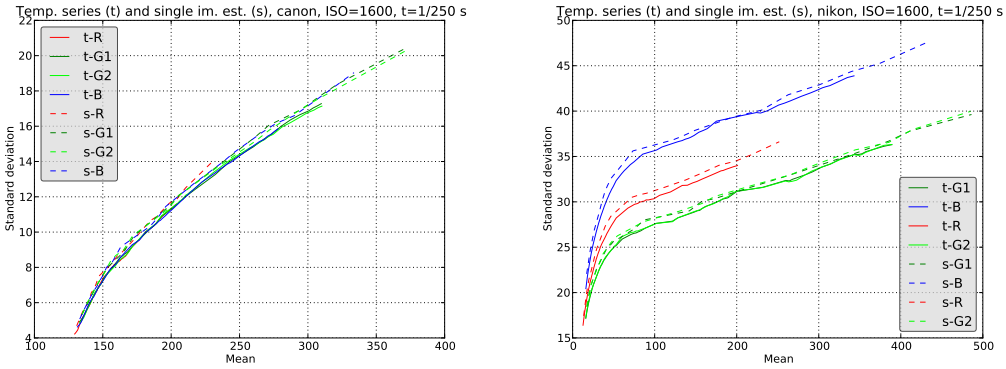


FIGURE 4. Comparison between the temporal series standard deviation (ground-truth) and the noise curve obtained by the Ponomarenko et al. algorithm in raw images. The images were taken with ISO 1600 and t=1/250s. The solid lines correspond to the temporal series and the dashed to the single image estimation with the Ponomarenko et al. algorithm. Left: Canon camera. Right: Nikon camera.

Since noise is Poisson distributed, the variance of the noise is directly related to the intensity of the underlying signal. Therefore, the variance of the noise should follow a simple $\sigma^2 = a + bu$ linear relation, where u is the intensity of the ideal noiseless image and constant a a is due to the other mentioned sources of uniform white noise explained in Section 1.2, dark, readout, and electronic noise. In the case of raw images, the noise is only a function of ISO, since it controls the intensity multipliers. Therefore, it is possible to combine the curves obtained with different exposure times (and the same ISO) to get a unique ground-truth curve.

However, some cameras pre-process the data acquired at the CCD before writing the raw image. Also, the saturation of the captors at the most dark and bright zones breaks the linearity between the intensity and the variance. Figure 5 shows the square root of the linear model of the variance obtained by the least squares method. In the case of the Nikon camera, the image is dark enough to show the non-linear effect of the saturation in the noise curve. It is therefore preferable and more reliable to estimate the noise curve without assuming *a priori* a noise model. Thus we shall prefer non-parametric estimation methods that give a signal-dependent noise curve from the image itself.

1.2.2. *Step 2/5: demosaicing.* The image acquired at the CFA is just a mosaic where each captor measures the intensity of the R, G_1, G_2 or B channels. Depending on the CFA, the disposition of the pixels associated with each channel may be different. The goal of the demosaicing step is to create a color image from the mosaic using an interpolation algorithm. In our tests we used the Adams-Hamilton **[80, 81]** because of its simplicity. Indeed, it bases its estimation on the gradients and pixels at distance one and two.

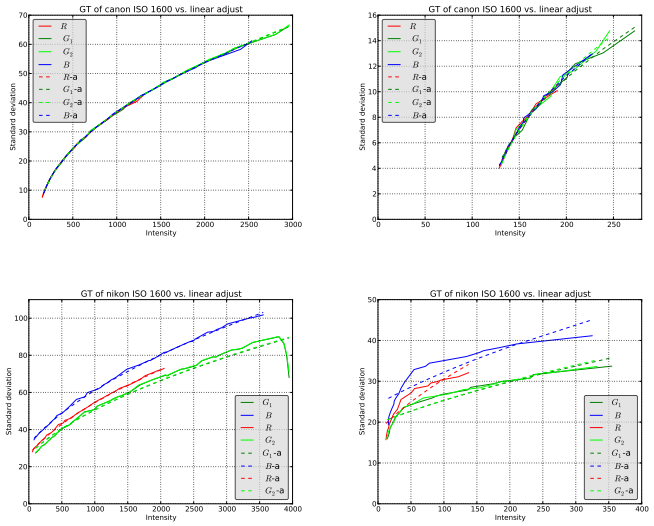


FIGURE 5. Linear approximation of the variance ground-truth for the Canon (up) and the Nikon (down) cameras with ISO 1600 (solid line: original values, dashed line: linear approximation). Globally, the linear approximation coincides with the computed ground-truth values. On the right, a detailed view of the estimation for the darkest pixels in the same noise curve. For the Nikon camera, the estimated noise in the dark gray level values does not coincide with the linear approximation, because of saturation.

Figure 6 shows the comparison between the temporal series and the estimation obtained using the Ponomarenko et al. algorithm for both the Canon (left) and Nikon (right) cameras.

It shows that

- each channel has a different noise level.
- the temporal series has more noise than the single image estimation.

Now each channel has a different noise level since the Adams-Hamilton algorithm does not process each channel in the same way. The temporal series and the single image estimation coincide in the raw image but not after the demosaicing. The reason is that the Adams-Hamilton demosaicing does, among other operations, an averaging of pixels at distance one, and therefore reduces the spatial standard deviation that is measured in a single image. However, in the temporal series the standard deviation is computed using pixels at the same location in the temporal series. This explains why the noise in the temporal series is higher than the noise measured using a single image. Since the image processing pipeline is sequential, the temporal noise curves and those measured on a single image will not coincide anymore after the demosaicing step. Figure 6 permits to compare the estimation obtained using the temporal series and using the Ponomarenko et al. algorithm.

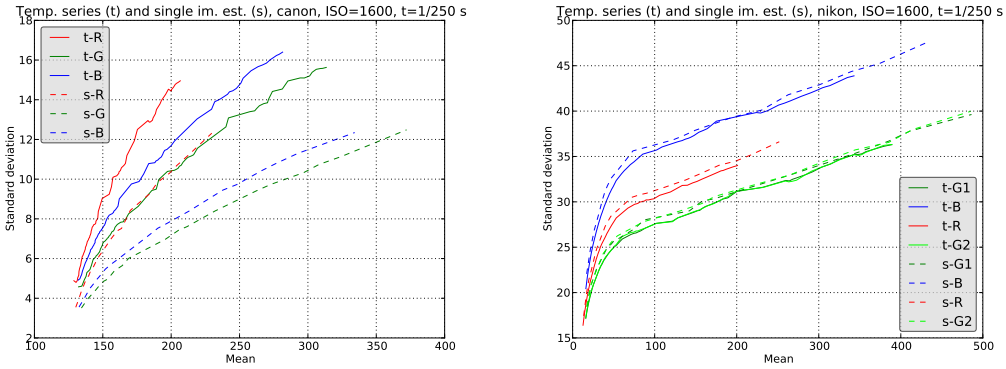


FIGURE 6. Comparison between the temporal series standard deviation and the noise curve obtained by the Ponomarenko et al. algorithm after demosaicing. The images were taken with ISO 1600 and t=1/250s. The solid lines correspond to the temporal series and the dashed to the single image estimation with the Ponomarenko et al. algorithm. The curves do not coincide anymore. Left: Canon camera. Right: Nikon camera.

In the raw image, the autocorrelation is close to zero for any location different from $[0, 0]$, meaning that the noise is not spatially correlated. In the demosaiced image the autocorrelation goes down to zero at distance two or more. This means that the pixel noise is correlated with the noise at pixels whose distance is one. Figure 7 shows that the maximum autocorrelation is attained at $[0, 0]$, but the four points at distance 1, $\{[0, -1], [0, 1], [-1, 0], [1, 0]\}$, also have a significant autocorrelation. This result is coherent with the Adams-Hamilton algorithm, where pixels at distance higher than one show a very low autocorrelation and can be considered non-correlated in practice.

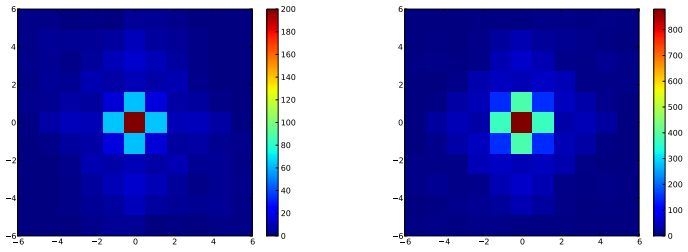


FIGURE 7. Autocorrelation function for a rectangle of the calibration pattern for the Canon (left) and Nikon (right) cameras with ISO 1600 and t=1/250s in the demosaiced image. Each pixel is correlated with four pixels at distance one. Pixels at a distance higher than one show a very low autocorrelation and they can be considered to be non-correlated in practice. Also, the Nikon camera shows a higher autocorrelation because the noise is higher.

1.2.3. *Step 3/5: white balance.* After the demosaicing each pixel of the image has a color, but it has to be corrected in order to show the same colors a human would observe. The reason is that the color filters of the CFA do not have all the same gain. Therefore, taking a picture of a gray object will produce a colored image if the gain for the R (red), G (green) and B (blue) channels is not balanced. For example, the photograph of the calibration pattern in Figure 1 (right) is bluish since the blue channel has more gain than the others. However, apart from correcting the colors, the white balance allows to simulate the kind of light on a scene. The kind of light is well defined using its *color temperature*, that is, the kind of light that would emit a blackbody at that temperature. For example, a candle light is within the range $[1000K, 2000K]$, the fluorescent lamps within $[4000K, 5000K]$, the light of a sunny day within $[5000K, 6500K]$, etc. Because the calibration pattern only contains gray rectangles, the multipliers are chosen in such a way that the resulting rectangles are gray, that is, the same intensity at each color channel. For example, fixing ISO 1600 and exposure time $1/250s$, the multipliers for the Canon camera are $M_R = 1.30$, $M_G = 1.00$, $M_B = 1.08$ and for the Nikon they are $M_R = 1.8$, $M_G = 1.00$, $M_B = 1.14$. In both cameras, the green channel is more sensitive that the red or blue channels and therefore the white balance step increases them more. Figure 8 shows the results of the noise estimation after the white balance for both Canon and Nikon cameras, with ISO 1600 and exposure time 1/250s. Since the white balance factors are higher than one, the whole dynamic range of the image is increased. Compare the maximum intensities of the noise curves corresponding to the demosaicing step and the white balance step. The white balance of course increases the noise. For example, the factor M_R is higher than M_G and M_B as it can be seen in the curves. The red is now the most noisy channel (then the blue and finally the green channel, which was not modified by the white balance). Because of the dark, readout, and electronic noises, the minimum intensity registered by the camera is not encoded at zero, but at some fixed value. It can be observed that this value in the Canon camera (near intensity 128) is higher than in the Nikon camera (near intensity 30) and therefore multiplying the red channel by 1.30 is much more noticeable in the Canon curve than in the curve corresponding to the Nikon camera.

1.2.4. *Step 4/5: gamma correction (tone curve).* Human vision perception is not linear with the signal intensity and can be modeled approximately with a power function. However, the charge accumulated by the CCD or CMOS detector is linear with the number of incident photons on the device during the exposure time. Since the information at the darkest zones is invisible to a human observer, it is applied a power function called the *gamma correction* to the linear data captured by the CCD or CMOS detector before attempting lossy compression. The idea is not to only enhance the contrast of the image but also to encode more accurately the information in the dark areas that is invisible in the raw image by enhancing it with the concave gamma function. This gamma function has the form $f_{k,\gamma}(\mathbf{u}) = k\mathbf{u}^\gamma$, where the typical values for γ vary from 1.8 to 2.2. However, commercial cameras do not apply this simple formula but use some precomputed tables to simulate the non-linearity, called *tone curves*. In general, the tone curves saturate the signal at the brightest zones of the image. Figure 9 shows the effect of applying the gamma correction

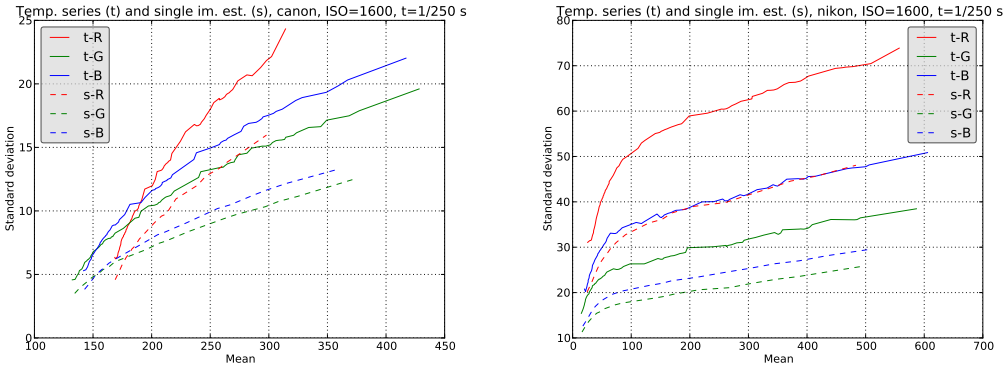


FIGURE 8. Comparison between the temporal series standard deviation and the noise curve obtained by the Ponomarenko et al. algorithm after white balance. The images were taken with ISO 1600 and t=1/250s. The solid lines correspond to the temporal series and the dashed to the single image estimation with the Ponomarenko et al. algorithm. The minimum intensity value is much lower for the Canon than for the Nikon. Therefore, the displacement to the right of the red channel is more noticeable for the Canon camera. The dynamic range is increased and since each channel has different multipliers, the noise now is higher for the red channel, then for the blue and finally for the green, that has not been modified. Left: Canon camera. Right: Nikon camera.

to the images obtained by the Canon and Nikon cameras. Previously, the demosaicing and white balance were applied.

It can be observed that

- the dynamic range is clearly increased, much more than the white balance stretching, specially in the dark areas;
- the noise increases significantly because of the power law function;
- the noise curve function is no longer monotonically increasing. When the intensity is over some threshold the noise saturates the captor and the measured standard deviation decreases instead of increasing (see Figure 9, right). Indeed, the derivative of the gamma function, which is concave, is larger than 1 in the dark part and smaller than 1 in the bright part. It is easily checked that in a first approximation, the noise standard deviation is multiplied by the derivative of the gamma function. Indeed, assuming that the noise $\mathbf{n}$ is small with respect to the signal $\mathbf{u}$, we have by a first order asymptotic expansion $\gamma(\mathbf{u}+\mathbf{n}) \simeq \gamma(\mathbf{u}) + \gamma'(\mathbf{u})\mathbf{n}$. Thus, after gamma correction the noise at level $\mathbf{u}$ is approximately $\gamma'(\mathbf{u})\mathbf{n}$.

1.2.5. *Step 5/5: JPEG compression.* The final step is to encode the image with the JPEG standard in order to reduce the size of the resulting file. The encoding is not done in the *RBG*

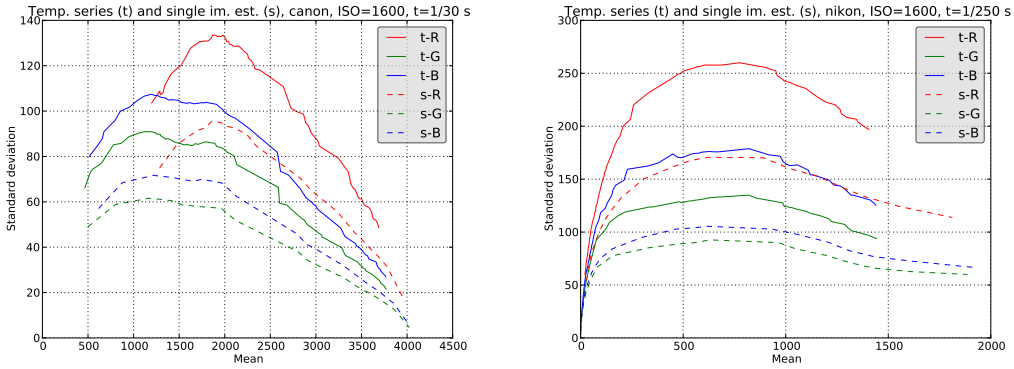


FIGURE 9. Comparison between the temporal series standard deviation and the noise curve obtained by the Ponomarenko et al. algorithm after gamma correction. The images were taken with ISO 1600 and t=1/250s. The solid lines correspond to the temporal series and the dashed to the single image estimation with the Ponomarenko et al. algorithm. The dynamic range is clearly increased specially in the dark areas. The noise increases significantly because of the power law function. When the intensity is over some threshold the noise level begins to decrease instead of increasing because the signal gets saturated. Left: Canon camera. Right: Nikon camera.

color space but in the $Y'C_BC_R$, where Y' is the luma component and C_B and C_R the blue-difference and red-difference chroma components, namely. Also, the number of bits is reduced to 8 bits/channel/pixel. Since human perception is much more sensitive to changes in luma than in chroma components, the C_BC_R components are subsampled. Usually the $4:2:0$ subsampling is performed, which means that both the horizontal and vertical resolutions are halved. This step implies loss of information. After the chroma subsampling, the image is tessellated into blocks of 8×8 pixels and the 2D DCT-II of each is computed. Human perception is not adapted to distinguish accurately differences in luma and chroma when the signal varies rapidly, that is, at its high frequencies. Therefore, it is possible to quantize the information at the high-frequencies without the notice of human observers. Also, it is well known that the variance of the high frequency coefficients in the 2D-DCT transformation of a block is mainly explained by the noise. Since JPEG encoding implies quantizing these high-frequency coefficients, most of the methods will not be able to detect the noise by using these high frequencies. For example, the Ponomarenko et al. method **[16]** which tries to estimate the noise using these high-frequency coefficients, gives an underestimation if the image is JPEG encoded.

For each coefficient $C_{i,j}$ of the DCT block of the image ($i, j \in [0, 7]$), JPEG applies the operation

$$J_{i,j} = \text{round}\left(\frac{C_{i,j}}{R_{i,j}}\right) R_{i,j},$$

$$\begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

FIGURE 10. Example of quantization matrix for JPEG encoding, as given in the standard [**79**].

where $J_{i,j}$ is the resulting quantized coefficient and $R_{i,j}$ is the value of the quantization matrix for the coefficient at frequency $[i, j]$. Figure 10 shows the quantization matrix proposed by the JPEG standard [**79**].

Matrix R contains low values for the low-frequencies and higher values for the high-frequencies, to quantize them more. It is designed such a way that it causes a great energy loss in the high frequencies (the energy at the high-frequencies comes mainly from edges and textures) whereas the medium and low frequencies (where most of the visual information is located, with the exception of edges) are mostly respected. The actual values of the matrix depend on the quality at which the JPEG encoder is configured, where the less quality, the higher the quantization. The quality of the final JPEG image is determined by a quality parameter that goes from $Q = 0$ (worst quality) to $Q = 100$ (best quality). Most of the cameras use quality $Q = 92$ as the standard quality by default.

As an example, let us consider an image made of samples of pure white Gaussian noise of mean zero and standard deviation $\sigma = 10$. Then, compute the 2D DCT-II of all non-overlapping 8×8 blocks in the image. If we denote by $D_k[i, j]$ the coefficient of frequency $[i, j] \in [0, 7]^2$ of block $k \in [0, M - 1]$, then the empirically estimated variance of the noise at that frequency along all the blocks is $\mathrm{Var}_k\left(D_k[i, j]\right) = \dfrac{1}{M-1} \displaystyle\sum_{k=0}^{M-1} \left(D_k[i, j]\right)^2$, where M is the number of non-overlapping blocks. Table 11 shows the averaged STD of all blocks according to the frequency. On the left, result from an image of pure white Gaussian noise of STD=10. On the right, the same image after JPEG encoding with quality $Q = 70$. The $[0, 0]$ frequency (DC) is at the leftmost top corner. Note that after JPEG compression, the STD decreases as the frequency increases, because of the quantization matrix. The slightly increased energy overall the matrix is due to the blocking artifacts created by JPEG compression.

After quantization, the number of possible values for each DCT coefficient is reduced. In this situation the lossless Huffman encoder can achieve good compression ratios. This is the last step of the compression, that encodes the quantized coefficients found following a zig-zag scan of the DCT block. Since the energy of the signal in natural images decreases as the frequency increases,

$$\begin{pmatrix} 10.00 & 9.83 & 10.04 & 9.95 & 10.08 & 9.85 & 9.88 & 9.92 \\ 10.09 & 10.00 & 9.87 & 10.02 & 10.03 & 10.23 & 9.91 & 9.93 \\ 9.94 & 10.10 & 9.87 & 9.91 & 9.92 & 10.07 & 10.20 & 10.07 \\ 10.05 & 10.14 & 10.04 & 9.84 & 9.92 & 9.91 & 9.91 & 9.90 \\ 10.10 & 10.16 & 10.06 & 10.18 & 9.98 & 9.75 & 9.98 & 10.14 \\ 10.02 & 10.08 & 9.94 & 9.98 & 10.08 & 9.85 & 10.10 & 9.97 \\ 9.88 & 9.98 & 9.93 & 9.87 & 9.90 & 9.92 & 10.09 & 9.85 \\ 9.99 & 10.05 & 10.06 & 10.03 & 10.08 & 9.91 & 10.00 & 10.06 \end{pmatrix}$$

$$\begin{pmatrix} 10.39 & 10.06 & 10.21 & 10.42 & 10.82 & 11.39 & 10.59 & 9.39 \\ 10.31 & 10.24 & 10.12 & 10.57 & 11.14 & 10.52 & 9.27 & 10.38 \\ 10.22 & 10.34 & 10.30 & 10.71 & 11.56 & 10.40 & 8.28 & 10.20 \\ 10.32 & 10.52 & 10.72 & 10.84 & 10.45 & 4.50 & 5.64 & 9.24 \\ 10.57 & 10.79 & 11.75 & 10.51 & 8.30 & 2.24 & 2.62 & 7.33 \\ 10.81 & 11.52 & 10.46 & 9.04 & 6.14 & 2.63 & 1.67 & 4.29 \\ 11.09 & 9.26 & 6.52 & 4.78 & 2.62 & 1.47 & 1.05 & 2.27 \\ 7.90 & 4.01 & 3.76 & 3.70 & 0.98 & 3.15 & 2.47 & 3.51 \end{pmatrix}$$

FIGURE 11. Averaged variance along all the blocks at all frequencies. Up: result from an image of pure white Gaussian noise of STD=10. Down: the result using same image after JPEG encoding with quality $Q = 70$. The $[0, 0]$ frequency (DC) is at the leftmost top corner. Note that after JPEG compression, the STD decreases as the frequency increases, because of the quantization matrix. The slightly increased energy overall the matrix is due to the blocking artifacts created by JPEG compression.

the encoder uses a special code word *EOB* (end-of-block) which indicates that next coefficients in the zig-zag scan are all zero. This improves even more the compression ratio.

To study the effect of JPEG encoding, the parameter $Q = 92$ was used to encode the images for the experiments of this chapter. Figure 12 shows the noise curves for both the Canon and Nikon cameras after JPEG encoding.

It can be observed that

- the dynamic range of the image has not changed after JPEG encoding.
- the noise is reduced after JPEG encoding, because of the quantization of the coefficients, particularly those corresponding to the high frequencies.

Figure 13 shows the autocorrelation function after JPEG encoding for both the Canon and Nikon cameras. Each pixel is strongly correlated with the four pixels at distance 1, and since JPEG removes most of the energy at the high-frequencies, it can also be observed some small correlations

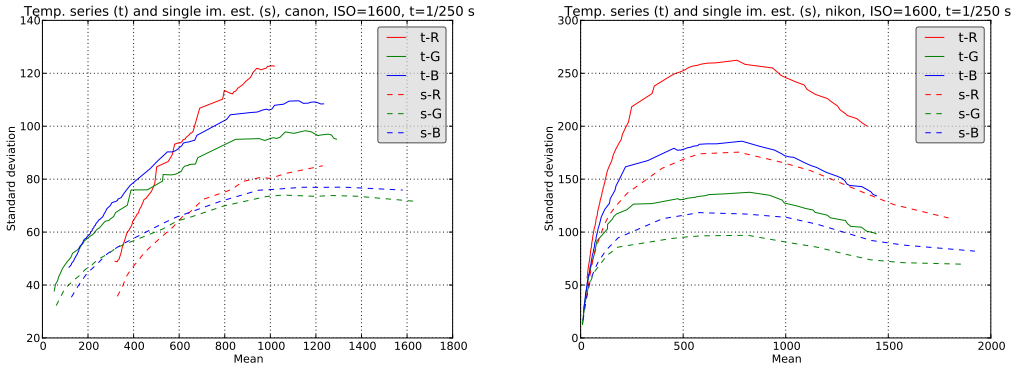


FIGURE 12. Comparison between the temporal series standard deviation and the noise curve obtained by the Ponomarenko et al. algorithm after JPEG encoding. The images were taken with ISO 1600 and t=1/250s. The solid lines correspond to the temporal series and the dashed to the single image estimation with the Ponomarenko et al. algorithm. The dynamic range is not modified because of JPEG encoding. The curves are shown using a 12 bits intensity range to compare with the previous steps, although in reality the image is encoded with 8 bits instead of 12 bits/color channel. Left: Canon camera. Right: Nikon camera.

with pixels with distances up to 5 pixels. Also, the Nikon camera shows a higher autocorrelation because the noise is higher than the noise of the Canon camera.

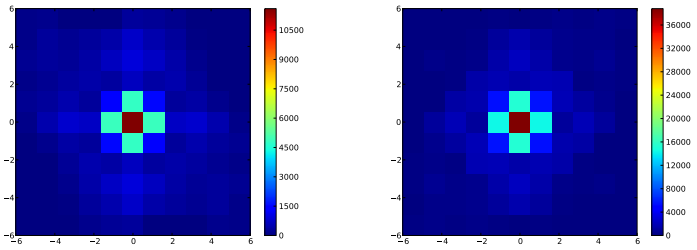


FIGURE 13. Autocorrelation function for a rectangle in the calibration pattern for the Canon (left) and Nikon (right) cameras with ISO 1600 and t=1/250s in the JPEG-encoded image. Each pixel is strongly correlated with the four pixels at distance 1, and since JPEG removes most of the energy at the high-frequencies, it can also be observed some small correlations with pixels with distances up to 5 pixels. Also, the Nikon camera shows a higher autocorrelation because the noise is higher than the noise of the Canon camera.

1.3. Synthesis: effect of the complete image processing pipeline on the noise curve. In this chapter we show the evolution of the noise curves throughout the complete processing

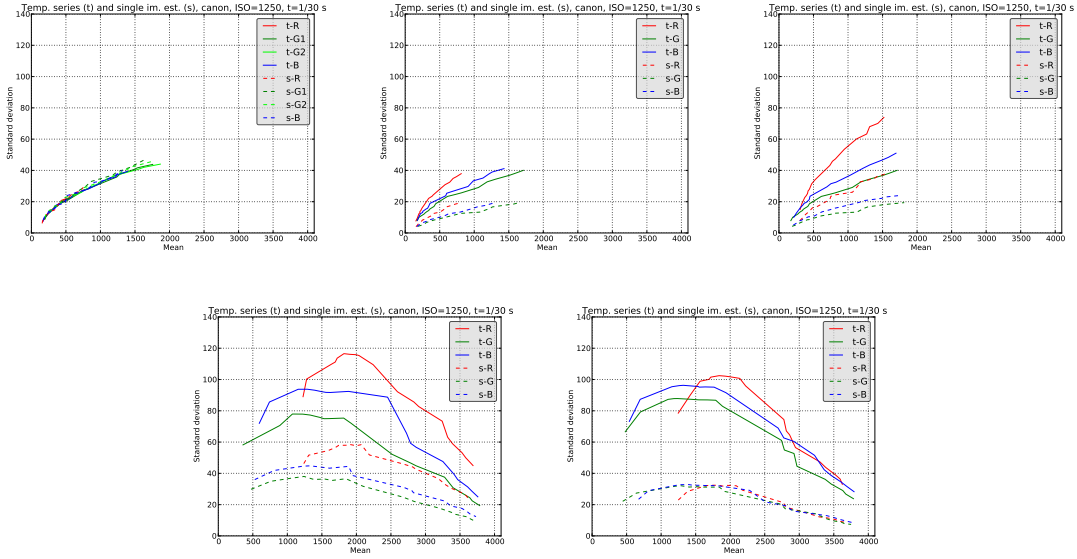


FIGURE 14. Complete pipeline for ISO 1250, t=1/30s, Canon: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression. In the first step (raw image), all four color channels show share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts de decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain.

pipeline of the cameras, from the initial raw to the final JPEG image. We fixed an exposure time t=1/30s and two ISO speeds, ISO 1250 (Figure 14) and ISO 1600 Figure 15 for both the Canon and Nikon cameras. In the first step (raw image), all four color channels show share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts de decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain. Comparing the raw noise curves from both pipelines, it can be observed that the noise of the images with ISO 1600 is higher than the noise of the images with ISO 1250, as expected. Also, the dynamic range is larger with ISO 1600, since the sensitivity is higher. Apart of the different dynamic ranges, the curves from ISO 1250 and ISO 1600 show the same behavior along the processing chain. Figure 16 and Figure 17 show the curves corresponding to ISO 1250 and ISO 1600 for the Nikon camera.

The main difference between the Canon and Nikon cameras that can be observed is that when using the same exposure time and ISO, the dynamic range of the Nikon camera is larger in the Nikon compared to the Canon camera. In other terms, the Nikon camera is more sensitive to light than the Canon, under the same ISO configuration. This explains why in all measurements the

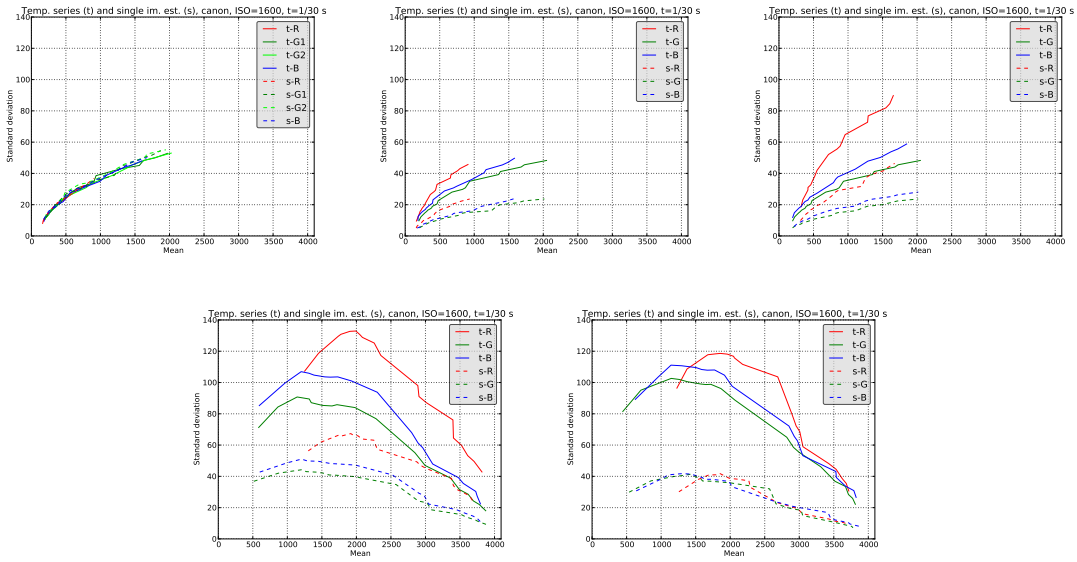


FIGURE 15. Complete pipeline for ISO 1600, t=1/30s, Canon: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression. In the first step (raw image), all four color channels show share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts de decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain.

Nikon camera always show more noise. Since the Nikon camera is more sensitive to light than the Canon, the gamma correction functions start to saturate the signal earlier. This effect can be seen looking at the curves after the gamma correction step. For example, the noise at the red channel begins to decrease at intensity 2000 approximately with both ISO 1250 and ISO 1600 in the Canon camera. In the case of the Nikon camera it begins at intensity 700, approximately.

2. Overlapping of noise curves with different exposure times

Historically, the ISO speed (standard ISO 5800:2001) was related to the sensitivity of the photographic film used to take a picture with analogic cameras. With modern digital cameras, the ISO level follows the ISO 12232:2006 norm, that sets the reference values for the calibration of digital still cameras. In general, the ISO speed is simulated by multiplying the readouts at the raw image by some fixed factors. As a consequence, the standard deviation of the noise varies linearly with the ISO factors. However, changing the exposure time using the same ISO does not alter the shape noise curve, but only increases the dynamic range with the exposure time. In Section 1.2, several noise curves corresponding to different exposure times were shown. Since the shape of the noise curve just depends on the ISO, it makes sense to combine the curves obtained at different

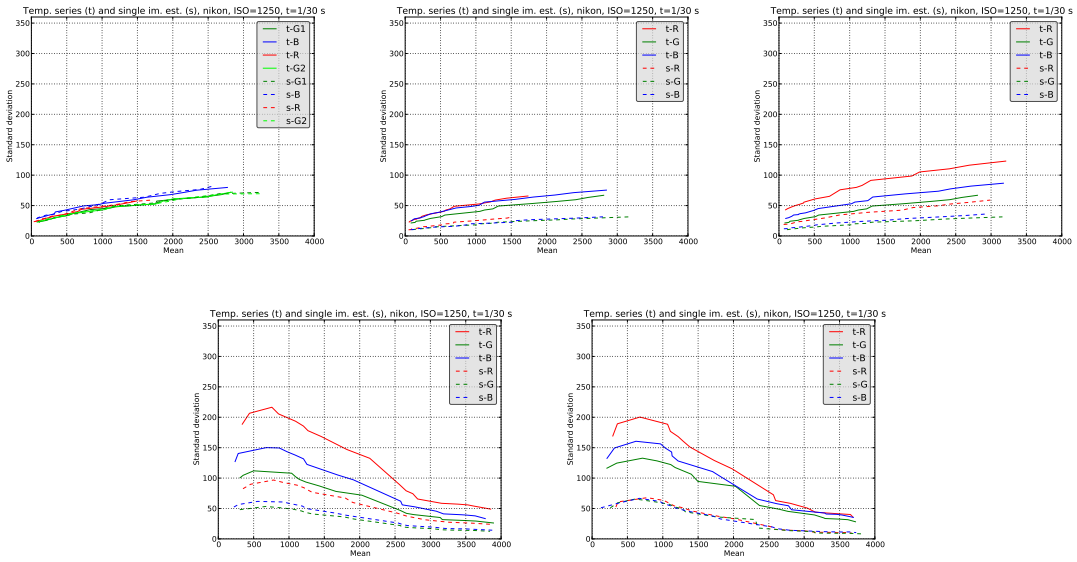


FIGURE 16. Complete pipeline for ISO 1250, t=1/30s, Nikon: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression. In the first step (raw image), all four color channels show share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts de decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain.

exposure times and same ISO to get a unique noise curve. If this procedure is applied to the raw image, the resulting curve can be used as a ground-truth for some ISO of a given camera. Figures 18 and 19 show the overlapping of the noise curves corresponding to 1/30s, 1/250s, 1/400s and 1/640s for the Canon and Nikon cameras at the raw, demosaicing, white balance, gamma correction and JPEG encoding steps.

3. Mean ground-truth curves

Section 2 showed that noise curves coming from different exposure times overlap at the raw image step. After the demosaicing step, the curves do not overlap anymore because the demosaicing algorithm computes different interpolations depending on the channel. Since the image at the raw step has not yet been processed by the demosaicing and further algorithms, it is only a function of the exposure time and ISO speed and can be used as a ground-truth curve. This is particularly useful to get an estimation of the accuracy of noise estimation algorithms. The noise curves corresponding to different exposure times should overlap. In practice, there is some small error due to the random nature of noise itself and the limited accuracy of the noise estimation algorithms. Figures 18 and 19 show that the noise curves overlap almost exactly in the raw image step, but

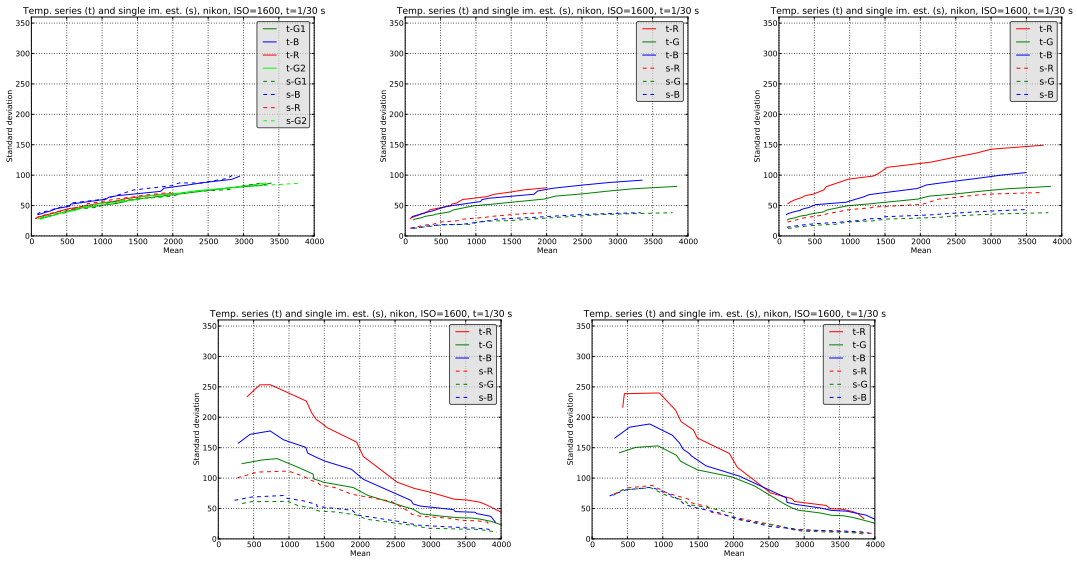


FIGURE 17. Complete pipeline for ISO 1600, t=1/30s, Nikon: raw image, demosaicing, white balance, gamma (tone curve) correction and JPEG compression. In the first step (raw image), all four color channels show share the same noise curve. After demosaicing, each color channel have a different noise curve, since the Adams-Hamilton algorithm treats each channel in a different way. Finally, the gamma correction saturates the noise curve, which starts de decrease from a certain intensity. The final JPEG noise curve exhibits the combination of all these effects along the processing chain.

with some negligible error. To get a unique noise curve, the average of the values of the curves corresponding to different exposure times is computed (see Algorithm 13). Figures 20 and 21 show the computed ground-truth for the Canon camera for ISO 1250 and ISO 1600. Figures 22 and 23 show the same for the Nikon camera. On the right, a detailed view of the noise curve at darkest intensities.

The darkest zone of the noise curve has a particular interest, since the gamma correction increases the energy of the image specially in these dark zones, making the noise really noticeable afterwards. After the white balance step, the noise has been multiplied by a different factor that depends on the channel. If the noise is convolved by some kernel, this visible noise will appear to the observer as colored stains on the image. The DCT coefficient quantization step in JPEG is similar to convolution with a kernel that low-passes the image and therefore these kind of artifacts are still visible after JPEG compression.

4. Comparison of the autocorrelation functions at different scales

Consider the operator $\mathbf{S}$ that tessellates the image into sets of 2×2 pixels blocks, and replaces each block by a pixel having the average of the four pixels. We define the n-th *subscale* of an

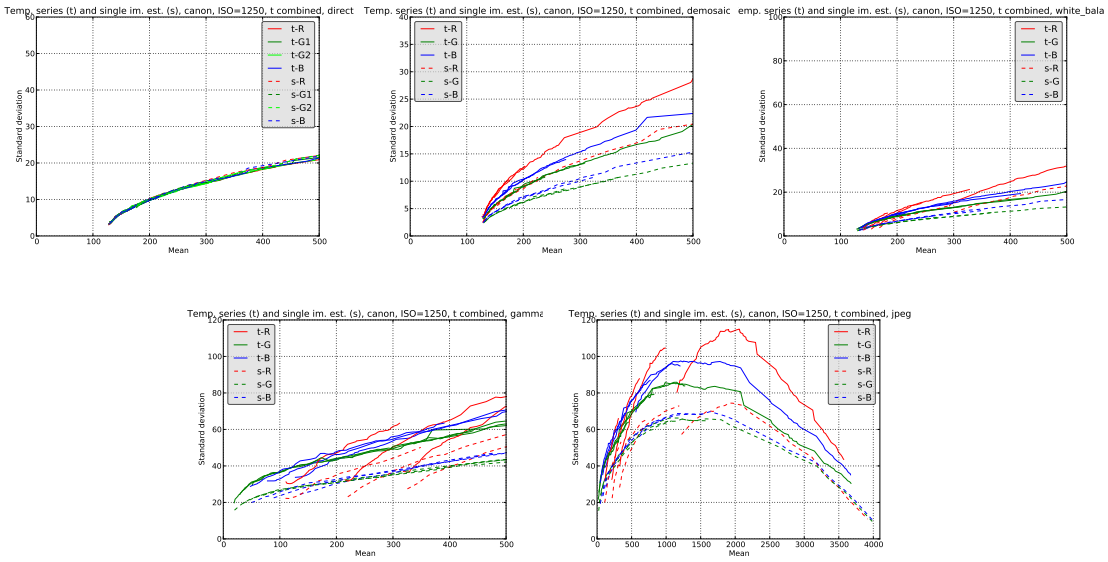


FIGURE 18. Noise curves overlapping for 1/30 s, 1/250 s, 1/400 s and 1/640 s exposure times for the Canon camera with ISO 1250. Up: raw image, after demosaicing and after white balance. Down: after the application of the tone curve, and after JPEG compression.

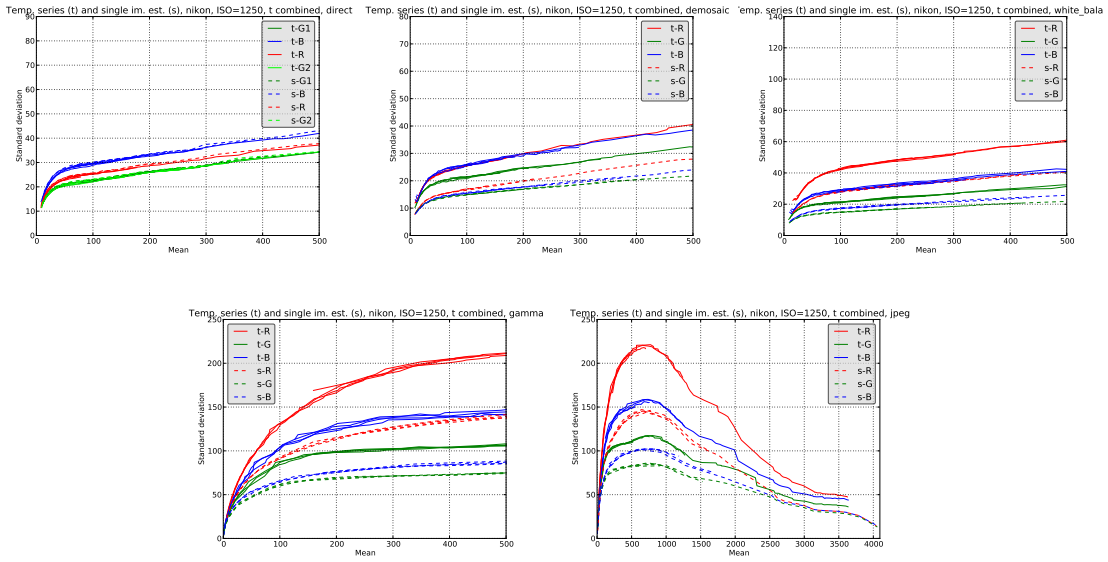


FIGURE 19. Noise curves overlapping for 1/30 s, 1/250 s, 1/400 s and 1/640 s exposure times for the Nikon camera with ISO 1250. Up: raw image, after demosaicing and after white balance. Down: after the application of the tone curve, and after JPEG compression.

Algorithm 13 Compute the average of overlapping noise curves.

MEAN_OVERLAPPING - Computes the average of the standard deviations coming from noise curves of different exposure times with the same ISO speed. **Input** X_t, X_y: noise curves vectors (intensity and standard deviation) from different exposure times and the same ISO speed. **Output g**: **u** with one quarter of its medium/high frequencies set to zero.

```
x_range = arange(0.0, 4096, 0.1)          ▷ The whole intensity range with 12 bits and step 0.1
Y_mean = zeros(num_channels, len(x_range))          ▷ Output array with the averaged stds.
for ch = 0 ... num_channels do
    x_idx = 0
    for x ∈ x_range do
        num = 0
        mean = 0.0
        for t ∈ range(4) do                          ▷ 4 is the number of captors at the CFA
            if min(X_t[t, ch, :]) ≤ x ≤ max(X_t[t, ch, :]) then          ▷ If intensity belongs to any curve
                idx = argmin_i (|X_t[t, ch, i] − x|)
                while X_t[t, ch, idx] > x do
                    idx = idx - 1
                end while
                if idx == num_bins - 1 then
                    idx = idx - 1
                end if
                                          ▷ The coordinates of the control points in between
                x_1, x_2 = X_t[t, ch, idx], X_t[t, ch, idx + 1]
                y_1, y_2 = Y_t[t, ch, idx], Y_t[t, ch, idx + 1]
                mean = mean + y_1 + (x − x_1) × (y_2 − y_1)/(x_2 − x_1)          ▷ Interpolation of the std.
                num = num + 1
            end if
        end for
        if num > 0 then                          ▷ If at least one overlapping noise curve
            mean = mean / num
            Y_mean[ch, x_idx] = mean                          ▷ Store final averaged std.
        end if
        x_idx = x_idx + 1                          ▷ Next intensity
    end for
end for
```

image **u** as the result of applying n times operator **S** to **u**. If **u** is a discrete pure Gaussian noise image with standard deviation σ, then $\mathbf{S}(\mathbf{u})$ has standard deviation $\frac{\sigma}{2}$. Indeed, if a block $\mathbf{W}$ contains pixels $\{p_1, p_2, p_3, p_4\}$ each one with variance σ^2, the variance of the mean of $\mathbf{W}$ is $\mathrm{Var}(\bar{\mathbf{W}}) = \mathrm{Var}\left(\frac{p_1+p_2+p_3+p_4}{4}\right) = \frac{1}{16}[\mathrm{Var}(u_1) + \mathrm{Var}(u_2) + \mathrm{Var}(u_3) + \mathrm{Var}(u_4)] = \frac{1}{16}[4\sigma^2] = \frac{\sigma^2}{4}$. The standard deviation is $\mathrm{Std}(\bar{\mathbf{W}}) = \frac{\sigma}{2}$, that is, the noise is divided by two at each subscale.

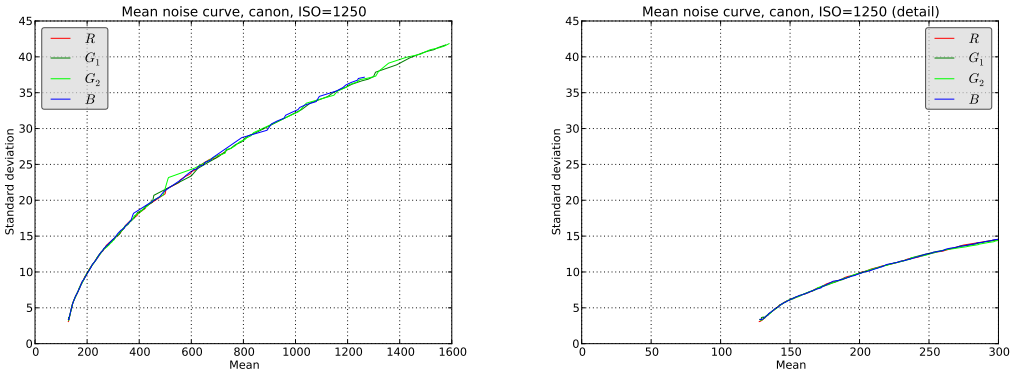


FIGURE 20. Mean noise curve for the Canon camera with ISO 1250. Since this noise curve was obtained from raw images, it can be used as the ground-truth of the noise of the camera for that ISO. On the right, a detail of the curve at the darkest pixels.

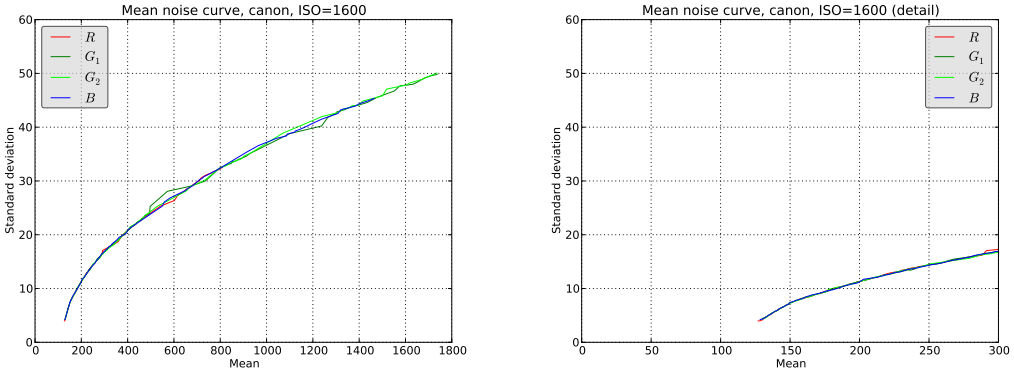


FIGURE 21. Mean noise curve for the Canon camera with ISO 1600. Since this noise curve was obtained from raw images, it can be used as the ground-truth of the noise of the camera for that ISO. On the right, a detail of the curve at the darkest pixels..

In this chapter we compare the autocorrelation matrix of a 10×10 pixels block of an rectangle in one snapshot of the calibration pattern with ISO 1600 and exposure time $1/250s$ for both the Canon and Nikon cameras. The autocorrelation function has support 11×11 pixels.

Figure 24 shows the autocorrelation matrices with support 11×11 with the calibration pattern obtained using the Canon camera. From up to bottom, the autocorrelation functions obtained at the raw, demosaiced, white balance, gamma correction and JPEG compression accumulated steps of the chain. From left to right, the measurements at the first, second, third and fourth subscales of the image. Note that the scale of the color bar changes at each processing step. Figure 25 shows the same for the Nikon camera. The following discussing is applicable to both cameras. At

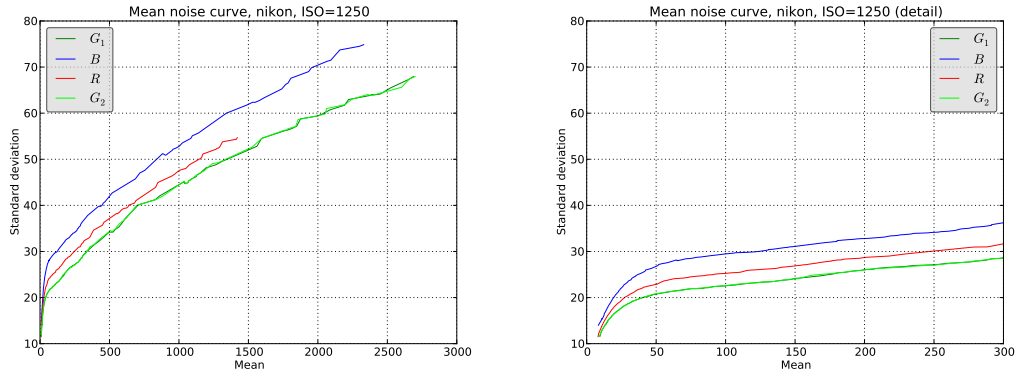


FIGURE 22. Mean noise curve for the Nikon camera with ISO 1250. Since this noise curve was obtained from raw images, it can be used as the ground-truth of the noise of the camera for that ISO. On the right, a detail of the curve at the darkest pixels.

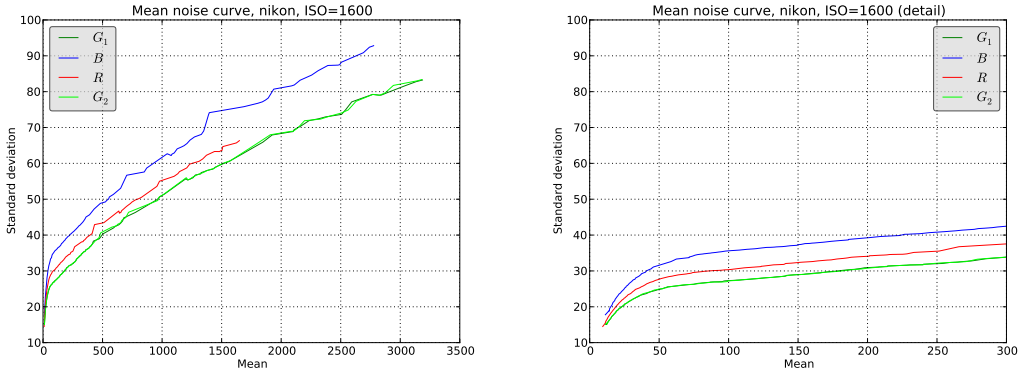


FIGURE 23. Mean noise curve for the Nikon camera with ISO 1600. Since this noise curve was obtained from raw images, it can be used as the ground-truth of the noise of the camera for that ISO. On the right, a detail of the curve at the darkest pixels.

the raw image step, it can be observed that the noise is not correlated at any scale. The central value at $[0, 0]$ is the only significant value and corresponds to the correlation of the signal with itself without shifting. Since a subscale just consists of computing the average of each disjoint set of 2×2, the autocorrelation values decrease as the number of scales increase. This can be observed no matter the step of the camera processing chain. The effect of demosaicing (by the Adams-Hamilton algorithm) is clearly noticeable, as it correlates strongly the pixels at distance 1 and slightly the pixels at distance $\sqrt{2}$. As expected, this effect is minimized at the second scale and beyond, due to the pixel averaging. Finally, the JPEG step result is similar to that obtained at the gamma correction. The value at $[0, 0]$ is lower, since the JPEG algorithm averages the red,

green and blue color channels in order to turn the RGB color space into $Y'C_BC_R$. Note that the autocorrelation pattern observed after the demosaicing step will remain along all the rest of the camera processing chain steps. The origin of the correlation observed in the JPEG image is the demosaicing step.

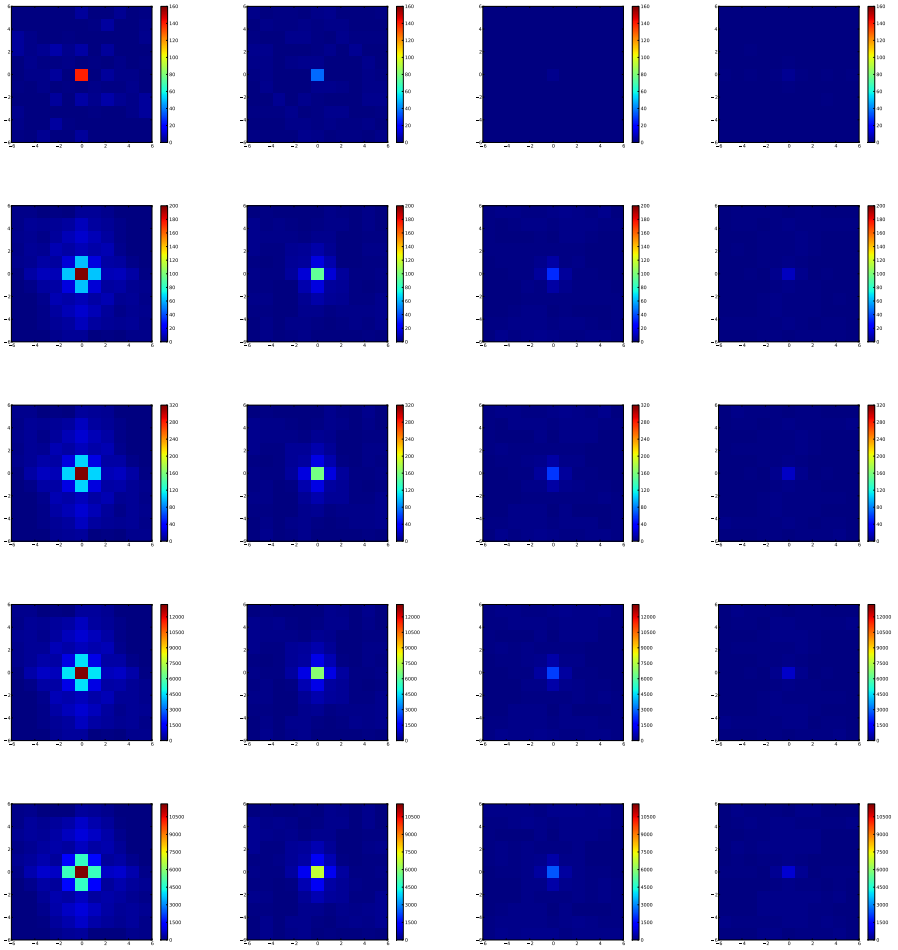


FIGURE 24. Autocorrelation functions for the Canon camera, ISO 1600 and exposure time 1/250s. From up to bottom: results of the steps raw, demosaicing, white balance, gamma correction and JPEG correction. From left to right: first, second, third and fourth scales. Note that the scale of the color bar changes at each processing step.

5. Conclusions

Under the same ISO speed and exposure time conditions, the noise curves obtained by different cameras differ. Some cameras do not pre-process at all the data acquired at the CCD or CMOS detector and therefore it would be possible to assume a Poisson model for the noise. However, in other cameras the data at the raw image has been already altered in an unknown way, thus

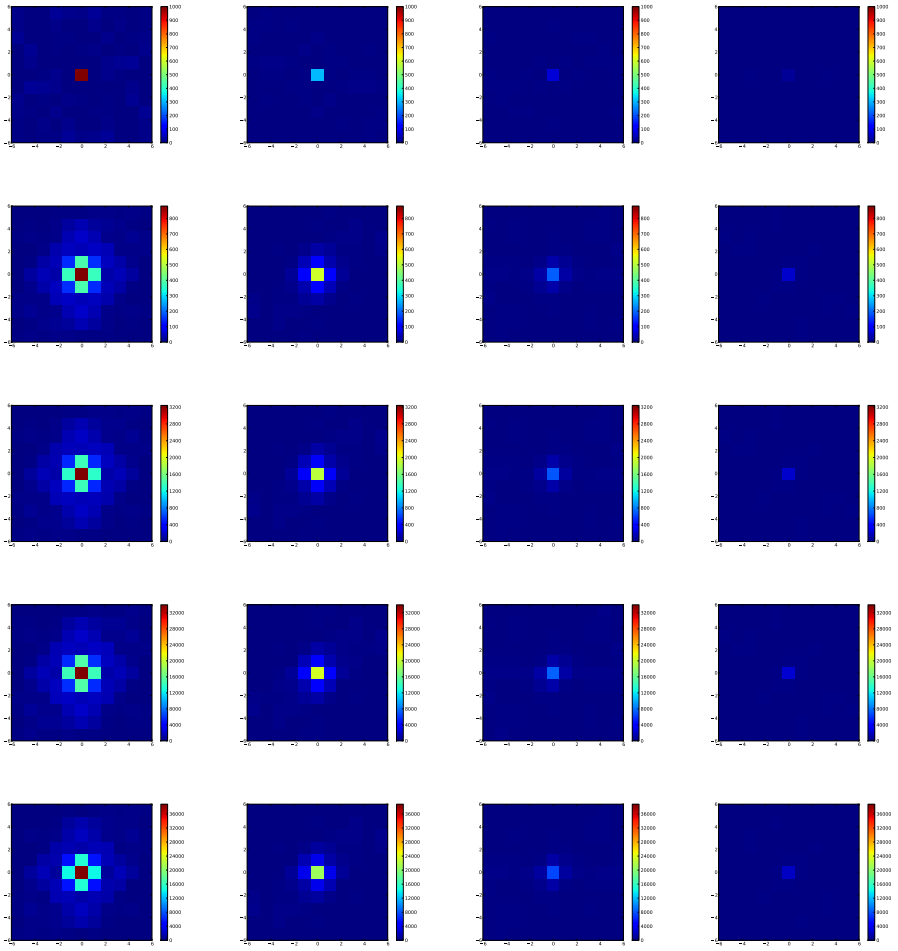


FIGURE 25. Autocorrelation functions for the Nikon camera, ISO 1600 and exposure time 1/250s. From up to bottom: results of the steps raw, demosaicing, white balance, gamma correction and JPEG correction. From left to right: first, second, third and fourth scales. Note that the scale of the color bar changes at each processing step.

making impossible to assume any model. Therefore, it is preferable to use non-parametric models that directly estimate a noise curve from the image itself with assuming a predefined model. The only necessary assumption of such estimation methods is that the noise is signal and frequency dependent. The frequency dependency is justified because the final JPEG encoding is equivalent to low-pass filtering the image (besides adding block artifacts and shot noise). Since the quantization matrix of the JPEG encoder depends on the unknown quality parameter, the only way to find a proper model for the noise is again to estimate the noise at each frequency, avoiding any predefined model for the JPEG filter. In general, denoising algorithms need an accurate estimation of the noise to properly denoise an image. Since the exact transformations that have been applied to the

noisy image are unknown, assuming a model is too risky and it is preferable to get a profile of the noise depending of the intensity and the frequency from the noisy image itself.

In raw images both the spatial and temporal estimations matched accurately, but after the next step (demosaicing), they did not. The explanation for this is that demosaicing correlates the noise (thus, it becomes frequency-dependent) and the signal-dependent noise model is not enough to measure the noise in these kind of real images. Most of the signal-dependent noise estimators assume that the variance at any frequency is the same, and thus they compute an average of the variances measured at the high-frequencies (where the noise dominates over the geometry of the image). After demosaicing, this does not hold. In Chapter 4 a method for estimating both intensity and frequency dependent noise is presented.

CHAPTER 4

Multiscale estimation of intensity and frequency dependent noise

The camera calibration and the image processing chain that generated a given image are generally no more accessible to the receiver. This happens for example with scanned photographs and for most JPEG images. These images have undergone various nonlinear contrast changes and linear and nonlinear filters. To deal with remnant noise in such images, we introduce a general non parametric intensity and frequency dependent noise model. We demonstrate by simulated and real experiments that this model, which requires the estimation of more than 1000 parameters, performs an efficient noise estimation.

The proposed noise model is a patch model. Its estimation can therefore be used as a preliminary step to any patch-based denoising method. Our noise estimation method introduces several new tools for performing this complex estimation. One of them is a new sparse patch distance function permitting to detect efficiently pure noise patches. We also show a new way to avoid the bias of noise estimation methods based on the use of low variance patches.

A validation of the noise model and of its estimation method is obtained by comparing its results to ground-truth noise curves for both raw and JPEG-encoded images, and by visual inspection of the denoising results of real images. A fair comparison to the state of the art is also performed.

This chapter presents a complex noise model that makes it possible to measure the noise even in images where the noise is highly correlated, as in the case of JPEG-encoded images. It is the final step of the study on noise estimation presented in this thesis, that begins with the simple but absolutely unrealistic simulated homoscedastic white Gaussian noise (Chapter 1), later the signal-dependent noise, which is only useful for raw images (Chapter 2), and finally the intensity-frequency dependent noise that is presented in this chapter. Chapter 8 describes a new denoising algorithm able to denoise even JPEG-encoded images using the noise estimation algorithm presented here, where medium and specially low-frequency noise remains after compression.

1. Introduction

The noise initially present in a raw digital image is transformed at each step of the processing chain of the camera. When acquired at the focal plane in a color filter array, the noise is Poisson distributed, intensity-dependent and frequency-independent. Yet the image at the Color Filter Array (CFA) is possibly saturated, which infringes the simple linear dependency of the noise variance with the intensity [**7**]. Even without saturation, the variance of the noise may not follow the linear

model, depending on the characteristics of the detector [**8**]. At the very first step of the camera processing chain a demosaicing algorithm [**81, 80**] must be applied to get a color image from the raw mosaic acquired at the CFA. This causes the noise to be spatially correlated. It therefore becomes frequency-dependent. The colored noise caused by the demosaicing undergoes further nonlinear transforms such as the white balance and a gamma-correction. Finally, JPEG-encoding [**79**] accentuates the frequency dependence, as JPEG encoding applies a frequency-dependent quantification matrix to the coefficients of the 8×8 DCT-II blocks of the image. Therefore, the remaining noise in a JPEG image is signal dependent and highly correlated. It generally contains little high frequency noise, as the quantization removes the image high frequencies. But it too often still contains strong noise at the low and medium DCT block frequencies. This annoying noise is hard to evaluate and to remove.

Such noise characteristics are observed in modern digital images, but also in scans of old photographs, which contain chemical noise. The assumption that their final observed noise is both signal and frequency dependent (SFD) is clearly a minimal model. The purpose of this chapter is to develop a method for estimating such SFD noise, and to validate it by comparing the estimated results to the appropriate ground-truth.

Little has been written on SFD noise estimation from a single digital image. A method estimating a "JPEG compression history" from a single image can be found in [**82**]. The noise estimation method for JPEG images proposed in [**52**] estimates a signal dependent noise level which is not frequency dependent and therefore only gives a "noise level". One of the most complete attempt to estimate a general noise model is contained in the blind denoising method [**37**], which estimates multiscale noise covariances for noise wavelet coefficients. This model is nevertheless not signal dependent. The recent method for estimating frequency dependent noise on patches in [**83**] is probably the closest to our endeavor. We will detail the points in common and the proposed extensions and improvements of this method. To the best of our knowledge, no method has proposed so far to estimate a general SFD noise patch model. The situation is nonetheless favorable, as most homoscedastic noise estimation algorithms are actually patch based [**2, 3, 4, 5, 6, 1**], and can therefore be adapted to measure SFD noise models on patches.

Our plan is this Chapter follows. Section 2 develops the principles of blind noise estimation, defines the signal and frequency dependent (SFD) model, and explains progressively how to estimate it. Section 3 details the proposed algorithm. Section 4 performs a comparison of the method with the current state of the art. Section 5 is the core of the chapter. It validates the sufficiency of the SFD model for JPEG images by calculating a ground truth SFD model and checking that it is indeed obtained by the algorithm. This section also performs a final consistency check by displaying denoising results obtained with a multiscale version of the NL-Bayes algorithm [**84, 22**]. To that aim, it compares the estimated noise model before and after denoising. Section 5.1 contains the conclusions.

2. Blind noise estimation principles

This chapter is a progressive presentation of patch based noise modeling. It gives a definition of the SFD noise model and discusses how to estimate it with minimal bias from well chosen blocks extracted from the image itself.

2.0.0.1. The search for blocks with low variance. The first principle adopted by most recent methods is to look for small image blocks that contain only noise. Indeed, all image blocks contain a sum of signal and noise. Being independent, the variances of noise and signal add. This leads to the conclusion that blocks with minimal variance extracted from the image are likely to contain no signal, and therefore only noise. In the algorithms adopting this strategy, all image blocks are ordered by their variance, and the noise estimation is performed on their lowest quantile (typically taking the 0.5% blocks with lowest variance).

2.0.0.2. Compensating the bias. Computing a median of these variances gives an estimate of the noise variance. Yet, this estimation is biased. Indeed, if the image were a pure white noise realization, the above method would estimate the noise variance on the blocks with lowest variance, leading to an obvious underestimation. This bias can nevertheless be compensated by a multiplicative correction factor learnt on a white noise image of the same size.

2.0.0.3. Using instead block differences. A recent method **[83]** has proposed a clever way to extend the low variance block method by involving the image self similarity. The idea is to associate with each block its most similar block in a neighborhood. Then, assuming that this similarity is essentially caused by the signal, the difference of both blocks is assumed to give a pure noise block, with twice the variance of the original block. Then, again, only a small quantile of those "noise blocks" should be retained for the final noise estimation. In practice, most of the selected blocks correspond to flat zones (as we shall see in the comments of Figure 1). If the image lacks flat zones, this selection of block differences may nevertheless also contain differences of non flat patches with similar geometry.

2.0.0.4. Dealing with signal dependency. The emission of photons by a physical body is a random process that can be modeled with a Poisson distribution, for which the mean is equal to the variance. Thus, the image formed at the CFA contains noise that depends on the intensity of the underlying image. This intensity dependence of the noise model begins at the very first step (acquisition at the detector: the raw image) and remains until the last step of the camera processing chain (JPEG encoding). The value of the tabulated gamma correction function is generally unknown. Even when this information is available, the CCD or CMOS detector do not necessarily follow a simple linear relation intensity/variance **[8]** when acquired at the detector.

Therefore, the noise estimation algorithm must estimate intensity-dependent noise. A famous alternative is to transform the data to turn it into homoscedastic noise via an Anscombe transform **[85, 43]**. Yet an Anscombe transform is only possible if we deal with raw images. In the general setting of a signal dependency that can be different for every frequency, there is no other way to estimate the signal dependent noise model than dividing the set of blocks into disjoint bins,

each for a different intensity and to estimate a separate frequency-dependent noise model on each intensity bin.

Fortunately, as recalled in **[1, 9]**, it is possible to adapt most patch-based homoscedastic noise estimation methods **[2]** **[3]** **[4]** **[5]** **[17]** **[15]** to measure intensity-dependent noise, by simply splitting the list of input blocks into sets of blocks disjoint in mean intensity (*bins*).

2.0.0.5. Dealing with frequency dependency: the DCT diagonal assumption. According to the above considerations, the SFD noise estimation should start by finding a set of pure noise patches for each intensity bin. Our main assumption will be that the unknown linear and nonlinear transforms that have been applied to the image can be approximated by a diagonal operator on the DCT patch coefficients. There are two arguments in favor of this diagonality. First, it is easily checked that every linear real symmetric filter applied to an image is a diagonal operator on the DCT transform. This observation is actually exact for a global DCT transform, but remains approximately true for the (local) block DCT. Second, we mentioned that JPEG 1985 also is a diagonal (nonlinear) operator on the DCT coefficients. The demosaicing operation itself is more complex, but a good demosaicing algorithm must avoid creating structure in noise, and must therefore be close to a linear isotropic filter, which again is a diagonal operator. Nevertheless, these arguments are no end proofs. The DCT diagonal assumption can only be conjectured and must be verified empirically, as we shall do in the experimental chapters. Under this assumption, a *patch noise model* is described by the variances of its DCT coefficients.

2.0.0.6. Definition of the SFD noise model and its evaluation. The proposed signal and frequency dependent (SFD) noise model follows from the above observations, that also hint at an estimation algorithm. It is enough to find sufficiently many noise patches in a given image, and to apply to them a DCT before measuring the variance of their DCT coefficients. In that way the SFD model is defined (and can be empirically estimated) for each patch size w as a map:

$$(i, j, b) \in [[0, w-1]]^2 \times [[0, B-1]] \to \sigma(i, j, b),$$

where (i, j) is the DCT frequency, w the block size, B the number of color level bins, and $\sigma(i, j, b)$ the observed noise standard deviation for this particular frequency and bin. This model must be estimated independently for each color channel. The SFD model has therefore many parameters. For example for a (minimal) block size $w = 4$, $B = 25$ bins and a three channels, $16 \times 25 \times 3 = 1200$ parameters must be estimated. This also explains why a more sophisticated model can hardly be envisaged. It might for example seem natural to estimate a whole covariance matrix for the noise patches, but this would require many more samples than those available in a single image. (Nevertheless, this extension might be considered for video noise estimation.)

The method proposed in **[83]** is the closest to the above considerations. It computes a frequency dependent noise model for the purpose of estimating homoscedastic frequency dependent noise. We shall now propose several extensions and improvements to make it fit for accurate SFD estimation, when many more free parameters are being estimated. Our goal is indeed to extend the frequency dependent model to a general enough model permitting to cope with most images.

We saw that introducing a signal dependence in the noise model is a necessary extension. The next improvement will be to define an unbiased block distance as explained in the next paragraph.

2.0.0.7. A sparse semi-distance between blocks. The frequency dependent noise estimation of **[83]** proposed to generate pure noise blocks by pairing similar blocks and making their difference to eliminate the signal. Their distance is evaluated in this method on a random choice of 50% of the block DCT coefficients. We shall explain the bias caused by this method and try to alleviate it. We propose to compare blocks with a sparse semi-distance which we introduce now. Given an image DCT block $\tilde{m}_1$, the idea it to only use its $w^2/4$ coefficients with largest absolute value for comparing them with the coefficients at the same frequencies in another block $\tilde{m}_2$. According to the transform thresholding principle **[73]**, these high-energy coefficients belong with high probability to the signal and not to the noise. Indeed, the $w \times w$ DCT-II matrices of the blocks in a natural and not noisy image are generally sparse, and the signal is mainly contained in some low and medium frequency coefficients. Thus, we can assume that the semi-distance[1] computed with that set of coefficients is not guided by the noise in the patch, but by the image geometry.

To compute the sparse semi-distance between a reference block $\tilde{m}_1$ and a candidate $\tilde{m}_2$, the absolute values of the DCT coefficients $\mathbf{A}[i,j], [i,j] \in [0, w-1]^2$ of $\tilde{m}_1$ are sorted. The decreasing sorting indices are stored in $\mathbf{Q}$. The sparse semi-distance is then computed based on the $w^2/4$ DCT

Algorithm 14 Sparse distance algorithm

1: **Input :** $\tilde{m}_1$: patch of size $w \times w$
2: **Input :** $\tilde{m}_2$: patch of size $w \times w$
3: **Output :** sparse distance between $\tilde{m}_1$ and $\tilde{m}_2$.
4: Compute the absolute value of the DCT-II coefficients of $\tilde{m}_1$, $\mathbf{A}[i,j] = \{|\tilde{m}_1[i,j]| : [i,j] \in [0, w-1]^2\}$.
5: Sort $\mathbf{A}$ from the highest to the lowest value and put the sorting indices in $\mathbf{Q}$; $\mathbf{Q} = \text{argsort}(\mathbf{A})$.
6: Compute the SD on the first $w^2/4$ sorting indices in $\mathbf{Q}$: $\text{SD}_{\tilde{m}_1,\tilde{m}_2} = \sum_{q=0}^{w^2/4-1} (\tilde{m}_1[\mathbf{Q}_q] - \tilde{m}_2[\mathbf{Q}_q])^2$.

coefficients in $\tilde{m}_1$ with the largest absolute value. This computation is summarized in Algorithm 14.

2.0.0.8. Should noise be estimated on block differences? The proposal made in **[83]** to choose the differences $\frac{1}{\sqrt{2}}(\tilde{m}_1 - \tilde{m}_2)$ as noise block samples is quite tempting. Indeed, the sparse semi-distance ensures that in this difference the part of the signal that was contained jointly in $\tilde{m}_1$ and $\tilde{m}_2$ has been canceled, thus giving a pure noise sample. Nevertheless, we found that it was better not to operate this subtraction, and that the noise estimation based on the blocks $\tilde{m}_1$ such that $\tilde{m}_1 - \tilde{m}_2$ is minimal was more accurate than the estimation based on the normalized differences $\frac{1}{\sqrt{2}}(\tilde{m}_1 - \tilde{m}_2)$. Experimental evidence confirms that the subtraction is advised only in the case where the image is known *a priori* to be made mainly of textures and to lack flat or smooth zones. We can anticipate the following explanation. Consider an image of pure Gaussian noise of

[1]We call it semi-distance and not distance, as it is not symmetric and does not satisfy the triangle inequality.

variance σ^2. Given a block $\tilde{m}_1$, and its most resembling block $\tilde{m}_2$, the similar DCT coefficients in both $\tilde{m}_1$ and $\tilde{m}_2$ are anyway due to noise. Thus the (small) block $\tilde{m}_1 - \tilde{m}_2$ generally looses low frequency coefficients belonging to the noise. In this situation, the estimated noise can be drastically distorted and underestimated. On the other hand, it turns out that the blocks $\tilde{m}_1$ such that the distance $\tilde{m}_1 - \tilde{m}_2$ is minimal are a better choice than blocks with minimal variance. Indeed, the minimal difference criterion is less biased than the minimal variance criterion. This empirical statement will be checked experimentally in Section 4. This explains why we shall select the best noise blocks $\tilde{m}_1$ based on their difference with the most resembling block, but keep $\tilde{m}_1$ for the final noise estimation.

2.0.0.9. The multiscale approach to estimate low frequency noise. The selection of noise blocks faces another dilemma: on the one hand it is much easier to find small (typically 4×4) blocks containing only noise, than larger (e.g. 8×8) blocks. Yet, small blocks do not permit to estimate the noise low frequencies. Such low frequencies are prominent in JPEG images because of the demosaicing, which creates sometimes long range correlation, and because of the JPEG compression itself. So noise can appear in large spots, as shown in Figure 9. This image is the result of convolving an image of pure Gaussian noise with mean 127 and $\sigma = 50$ with the kernel $\mathcal{G}$ in Equation 4. A multiscale approach solves the dilemma. Defining the input noisy image as the image at the *first scale*, a second scale image can be obtained by a 2-subsampling. Estimating again noise in this subsampled image permits to catch the noise low frequencies.

2.0.0.10. Denoising. Once the STD of the noise at each intensity b and each frequency $[i, j] \in [0, w-1]^2, [i, j] \neq [0, 0]$ is known, it is possible to fully characterize the noise by its covariance matrix. Thus, it is possible to denoise the image by obtaining the covariance matrix of the noise at each scale, and then denoising each scale using the obtained noise profile. Since the number of samples is divided by 4 after each subscaling, the number of pixels of the input image is a limiting factor (highly correlated noise cannot be measured in small images). The Nonlocal Bayes **[84, 22]** algorithm will be used for that purpose, as it only requires a knowledge of the noise covariance matrix.

3. Noise estimation algorithm

Our proposed SFD noise estimator (Algorithm 15) follows from the considerations of the preceding chapter.

In the first step, $w \times w$ (typically, $w = 8$ or $w = 4$) overlapping blocks are extracted from the input noisy image, and their 2D orthonormal DCT-II is computed. For each of these DCT blocks $\tilde{m}_1$, the most similar block $\tilde{m}_2$ in a fixed neighborhood is found. The candidate blocks are at a spatial range belonging to the interval $[r1, r2] = [4, 14]$ (the spatial distance is defined as the maximum between the horizontal and vertical distances of the block centers). Given a block $\tilde{m}_1$ at spatial position $[x_1, y_1] \in [0, N_x - w] \times [0, N_y - w]$, the candidate blocks are searched only at positions $[x_2, y_2] : x_2 > x_1 \wedge y_2 > y_1$ to avoid repeating matching pairs. The similarity between $\tilde{m}_1$ and any other block in its neighborhood is evaluated with the Sparse Distance (SD) function

(Algorithm 14), designed to avoid the interference of the noise in the similarity evaluation. For each $\tilde{m}_1$ in the image, a corresponding most similar block $\tilde{m}_2$ is found, and $\tilde{m}_1$ and the sparse distance $\mathrm{SD}_{\tilde{m}_1,\tilde{m}_2}$ are stored in list $\mathbf{L}$.

Algorithm 15 Noise estimation algorithm

1: **Input :** Noisy image $\mathbf{u}$ of size $N_x \times N_y$ pixels.
2: **Input :** $w \times w$ size of the block in pixels.
3: **Output :** SFD noise curve $\tilde{\sigma}$.
4: Extract from the input image $\mathbf{u}$ of size $N_x \times N_y$ all possible $M = (N_x - w + 1)(N_y - w + 1)$ overlapping $w \times w$ blocks $\mathbf{B}_k$ and compute their 2D orthonormal DCT-II, $\tilde{\mathbf{B}}_k, k \in [0, M-1]$.
5: Set $\mathbf{L} = \emptyset$ ▷ Empty set.
6: **for** each DCT block $\tilde{m}_1 \in \tilde{\mathbf{B}}$, **do**
7: Find the block $\tilde{m}_2$ that minimizes the sparse distance $\mathrm{SD}_{\tilde{m}_1,\tilde{m}2}$ (see Algorithm 14). Consider only $\tilde{m}_2$ blocks whose horizontal and vertical distance with respect to $\tilde{m}_1$ belongs to the interval $[r_1, r_2] = [4, 14]$.
8: Extract from $\tilde{m}_1$ the mean of m_1.
9: Add block $\tilde{m}_1$ and the associated sparse distance ($[\tilde{m}_1, \mathrm{SD}_{\tilde{m}_1,\tilde{m}2}]$) to list $\mathbf{L}$.
10: **end for**
11: Classify the elements of list $\mathbf{L}$ into disjoint bins according to the mean intensity of the blocks **[1, 75]**. Each bin contains (with the exception of the last) 42000 DCT blocks.
12: **for** each bin, **do**
13: Obtain the set $\mathbf{S}_p$ made by those DCT blocks inside the current bin whose SD is below the p-quantile, with $p = 0.005$.
14: Assign to the current bin the intensity I (Equation 3).
15: **for** each frequency $[i, j]$ with $[i, j] \in [0, w-1]^2, [i, j] \neq [0, 0]$, **do**
16: Compute the (biased) STD of the noise at the current bin and frequency $[i, j]$ using the MAD estimator (Equation 1).
17: Correct the biased $\hat{\sigma}[I][i, j]$ and obtain the final STD estimate (Equation 2).
18: **end for**
19: **end for**

In order to estimate intensity-dependent noise, the set of blocks in $\mathbf{L}$ is split according to the mean intensity of the blocks. Each division of $\mathbf{L}$ is called a *bin* and contains only the blocks (and their associated SD) whose mean intensity belongs to a certain range. The bins are disjoint in intensity and their union gives the whole intensity range of the image.

At each bin the list $\mathbf{S}_p$ is filled in with the blocks whose associated SD is below the p-quantile, with $p = 0.005$. The value of p must be kept small, to ensure that $\mathbf{S}_p$ contains blocks for which it was possible to find another block with minimal SD between them.

In step 9, block $\tilde{m}_1$ and the sparse distance $\mathrm{SD}_{\tilde{m}_1,\tilde{m}_2}$ are stored in list $\mathbf{L}$.

Finally, for the DCT blocks in $\mathbf{S}_p$, the standard deviation (STD) according to each frequency $[i,j] \in [0, w-1]^2, [i,j] \neq [0,0]$ is computed using the MAD estimator (Equation 1).

$$\begin{aligned} \hat{\sigma}[I][i,j] &= \mathrm{MAD}(\mathbf{S}_p) \\ &= \underset{\tilde{n} \in \mathbf{S}_p}{\mathrm{median}} \left(\left| \tilde{n}[i,j] - \underset{\tilde{m} \in \mathbf{S}_p}{\mathrm{median}} \left(\tilde{m}[i,j] \right) \right| \right). \end{aligned} \tag{1}$$

Equation 2 gives the correction factor for the STD depending on the size of the blocks, for $p = 0.005$. A correction of the STD is needed because MAD is a biased estimator of the STD and also because the available number of coefficients to compute the (sample) variance is finite and thus biased. To obtain the correction factors, we added simulated homoscedastic noise of STD $\sigma = 5$ to a synthetic image of a calibration pattern with large flat zones of several different grayscale intensities. The biased STD $\hat{\sigma}$ was estimated with our algorithm and compared with $\sigma = 5$. The ratio $\sigma/\hat{\sigma}$ gives the correction factor. In Equation (2) we only give corrections for $w = 4$ and $w = 8$, but of course more correction factors may be tabulated if needed.

$$\tilde{\sigma}[I][i,j] = \begin{cases} 1.775 \times \hat{\sigma}[I][i,j] \text{ if } w = 4; \\ 1.677 \times \hat{\sigma}[I][i,j] \text{ if } w = 8. \end{cases} \tag{2}$$

The corresponding intensity I is computed with Equation (3), as the median of the mean intensities under the p-quantile of blocks SD.

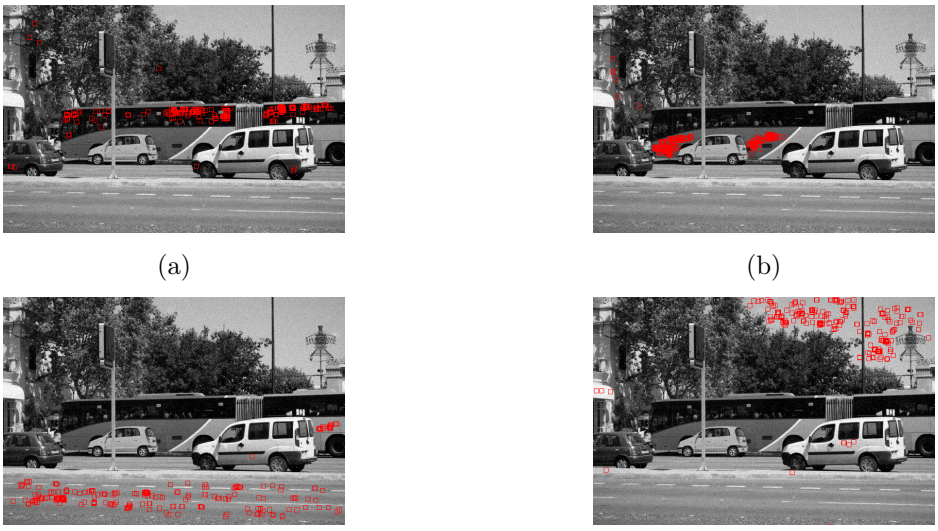
$$I = \underset{\tilde{m} \in \mathbf{S}_p}{\mathrm{median}} \left(\tilde{m}[0,0]/w \right) \tag{3}$$

The size of the block depends on the application of the noise estimation. A small block, say 4×4 ($w = 4$) is preferable for highly textured images, as the probability of capturing flat zones decreases with the size of the block. However, a larger block gives a more accurate frequency information about the noise and is more consistent with JPEG compression. As we mentioned the solution to this dilemma is to perform a two scale estimation with small blocks.

4. Comparison

In this chapter we compare our proposed noise estimation method with the best state-of-the-art paper on estimation of correlated noise [**83**]. This method is designed to estimate homoscedastic noise, but since it is patch-based, it is relatively easy to adapt it to intensity-dependent noise. As explained in Section 2, being able to estimate intensity-dependent noise is a mandatory step for any noise estimation methods that intends to be used with real images, where the noise is highly correlated after the demosaicing step.

We introduced the sparse distance function to avoid the interference of noise in the similarity measure between two noisy blocks. Figure 1 shows the blocks selected by Algorithm 15 using the sparse similarity function of Algorithm 14. If the image contains flat or smooth zones, the algorithm



(a) (b)

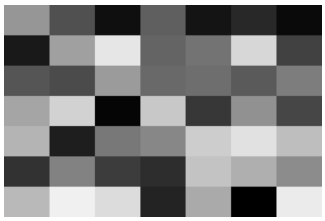
(c) (d)

FIGURE 1. Blocks selected by Algorithm 15 using the sparse similarity function of Algorithm 14, using the *traffic* test image shown in Figure 3 after adding homoscedastic noise of STD = 10, for the bins #0 (a), #2 (b), #4 (c), and #6 (d) (7 bins are used). In a highly textured image, where is difficult to find smooth zones, the sparse similarity function will choose patches with similary geometry.

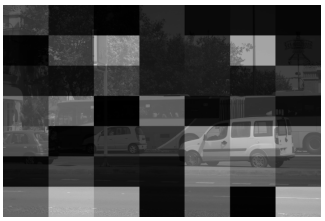
will prefer them to estimate the noise, since they give the maximum similarity. Finally, in **[83]** the difference between similar blocks is stored for subsequent noise estimation, but as explained in Section 2, it is better not to compute the difference, to prevent the possibility of choosing blocks that are similar because of the noise and thus giving underestimations.

Now we will pass to compare and discuss the influence of two decisions taken in the design of our proposed SFD noise estimation: the subtraction or not of similar blocks in the list of DCT blocks under the 0.005-quantile and the performance of the new similarity function (Algorithm 14). In order to measure the influence of textures in the performance of the compared noise estimation methods, we used a synthetic noise-free calibration pattern (see Fig.2, left). Since calibration patterns lack texture, it is easy to find flat zones for which any variation of the intensity is due to the noise. Thus most noise estimation methods are expected to perform optimally on such images. To simulate the effect of textures, we considered an image that combines both the calibration pattern and a noise-free image. For example Figure 2 (right) shows the noise-free image obtained by adding 80% of the intensity of the calibration pattern and 20% of a noise-free textured image *traffic*. As both combined images are noise-free, the result is still noise-free, but textured.

To show the influence of textures in the noise estimation depending on the method, we added simulated intensity-dependent noise of variance $\sigma^2 = 100 + 7\mathbf{u}$ to the combination of the calibration pattern image with several noise-free images and then estimated the Root Mean Squared Error (RMSE) along all frequencies and intensity bins ($\mathbf{u}$ is the pixel intensity of the combined image). The level of texture of the image analyzed was controlled by a parameter α, the combination being



Noise-free calibration pattern



Combined with *traffic*

FIGURE 2. On the left, synthetic noise-free calibration pattern. On the right, the weighted sum of the calibration pattern with the noise-free test image *traffic* (with weights 0.8 for the calibration pattern and 0.2 for the *traffic* image. Since both combined images are noise-free, the result is still noise-free, but textured.

Bag

Computer

Flowers

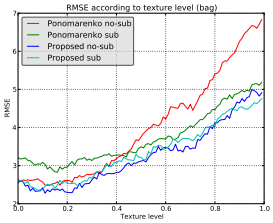
Traffic

FIGURE 3. Noise-free images used to measure the robustness of the methods to the presence of texture. Each image is 704×469 pixels.

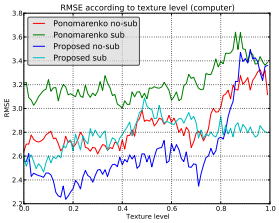
$\alpha\mathcal{P} + (1-\alpha)\mathcal{T}$, where $\mathcal{P}$ is the calibration pattern and $\mathcal{T}$ the noise-free image that brings the texture. We used the four noise-free images shown in Figure 3.

Figure 4 shows the RMSEs obtained for the test images in Figure 3 using 8×8 blocks. In the horizontal axis, the value of $\alpha \in [0, 1]$ (the texture level) and in the vertical axis, the RMSE along all frequencies and intensity bins.

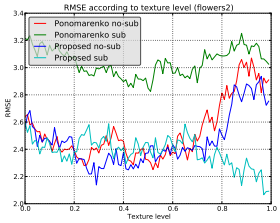
We compared the adaptation to intensity-dependent noise of the Ponomarenko et al. method **[83]** (state-of-the-art in frequency-dependent noise estimation using patches) and our method, with two variants for each method: subtracting the blocks under the MSE quantile and not subtracting them (see the discussion of Section 2). It can be observed that, regardless of the texture level, it is almost systematically better not to subtract similar blocks before estimation. Compare the Ponomarenko method (labeled *Ponomarenko sub*) with the variant which does not subtract similar blocks (labeled *Ponomarenko no-sub*): the estimation obtained with subtraction has a lower RMSE



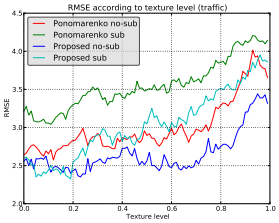
Bag



Computer



Flowers



Traffic

FIGURE 4. RMSEs obtained for the test images in Figure 3 using 8×8 blocks. In the horizontal axis, the value of $\alpha \in [0, 1]$ (the texture level) and in the vertical axis, the RMSE along all frequencies and intensity bins. We compared the adaptation to intensity-dependent noise of the Ponomarenko et al. method **[83]** and our method, and two variants for each method: subtracting the blocks under the low quantile and not subtracting them (see the discussion in Section 2). In general, avoiding the subtraction is the best option and it is what we propose in the presented method.

than when performing subtraction. Only in the case of extremely highly-textured images (as in the case of the *bag* image), the subtraction brings to a lower RMSE. The plots also compare the proposed method (labeled *Proposed no-sub*) with the variant that subtracts the blocks (labeled *Proposed sub*). It can be clearly seen that the proposed method gives a lower RMSE thanks to the use of a better similarity function (Algorithm 14. The new similarity function is less affected by textures, since it measures the difference between blocks using only the coefficients with most energy. This coefficients are related to the geometry of the image and are not biased by other coefficients that carry information from the noise (as it happens in **[83]**). In the plot of the *Bag* image it can be observed that for high values of α, the RMSE of the variant that does not subtract the blocks is similar to the variant subtracting the blocks.

Depending on the image (*computer*, *flowers*), the variant subtracting the blocks is slightly better than the variant without subtraction, for high values of α, but only for highly-textured images, as shown in Figure 4. But the proposed method has a better overall RMSE.

5. Validation with ground truth JPEG noise

The SFD noise estimation method gives an estimation of the standard deviation (STD) of the noise that depends both on the intensity and frequency in a single image. It uses the observation of blocks at many spatial locations and will therefore be called in this chapter the *spatial* estimation, to match it with the ground truth *temporal* estimation. We validated the spatial estimation method by taking raw and JPEG photographs with a given camera. The value of the spatially estimated STD *on a single image* should match the ground-truth STD for that camera for the configured ISO speed [**7**], obtained from multiple frames. Note that with *JPEG images* we do not refer to the result of a mere JPEG compression, but to the result of the whole camera processing chain applied the raw image acquired at the focal plane of the camera on the CCD (or CMOS), including demosaicing, white balance, gamma correction, and the final JPEG encoding.

For that purpose, we took a sequence of pictures of the same scene taken with fixed camera position and constant lighting. Under these conditions, any variation of the intensity in any pixel through the sequence is only attributable to the effect of the noise. It is therefore possible to build a ground-truth noise curves for both raw and JPEG-encoded images, associating with each observed mean signal value the corresponding STD of its observed samples. Similarly, by *frequency noise curve* we mean a numerical function associating with each value of the block mean the STD of the DCT coefficient of the noise at that frequency. *Thus, there are as many noise curves as DCT coefficients.* To obtain such curves, instead of measuring the variation of the intensity of the pixels in a fixed position along the sequence, we consider all M overlapping $w \times w$ blocks in the image, compute their orthonormal DCT-II, and measure the variance at the intensity of the bin and frequency $[i, j] \in [0, w-1]^2, [i, j] \neq [0, 0]$ along the coefficients of the blocks at the same spatial position and with varying image index.

The noise curve obtained this way for *each DCT frequency* will be called the *temporal* estimation and can be used as a ground-truth to compare with the spatial estimation. Even if a noise model for JPEG images has never been proposed in the literature, it is therefore possible to obtain reliable empirical curves for JPEG images. To obtain them, it suffices to JPEG-encode each image of the snapshot with the same quality parameter, and to apply the described procedure.

The objective of this chapter is to verify that the spatial STD measured at any frequency $[i, j] \in [0, w-1]^2, [i, j] \neq [0, 0]$ using the algorithm in Section 3 coincides with the STD of the temporal series measured only at that frequency for all intensities. To build the temporal STD noise curve we used 100 snapshots of the same calibration pattern, for both raw and JPEG-encoded images. In principle, any image might be used to get the temporal STD of the noise, but it is preferable to use an object with large flat regions of different gray levels, in order to avoid the effect of textures in the temporal estimation. To be robust to outliers (the edges between the large flat zones), we considered only the 0.05-quantile [**17**] of the STD estimations that was corrected afterwards to obtain an unbiased estimate.

The procedure to compute the ground-truth curve for JPEG-encoded images for frequency $[i, j] \in [0, w-1]^2, [i, j] \neq [0, 0]$ from a set of H images is detailed in Algorithm 16.

Algorithm 16 Algorithm to obtain the SFD ground-truth noise curve from a sequence of images

Input : Sequence of JPEG (or raw) images.
Input : H number of images in the sequence.
Input : $N_x \times N_y$ the size of each image in the sequence, in pixels.
Output : Ground-truth noise curve $\tilde{\sigma}$.

Set $M = (N_x - w + 1)(N_y - w + 1)$ the number of overlapping blocks.
Set $\mathrm{E1} = \mathrm{E2} = \mathrm{E3} = \mathrm{array}(M)$. ▷ Array of size M
for each JPEG (or raw) image of the series of H images, **do**
    Extract from the input image $\mathbf{u}$ of size $N_x \times N_y$ all possible M overlapping $w \times w$ blocks $\mathbf{B}_k$ and compute their 2D orthonormal DCT-II, $\tilde{\mathbf{B}}_k, k \in [0, M-1]$.
    for $k \in [0, M-1]$ **do**
        $\mathrm{E1}[k] = \mathrm{E1}[k] + (\tilde{\mathbf{B}}_k[i,j])^2$.
        $\mathrm{E2}[k] = \mathrm{E2}[k] + \tilde{\mathbf{B}}_k[i,j]$.
        $\mathrm{E3}[k] = \mathrm{E3}[k] + \tilde{\mathbf{B}}_k[0,0]/w$. ▷ The mean of $\mathbf{B}_k$
    end for
    $\mathrm{E1}[k] = \mathrm{E1}[k]/H$.
    $\mathrm{E2}[k] = \mathrm{E2}[k]/H$.
    $\mathrm{E3}[k] = \mathrm{E3}[k]/H$. ▷ Normalization
end for
Set $\mathbf{L} = \mathrm{array}(M)$ ▷ Array of size M
for $k \in [0, M-1]$ **do**
    Set $\mathbf{L}[k] = \left[\frac{k}{k-1}\left(\mathrm{E1}[k] - (\mathrm{E2}[k])^2\right)\right]^{1/2}$. ▷ STD
end for
Classify the elements of list $\mathbf{L}$ into disjoint bins **[1, 75]** according to the mean intensity $\mathrm{E3}[k]$ of the blocks. Each bin contains (with the exception of the last) 42000 sample variance estimations.
for each bin b, **do**
    Set $\mathbf{X}$ the means of the blocks in bin b.
    Set $\mathbf{Y}$ the STDs of the blocks in bin b.
    Get the 0.05-quantile of $\mathbf{Y}$ and set $\hat{\mu}$ the mean in $\mathbf{X}$ associated with it.
    Assign the 0.05-quantile of $\mathbf{Y}$ to $\hat{\sigma}[\hat{\mu}][i,j]$.
    Set $\hat{\sigma}[\hat{\mu}][i,j]$ the 0.05-quantile of $\mathbf{X}$. Set $\hat{\mu}$ the mean at the quantile position in $\mathbf{X}$.
    Correct $\hat{\sigma}$ biased by the quantile and obtain the final control point of the ground-truth for intensity $\hat{\mu}$ and frequency $[i,j]$:

$$\tilde{\sigma}[\hat{\mu}][i,j] = 1.22 \times \hat{\sigma}[\hat{\mu}][i,j].$$

end for

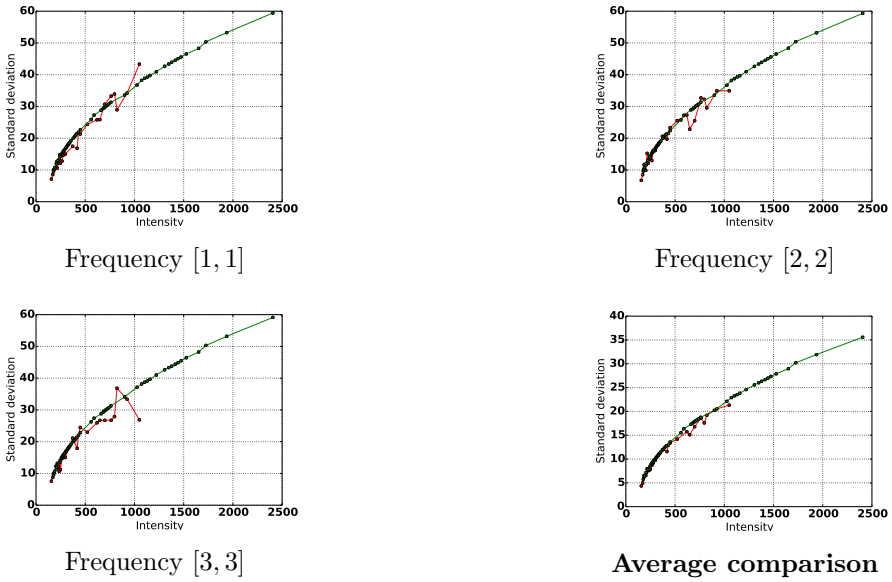


FIGURE 5. Comparison of the temporal (ground-truth, in green) and spatial STD (in red) for the Canon EOS 30D in raw images for ISO speed 1600 using blocks of 4×4 DCT coefficients. The temporal and spatial STD match despite some oscillation in the spatial estimation. The curve at the bottom right is the comparison between the averaged mean temporal STDs and the averaged mean spatial STDs (along all frequencies except DC), showing that on average both estimations match accurately.

In the sequel, we compare the results of the spatial estimation to the ground-truth, for both raw and JPEG-encoded images taken with a Canon EOS 30D camera with exposure time $t = 1/30s$, ISO speed 1600, and blocks of $w \times w$ DCT coefficients with $w = 4$ and $w = 8$. Figure 5 compares the temporal and the spatial STDs for raw images and Figure 7 shows the same for JPEG-encoded images with compression factor $Q = 92$ for $w = 4$. Only coefficients $[1, 1]$, $[2, 2]$, and $[3, 3]$ are shown, but equivalent results were obtained with all 15 coefficients. Respectively, Figure 6 and Figure 8 for $w = 8$. Only coefficients $[2, 2]$, $[5, 5]$, and $[7, 7]$ are shown for $w = 8$. The average of the estimations along all coefficients $[i, j] \in [0, w - 1]^2, [i, j] \neq [0, 0]$ is also given in both cases.

Despite small oscillations in the spatial estimation, there is an accurate match between both the spatial and temporal estimations in the case of raw and JPEG images. It can be concluded that the method is able to estimate reliably SFD noise.

This test was performed with snapshots of the calibration pattern, which is not textured and contains large flat areas whose spatial variations are caused mainly by the noise. Thus, the final validation must use real natural images compressed with JPEG. Since a proper noise model for JPEG encoding has not been already described, a visual comparison of the quality of the images before and after denoising using the frequency-by-frequency estimation given by the proposed method is needed. This comparison is performed in Section 5.1.

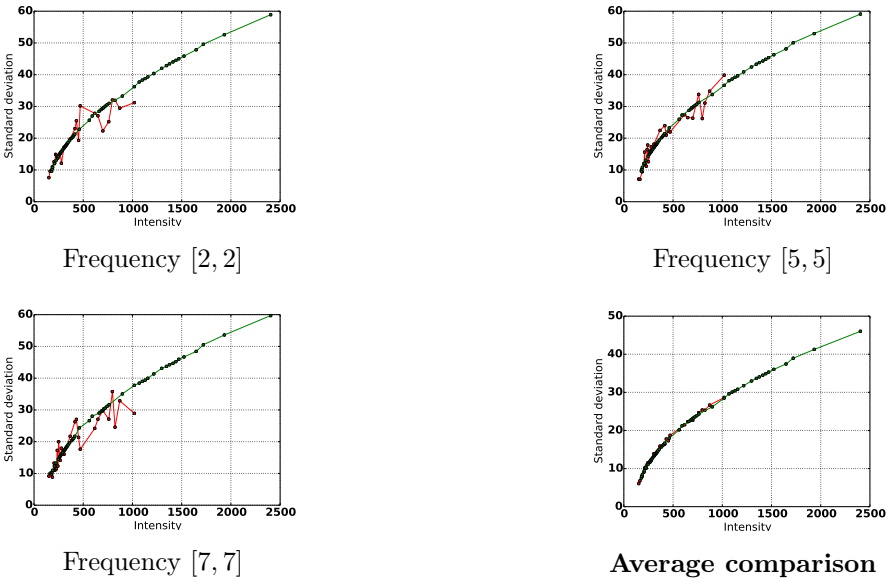
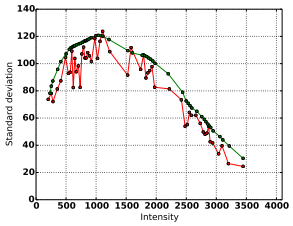


FIGURE 6. Comparison of the temporal (ground-truth, in green) and spatial STD (in red) for the Canon EOS 30D in raw images for ISO speed 1600 using blocks of 8×8 DCT coefficients. The temporal and spatial STD match despite some oscillation in the spatial estimation. The curve at the bottom right is the comparison between the averaged mean temporal STDs and the averaged mean spatial STDs (along all frequencies except DC), showing that on average both estimations match accurately.

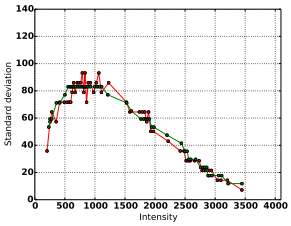
We also compared the accuracy of the proposed method by simulating colored noise and comparing the temporal STD (ground-truth) with the spatial estimation given by our algorithm for images of pure noise, frequency by frequency. To obtain the temporal STD, we created a list of 210 blocks of size 8×8 pixels made of simulated Gaussian noise of mean 127 and $\sigma = 10$ after applying a convolution with the discrete Gaussian kernel $\mathcal{G}$ in Equation 4. Figure 9 shows a crop of the convolved noise image, where it can be observed that it contains spatial structure, as the noise is correlated because of the Gaussian convolution.

$$\mathcal{G} = \frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}. \tag{4}$$

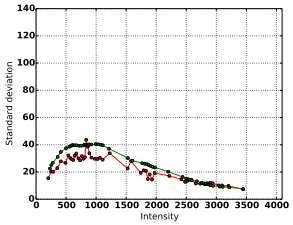
Table 1 compares for several frequencies (first column) the temporal STD obtained for the 210 8×8 blocks of pure noise (second column), the spatial STD estimation obtained by our method for pure noise after convolution with the discrete Gaussian kernel $\mathcal{G}$ in Equation 4 (third column), and the spatial STD estimation given by our algorithm after adding homoscedastic Gaussian noise



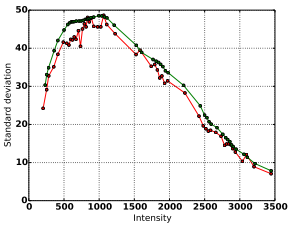
Frequency $[1, 1]$



Frequency $[2, 2]$



Frequency $[3, 3]$



Average comparison

FIGURE 7. Comparison of the temporal (ground-truth, in green) and spatial STD (in red) for the Canon EOS 30D in JPEG-encoded images with quality factor $Q = 92$ for ISO speed 1600 using blocks of 4×4 DCT coefficients. The curve at the bottom right is the comparison between the averaged temporal STDs and the averaged mean spatial STDs (along all frequencies except DC), showing that in average both estimations match.

of $\sigma = 50$ to the noise-free test image *computer* (in Figure 3, top right) and then convolving it with $\mathcal{G}$ (fourth column). With a small margin of error, the proposed method is able to measure the STD of the noise for both pure noise and textured natural images. If nevertheless the STD of the noise is below 0.4, the method is unable to estimate it accurately and in some cases the MAD estimator gives negative values that the algorithm sets to zero afterwards.

5.1. Denoising results. Old photographs are particularly well adapted to be evaluated with a SFD noise model. Indeed, as such images involve two different successive acquisition systems, one chemical and one digital, the noise model is fully unknown and could not be learnt but from the image itself. And there is, of course, no ground-truth. Yet the visual inspection of the noise gives a very good hint at its independence from the (recovered) signal. To denoise JPEG digital images of old photographs, we used a modified version of the NL-Bayes algorithm [**84**] using the noise DCT coefficients estimated by our algorithm in Section 3. The details of the denoiser can be found in [**22**]. Of course, other patch-based denoisers [**37, 36, 86, 87**] might be used instead. For the denoising tests shown in this chapter, we estimated the noise at two scales to go deeper in low frequencies: a noise patch model was estimated at the finer scale and a second noise model was also computed after a Gaussian image zoom in.

Figure 10 shows denoising results for images with unknown noise model. In the first row, details of the noisy images; in the second row, details of the denoised images; in the third row,

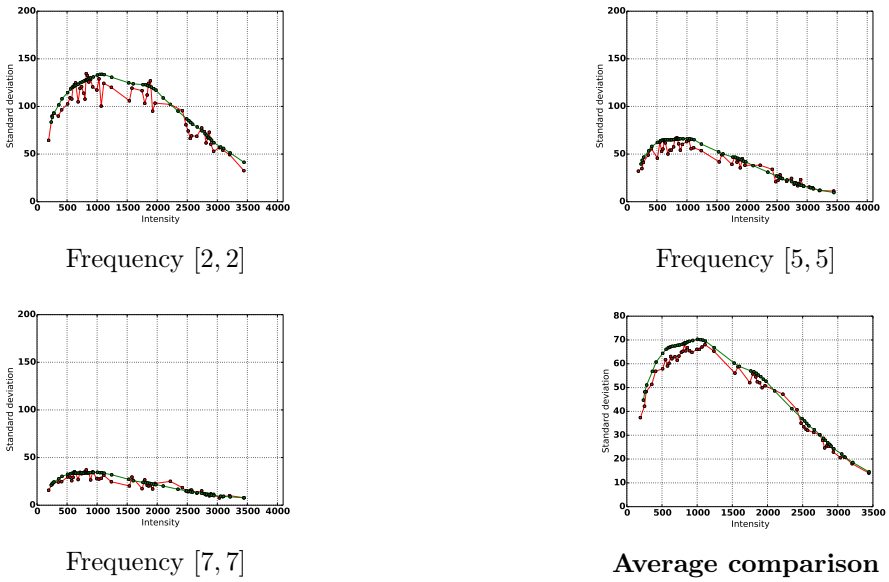


FIGURE 8. Comparison of the temporal (ground-truth, in green) and spatial STD (in red) for the Canon EOS 30D in JPEG-encoded images with quality factor $Q = 92$ for ISO speed 1600 using blocks of 8×8 DCT coefficients. The curve at the bottom right is the comparison between the averaged temporal STDs and the averaged mean spatial STDs (along all frequencies except DC), showing that on average both estimations match.

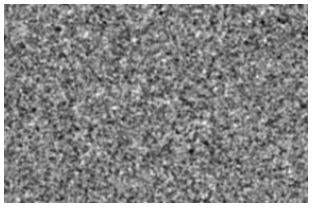


FIGURE 9. Crop of the image of pure Gaussian noise with mean 127 and $\sigma = 50$ after convolution with the kernel $\mathcal{G}$ in Equation 4. The noise has spatial structure, as it gets correlated after Gaussian convolution.

enhanced difference image (removed noise) between the noisy and denoised image. The color spots in the difference image and their random aspect at zones with the same intensity indicate that the denoising algorithm removed colored noise and, since details are kept at the denoised image, it can be concluded that the noise estimation was successful.

Figs. 11 and 12 show the noise curves corresponding to the low and high frequencies of the *Apollo* and *Kleiner* images for which a detail is shown at the bottom of Figure 10 using DCT blocks of 4×4 coefficients. A coefficient at frequency $[i, j] \in [0, 3]^2$ is assumed to belong to a "low-frequency" if $i + j \leq 2$ and to a "high-frequency" otherwise. Image *Apollo* was taken in 18 May 1969 during the prelaunch tasks at the Launch Control Center's Firing Room 3 at the Kennedy

TABLE 1. This table compares for several frequencies (first column), the temporal STD obtained for the 210 8×8 blocks of Gaussian noise of $\sigma = 50$ (second column), the spatial STD estimation obtained by our method for pure noise after convolution with the Gaussian kernel $\mathcal{G}$ in Equation 4 (third column), and the spatial STD estimation given by our algorithm after adding homoscedastic Gaussian noise of $\sigma = 50$ to the noise-free test image *computer* and then convolving it with $\mathcal{G}$ (fourth column). Both STD estimation in pure noise and in a textured natural image match with small error the temporal STD. For the image of pure noise a single bin is used and 7 bins for the *computer* image.

Frequency	**Temporal STD**	**Spatial (pure noise)**	**Spatial (*computer*)**
$[1, 1]$	31.45	29.20	30.72
$[2, 2]$	21.77	19.93	21.06
$[3, 3]$	10.03	11.16	10.73
$[4, 4]$	3.44	3.72	3.76
$[5, 5]$	0.73	0.62	0.61
$[6, 6]$	0.15	0	0
$[7, 7]$	0.13	0	0

Space Center[2] and image *Kleiner* is a picture of an old tramway called "Kleiner Hecht" taken in 1998 in Dresden[3]. Both images contain large low-frequency noise and JPEG compression artifacts. We show the mean of the noise curves from the low-frequencies before (a) and after (b) denoising, where it can be observed that most of the noise remains at the low-frequencies of the image and that is strongly reduced after denoising. We also show the means for the high-frequencies before (c) and after (d) denoising. Since JPEG quantizes the value of the DCT coefficients at the high-frequencies (thus canceling most of them), the noise is clearly lower that what is observed at the low-frequencies, but nevertheless the noise could also be removed.

This chapter presented a non-parametric noise estimation method for SFD noise. It can be applied to images where the noise model is not available [**8**], as in the case of JPEG images. In general, a non-parametric estimation of SFD noise is needed for almost any kind of image, since the very first step in the camera processing chain is to demosaic the raw image, which correlates the noise. In general, the information of what algorithm used to demosaic the image is not made available by camera makers. However, the goal of blind noise estimation and denoising is to retrieve the noise model from the image itself, without relying at additional information such as metadata

[2]This file is in the public domain because it was solely created by NASA. NASA copyright policy states that "NASA material is not protected by copyright unless noted". `http://dayton.hq.nasa.gov/IMAGES/LARGE/GPN-2000-001849.jpg`

[3]Image licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license, taken by Wikimedia Commons user Olaf1541. `http://commons.wikimedia.org/wiki/File:Kleiner_hecht.jpg`

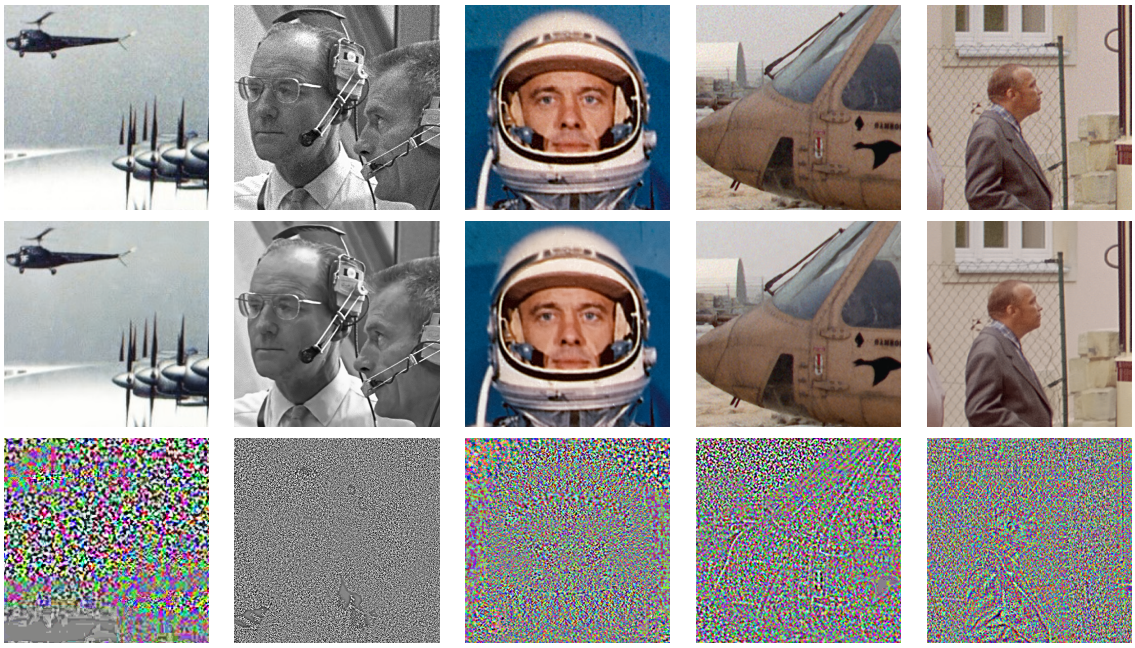


FIGURE 10. Denoising results of real images with unknown noise model. Up: detail of the noisy image. Middle: detail of the denoised image. Bottom: difference image (removed noise). The color spots in the difference image and its random geometry at zones with the same intensity indicate that the denoising algorithm removed colored noise and, since details are kept at the denoised image, it can be concluded that the noise estimation was accurate. In order to see clearly the low-frequency noise and the denoising results, the reader is invited to look at the images on the screen with a 400% zoom at least.

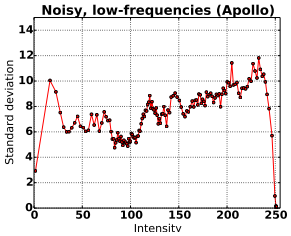
in the file format. Of course, for old photographs the metadata information might not exist at all (analogic photography) or may have been lost after image manipulation and re-encoding.

To determine if two patches contain the same geometric information, a new similarity function based on the sparse distance between the patches introduced. It exploits the sparsity between patches to give a likelihood measure which is robust to noise. The use of the MAD estimator to compute the STD confers more robustness to the method.

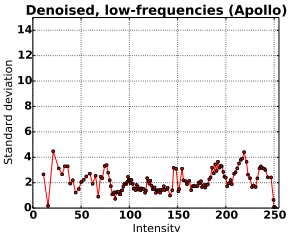
The method was validated by showing that the STD obtained at the temporal series (the ground-truth) coincides with the spatial STD given by the proposed algorithm, for both raw and JPEG images. The denoising results show that indeed the noise estimator is able to give an accurate estimation, as low frequency noise is removed and most of the fine details are kept. Our next endeavor would be to include an impulse noise estimator to the non-parametric noise estimation model. Old photographs can indeed present this sort of noise.

Nevertheless this estimation algorithm cannot be applied to *any* noisy image. For example, it does not apply if the noise is space dependent (and not only signal dependent), as can be observed in some synthetic images. Another limitation for estimating highly-correlated noise is the size of

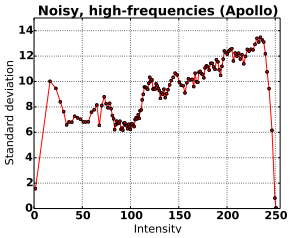
Original noisy image *Apollo* (a)



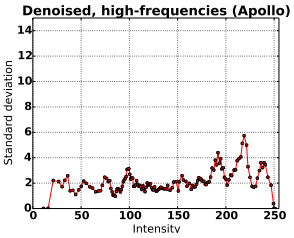
Low-freqs., noisy (b)



Low-freqs., denoised (c)



High-freqs., noisy (d)



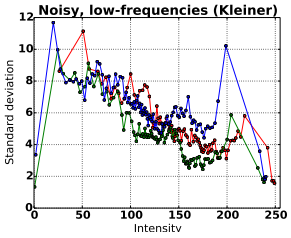
High-freqs, denoised (e)

FIGURE 11. Noise curves corresponding to the low and high frequencies of the *Apollo* image (a) for which a detail is shown at the second column of Figure 10 using DCT blocks of 4×4 coefficients. (b) and (c): mean noise curve at the low-frequencies before (b) and after (c) denoising. (d) and (e): mean noise curves at the high-frequencies before (d) and after (e) denoising. Most of the noise is at the low-frequencies.

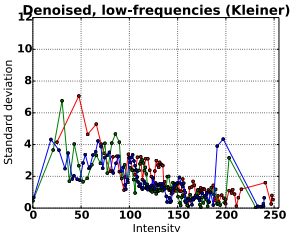
the noisy image, since two scales of the image are needed and the number of available samples (pixels) and the second scale is one quarter of the pixels in the noisy image. If the image is small and contains highly correlated noise, it may not be possible to characterize it properly.

Chapter 8 describes a new denoising algorithm able to denoised even JPEG-encoded images, using the noise estimation algorithm presented in this chapter.

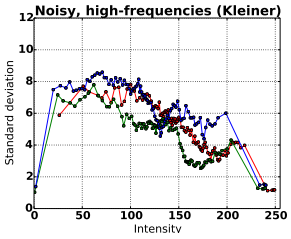
Original noisy image *Kleiner* (a)



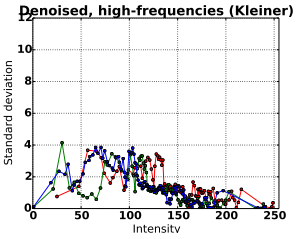
Low-freqs., noisy (b)



Low-freqs., denoised (c)



High-freqs., noisy (d)



High-freqs, denoised (e)

FIGURE 12. Noise curves corresponding to the low and high frequencies of the *Kleiner* image (a) for which a detail is shown at the last column of Figure 10 using DCT blocks of 4×4 coefficients. (b) and (c): mean noise curve at the low-frequencies before (b) and after (c) denoising. (d) and (e): mean noise curves at the high-frequencies before (d) and after (e) denoising. Most of the noise is at the low-frequencies. The color of each of the curves corresponds to each color channel of the image (red, green, and blue).

Part 2

PATCH DENOISING

Once the noise estimation models and methods have been studied in the first part of the dissertation, in this second part we focus on the problem of denoising. In Chapter 5 we discuss the Bayesian patch-based methods. This formalism permits to obtain a closed formula that given a noisy patch in the image gives a denoised version from it.

Regardless of the block-based denoising method used, it is possible to improve dramatically the results by using the three generic tools explained in Chapter 6: aggregation of estimates, iteration and "oracle" filters, and colorspace transform. This remark is valid for all block-based denoising principles and will be used in our final Noise Clinic (Chapter 8). The effect of this tools is demonstrated with an example of a simple DCT transform threshold, where the three generic tools are applied successively.

In Chapter 7 we give a detailed description and analysis of two denoising methods for which reliable faithful implementations are available: the Non-local means and Non-local Bayes methods. The algorithmic descriptions of both methods is provided. We shall extend this non-local Bayes algorithm in our final Chapter 8 on the Noise Clinic.

Chapter 8 presents a new algorithm for both blind noise estimation and denoising for images where the noise model is imperfectly known or even unknown. For such images, it is possible to estimate from a single image a noise model which is simultaneously signal and frequency dependent, as we showed in Chapter 4. We propose a multiscale denoising algorithm adapted to this broad noise model, based on the NL-Bayes denoising algorithm (Chapter 7). This brings to a blind denoising algorithm which we demonstrate on real JPEG images and on scans of old photographs for which the formation model is unknown. This algorithm is finally compared to the unique state of the art previous blind denoising method, based on Gaussian Scale Mixtures In The Wavelet Domain.

CHAPTER 5

Bayesian patch-based methods

Bayesian patch-based methods give an optimal formulation under the assumption that the patches similar to a given image patch follow a stochastic model. Given a noiseless patch P of the ideal noiseless image u, and Gaussian model and assuming a Gaussian noise model with independence of pixel values, the probability $\mathbb{P}(\tilde{P}|P)$ (the probability of observing a noisy patch $\tilde{P}$ with the prior information that the noiseless version of the noisy patch is P) can be obtained directly. Thanks to Bayes' formula, it is possible to obtain $\mathbb{P}(P|\tilde{P})$, the probability of the noiseless patch P knowing that the noisy patch $\tilde{P}$ has been observed. The goal is to deduce P from $\tilde{P}$ by finding P which maximizes $\mathbb{P}(P|\tilde{P})$. This chapter concludes with a closed formula that gives a restored patch $\hat{P}_1$ from an observed noisy patch $\tilde{P}$, using the covariance matrix $\boldsymbol{C}_{\tilde{P}}$ of $\tilde{P}$.

1. Obtaining a restored patch $\hat{P}_1$ from an observed noisy patch $\tilde{P}$

In this chapter, we will review the Bayesian patch-based method applied to denoising, under the additive white Gaussian noise model. The Bayesian principle is coupled with a Gaussian (or a mixture of Gaussians) model for noiseless patches. Given u the noiseless ideal image and $\tilde{u}$ the noisy image corrupted with Gaussian noise of standard deviation σ so that

$$\tilde{u} = u + n, \tag{5}$$

the conditional distribution $\mathbb{P}(\tilde{u} \mid u)$ is

$$\mathbb{P}(\tilde{u} \mid u) = \frac{1}{(2\pi\sigma^2)^{\frac{M}{2}}} \exp\left(\frac{-||u-\tilde{u}||^2}{2\sigma^2}\right), \tag{6}$$

where M is the total number of pixels in the image.

In order to compute the probability of the original image given the degraded one, $\mathbb{P}(u \mid \tilde{u})$, we need to introduce a prior on u. In the first models **[30]**, this prior was a parametric image model describing the stochastic behavior of a patch around each pixel by a Markov random field, specified by its Gibbs distribution. A Gibbs distribution for an image u takes the form

$$\mathbb{P}(u) = \frac{1}{Z} \exp\left(-E(u)/T\right),$$

where Z and T are constants and E is called the energy function and writes

$$E(u) = \sum_{C \in \mathcal{C}} V_C(u),$$

where $\mathcal{C}$ denotes the set of cliques associated to the image and V_C is a potential function. The maximization of the *a posteriori* distribution writes by Bayes formula

$$\operatorname*{Arg\,max}_u \mathbb{P}(u \mid \tilde{u}) = \operatorname*{Arg\,max}_u \mathbb{P}(\tilde{u} \mid u)\mathbb{P}(u),$$

which is equivalent to the minimization of $-\log \mathbb{P}(u \mid \tilde{u})$,

$$\operatorname*{Arg\,min}_u \|u - \tilde{u}\|^2 + \frac{2\sigma^2}{T} E(u).$$

This energy writes as a sum of local derivatives of pixels in the image, thus being equivalent to a classic Tikhonoff regularization **[30, 88]**.

Recent Bayesian methods have abandoned as too simplistic the global patch models formulated by an *a priori* Gibbs energy. Instead, the methods build local non parametric patch models learnt from the image itself, usually as a local Gaussian model around each given patch, or as a Gaussian mixture. The term "patch model" is now preferred to the terms "neighborhood" or "clique" previously used for the Markov field methods. In the nonparametric models, the patches are larger, usually 8×8, while the cliques were often confined to 3×3 neighborhoods. Given a noiseless patch P of u with dimension $\kappa \times \kappa$, and $\tilde{P}$ an observed noisy version of P, the same model gives by the independence of noise pixel values

$$\mathbb{P}(\tilde{P}|P) = c \cdot \exp\left(-\frac{\|\tilde{P} - P\|^2}{2\sigma^2}\right) \tag{7}$$

where P and $\tilde{P}$ are considered as vectors with κ^2 components, $||P||$ denotes the Euclidean norm of P, σ^2 the variance of the Gaussian noise, and c is the normalizing constant. Knowing $\tilde{P}$, our goal is to deduce P by maximizing $\mathbb{P}(P|\tilde{P})$. Using Bayes' rule, we can compute this last conditional probability as

$$\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}. \tag{8}$$

$\tilde{P}$ being observed, this formula can in principle be used to deduce the patch P maximizing the right term, viewed as a function of P. This is only possible if we have a probability model for P, and these models will be generally learnt from the image itself, or from a set of images. For example **[89]** applies a clustering method to the set of patches of a given image, and **[90]** applies it to a huge set of patches extracted from many images. Each cluster of patches is thereafter treated as a set of Gaussian samples. This permits to associate to each observed patch its likeliest cluster, and then to denoise it by a Bayesian estimation in this cluster. Another still more direct way to build a model for a given patch $\tilde{P}$ is to group the patches similar to $\tilde{P}$ in the image. Assuming that these similar patches are samples of a Gaussian vector yields a standard Bayesian restoration **[47, 91]**. We shall now discuss this particular case, where all observed patches are noisy.

Why Gaussian? As usual when we dispose of several observations but of no particular guess on the form of the probability density, a Gaussian model is adopted. In the case of the patches Q similar to a given patch P, the Gaussian model has some pertinence, as it is assumed that many contingent random factors explain the difference between Q and P: other details, texture, slight

lighting changes, shadows, etc. The Gaussian model in presence of a combination of many such random and independent factors is heuristically justified by the central limit theorem. Thus, for good or bad, assume that the patches Q similar to P follow a Gaussian model with (observable, empirical) covariance matrix $\mathbf{C}_P$ and (observable, empirical) mean $\overline{P}$. This means that

$$\mathbb{P}(Q) = c \cdot \exp\left(\frac{-(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}\right) \tag{9}$$

From (6) and (8) we obtain for each observed $\tilde{P}$ the following equivalence of problems:

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

This expression does not yield an algorithm. Indeed, the noiseless patch P and the patches similar to P are not observable. Nevertheless, we can observe the noisy version $\tilde{P}$ and compute the patches $\tilde{Q}$ similar to $\tilde{P}$. An empirical covariance matrix can therefore be obtained for the patches $\tilde{Q}$ similar to $\tilde{P}$. Furthermore, using (5) and the fact that P and the noise n are independent,

$$\mathbf{C}_{\tilde{P}} = \mathbf{C}_P + \sigma^2\mathbf{I}; \qquad E\tilde{Q} = \overline{P}. \tag{10}$$

Notice that these relations assume that we searched for patches similar to $\tilde{P}$ at a large enough distance, to include all patches similar to P, but not too large either, because otherwise it can contain outliers. Thus the safe strategy is to search similar patches in a distance slightly larger than the expected distance caused by noise. If the above estimates are correct, our MAP (maximum *a posteriori* estimation) problem finally boils down by (10) to the following feasible minimization problem:

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

Differentiating this quadratic function with respect to P and equating to zero yields

$$P - \tilde{P} + \sigma^2(\mathbf{C}_{\tilde{P}}-\sigma^2\mathbf{I})^{-1}(P-\overline{\tilde{P}}) = 0.$$

Taking into account that $\mathbf{I} + \sigma^2(\mathbf{C}_{\tilde{P}}-\sigma^2\mathbf{I})^{-1} = (\mathbf{C}_{\tilde{P}}-\sigma^2\mathbf{I})^{-1}\mathbf{C}_{\tilde{P}}$, this yields

$$(\mathbf{C}_{\tilde{P}}-\sigma^2\mathbf{I})^{-1}\mathbf{C}_{\tilde{P}}P = \tilde{P} + \sigma^2(C_{\tilde{P}}-\sigma^2\mathbf{I})^{-1}\overline{\tilde{P}}.$$

and therefore

$$\begin{aligned}
P &= \mathbf{C}_{\tilde{P}}^{-1}(\mathbf{C}_{\tilde{P}}-\sigma^2\mathbf{I})\tilde{P} + \sigma^2\mathbf{C}_{\tilde{P}}^{-1}\overline{\tilde{P}} \\
&= \tilde{P} + \sigma^2\mathbf{C}_{\tilde{P}}^{-1}(\overline{\tilde{P}}-\tilde{P}) \\
&= \overline{\tilde{P}} + \left[\mathbf{I} - \sigma^2\mathbf{C}_{\tilde{P}}^{-1}\right](\tilde{P}-\overline{\tilde{P}}) \\
&= \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I}\right]\mathbf{C}_{\tilde{P}}^{-1}(\tilde{P}-\overline{\tilde{P}})
\end{aligned}$$

Thus we have proved that a restored patch $\hat{P}_1$ can be obtained from the observed patch $\tilde{P}$ by the one step estimation

$$\hat{P}_1 = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I}\right]\mathbf{C}_{\tilde{P}}^{-1}(\tilde{P} - \overline{\tilde{P}}), \tag{11}$$

which resembles a local Wiener filter.

REMARK 1. *It is easily deduced that the expected estimation error is*

$$E||P - \hat{P}_1||^2 = Tr\left[\left(\boldsymbol{C}_P^{-1} + \frac{\boldsymbol{I}}{\sigma^2}\right)^{-1}\right].$$

Chapter 7 will examine two deriving patch-based denoising algorithms from variants of (11). The first question when looking at this formula is obviously how the matrix $\mathbf{C}_{\tilde{P}}$ can be learnt from the image itself. Each method proposes a different notion to learn the patch model.

Of course, other, non Gaussian, Bayesian models are possible, depending on the patch density assumption. For example **[11]** assumes a local exponential density model for the noisy data, and gives a convergence proof to the optimal (Bayes) least squares estimator as the amount of data increases.

CHAPTER 6

Generic tools for noise reduction

This chapter describes three generic tools (aggregation of estimates, iteration and "oracle" filters, and colorspace transform) that permit to increase the performance of any denoising principle. We shall illustrate them on DCT denoising. Starting from the application of a simple DCT transform threshold, the three generic tools will applied successively. We shall observe a dramatic improvement of the denoising performance. This remark is valid for all denoising principles and will be used in our final Noise Clinic (Chapter 8).

1. Aggregation of estimates

Aggregation techniques combine for any pixel a set of m possible estimates. If these estimates were independent and had equal variance, then a uniform average would reduce this estimator variance by a factor m. Such an aggregation strategy was the main proposition of the *translation invariant wavelet thresholding algorithm* [**92**]. This method denoises several translations of the image by a wavelet thresholding algorithm and averages these different estimates once the inverse translation has been applied to the denoised images.

An interesting case is when one is able to estimate the variance of the m estimators. Statistical arguments lead to attribute to each estimator a weight inversely proportional to its variance [**93**]. For most denoising methods the variance of the estimators is high near image edges. When applied without aggregation, the denoising methods leave visible "halos" of residual noise near edges. For example in the sliding window DCT method, patches containing edges have many large DCT coefficients which are kept by thresholding. In flat zones instead, most DCT coefficients are canceled and the noise is completely removed. The proposition of [**94**] is to use the aggregation for DCT denoising, approximating the variance of each estimated patch by the number of non zero coefficients after thresholding. In the online paper [**95**] one can test an implementation of DCT denoising. It actually uses an aggregation with uniform weights: "translation invariant DCT denoising is implemented by decomposing the image to sliding overlapping patches, calculating the DCT denoising in each patch, and then aggregating the denoised patches to the image averaging the overlapped pixels. The translation invariant DCT denoising significantly improves the denoising performance, typically from about 2 to 5 dB, and removes the block artifact".

The same risk of "halo" occurs with non-aggregated NL-means (Chapter 7), since patches containing edges have many less similar instances in the image than flat patches. Thus the non-local averaging is made over less samples, and the final result keeps more noise near image edges. The same phenomenon occurs with BM3D if the aggregation step is not applied [**36**]. As a consequence,

an aggregation step is applied in all patch-based denoising algorithms. This weighted aggregation favors, at each pixel near an edge, the estimates given by patches which contain the pixel but do not meet the edge.

Aggregation techniques aim at a superior noise reduction by increasing the number of values being averaged for obtaining the final estimate or selecting those estimates with lower variance. Kervrann et al [**96**] considered the whole Bias+Variance decomposition in order to also adapt the search zone of neighborhood filters or of NL-means. Since the bias term depends on the original image, it cannot be computed in practice, and Kervrann et al. proposed to minimize both bias and variance by choosing the smallest spatial neighborhood attaining a stable noise reduction.

Another type of aggregation technique considers the risk estimate rather than the variance to locally attribute more weight to the estimators with small risks. In [**12**], Van De Ville and Kocher give a closed form expression of Stein's Unbiased Estimator of the Risk (SURE) for NL-Means. (See also generalizations of the SURE estimator to the non-Gaussian case in [**97**].) The aim is to select globally the best bandwidth for a given image. In [**98**], Duval et al. also use the SURE technique for minimizing the risk by selecting locally the bandwidth. Deledalle et al. [**13**] apply the same technique for combining the results of NL-means with different window sizes and shapes. A similar treatment can be found in [**11**], but with the assumption of a local exponential density for the noisy patches.

2. Iteration and "oracle" filters

Iterative strategies to remove residual noise would drift from the initial image. Instead, a first step denoised image can be used to improve the reapplication of the denoising method to the initial noisy image. In a second step application of a denoising principle, the denoised DCT coefficients, or the patch distances, can be computed in the first step denoised image. They are an approximation to the true measurements that would be obtained from the noise-free image. Thus, the first step denoised image is used as an "oracle" for the second step.

For averaging filters such as neighborhood filters or NL-means, the image u can be denoised in a first step by the method under consideration. This first step denoised image denoted by $\hat{u}_1$ is used for computing more accurate color distances between pixels. Thus, the second step neighborhood filter is

$$YNF_{h,\rho}\tilde{u}(\mathbf{i}) = \frac{1}{C(\mathbf{i})} \sum_{\mathbf{j}\in B_\rho(\mathbf{j})} \tilde{u}(\mathbf{j}) e^{-\frac{|\hat{u}_1(\mathbf{j})-\hat{u}_1(\mathbf{i})|^2}{h^2}},$$

where $\tilde{u}$ is the observed noisy image and $\hat{u}_1$ the image previously denoised by Equation (12).

$$YNF_{h,\rho}\tilde{u}(\mathbf{i}) = \frac{1}{C(\mathbf{i})} \sum_{\mathbf{j}\in B_\rho(\mathbf{i})} \tilde{u}(\mathbf{j})\, e^{-\frac{|\tilde{u}(\mathbf{i})-\tilde{u}(\mathbf{j})|^2}{h^2}}. \tag{12}$$

Similarly, for linear transform Wiener-type methods, the image is first denoised by its classic definition, which amounts to approximate the oracle coefficients of Theorem 1 using the noisy ones.

In a second iteration, the coefficients of the denoised image approximate the true coefficients of the noise-free image. Thus, the second step filter following the first step (Equation 13) is

$$\mathbf{D}\tilde{U} = \sum_i a(i)\ \langle \tilde{U}, G_i \rangle\ G_i\,, \quad \text{with} \quad a(i) = \frac{|\langle \hat{U}_1, G_i \rangle|^2}{|\langle \hat{U}_1, G_i \rangle|^2 + \sigma^2},$$

where $\hat{U}_1$ is the denoised image by applying a first time the thresholding algorithm to the observed image $\tilde{U}$.

$$\mathbf{D}\tilde{U} = \sum_{i=1}^{M} a(i)\, \langle \tilde{U}, G_i \rangle\, G_i. \tag{13}$$

THEOREM 1. *Operator $\boldsymbol{D}_{inf}$ minimizing the mean square error (MSE),*

$$\boldsymbol{D}_{inf} = arg\min_{\boldsymbol{D}} E\{\|U - \boldsymbol{D}\tilde{U}\|^2\}$$

is given by the family $\{a(i)\}_i$, where

$$a(i) = \frac{|\langle U, G_i \rangle|^2}{|\langle U, G_i \rangle|^2 + \sigma^2}, \tag{14}$$

and the corresponding expected mean square error (MSE) is

$$E\{\|U - \boldsymbol{D}_{inf}\tilde{U}\|^2\} = \sum_{i=1}^{M} \frac{|\langle U, G_i \rangle|^2 \sigma^2}{|\langle U, G_i \rangle|^2 + \sigma^2}. \tag{15}$$

2.0.0.1. Alternatives and extensions: "twicing" and Bregman iterations. In the recent review paper [**99**], many denoising operators are formalized in a general linear framework, noting that they can be associated with a doubly stochastic diffusion matrix W with nonnegative coefficients. For example in NL-means, this matrix is obtained by the symmetrization of the matrix of the NL-means weights $w_{\tilde{P},\tilde{Q}}$ defined in Algorithm 1. Unless it is optimal, as is the case with an ideal Wiener filter, the matrix W associated with the denoising filter can be iterated. A study of MSE evolution with these iterations is proposed in [**99**] for several denoising operators, considering several different patch types (texture, edge, flat). Iteration is, however, different from the oracle iteration described above. In the oracle iteration, the matrix W is changed at each step, using its better estimate given by the previously denoised image. One does not generally observe much improvement by iterating the oracle method more than once. [**99**] points out another generic tool, used at least for total variation denoising, the so-called "twicing", term due to Tukey [**100**]. Instead of repeated applications of a filter, the idea is to process the residual obtained as the difference between the estimated image and the initial image. If the residuals contain some of the underlying signal, filtering them should recover part of it. The author shows that the Bregman iterations [**101**] used for improving total variation based denoising are a twicing and so is the matching pursuit method used in the K-SVD filter.

3. Dealing with color images

The straightforward strategy to extend denoising algorithms to color or multivalued images is to apply the algorithm independently to each channel. The use of this simple strategy often introduces color artifacts, easily detected by the eye. Two different strategies are observable in state of the art denoising algorithms.

Depending on the algorithm formulation, a vector-valued version dealing at the same time with all color channels can be proposed. This solution is adopted by averaging filters like neighborhood filters or NL-means. These algorithms compute color differences directly in the vector valued image, thus yielding a unified weight configuration which is applied to each channel.

The alternative option is to convert the usual RGB image to a different color space where the independent denoising of each channel does not create noticeable color artifacts. Most algorithms use the YUV system which separates the geometric and chromatic parts of the image. This change writes as a linear transform by multiplication of the RGB vector by the matrix

$$YUV = \begin{pmatrix} 0.30 & 0.59 & 0.11 \\ -0.15 & -0.29 & 0.44 \\ 0.61 & -0.51 & -0.10 \end{pmatrix}, \qquad Y_oU_oV_o = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{2} & 0 & -\frac{1}{2} \\ \frac{1}{4} & -\frac{1}{2} & \frac{1}{4} \end{pmatrix}$$

The second color transform to the space $Y_oU_oV_o$ is an orthogonal transform. It has the advantage of maximizing the noise reduction of the geometric component, since this component is an average of the three color. The geometric component is perceptually more important than the chromatic ones, and the presence of less noise permits a better performance of the algorithm in this part. It also permits a higher noise reduction on the chromatic components U_o and V_o, due to their observable regularity.

This latter strategy is adopted by transform thresholding filters for which the design of an orthonormal basis coupling the different color channels is not trivial.

4. Trying all generic tools on an example

This chapter applies incrementally the previous generic denoising tools to the DCT sliding window to illustrate how these additional tools permit to drastically improve the algorithm performance. We start with the basic DCT "neighborhood filter" as proposed by Yaroslavsky [**28**]. Its principle is to denoise a patch around each pixel, and to keep only the central denoised pixel.

Figure 1 displays the denoised images obtained by incrementally applying each of the following ingredients:

- Basic DCT thresholding algorithm by the neighborhood filter technique (keeping only the central pixel of the window). Each color channel is treated independently.
- Use of an orthogonal geometric and chromatic decomposition color system $Y_oU_oV_o$; grey parts are better reconstructed and color artifacts are reduced.
- Uniform aggregation; the noise reduction is superior and isolated noise points are removed.

- Adaptive aggregation using the estimator variance; the noise reduction near edges is increased, "halo" effects are removed.
- Additional iteration using “oracle” estimation; residual noise is totally removed and the sharpness of details is increased.

The $PSNR$'s obtained by incrementally applying the previous strategies respectively are 26.85, 27.33, 30.65, 30.73, 31.25. This experiment illustrates how the use of these additional tools is crucial to achieve competitive results. This last version of the DCT denoising algorithm, incorporating all the proposed generic tools, will be the one used in the comparison chapter. A complete description of the algorithm can be found in Algorithm 17. The color version of the algorithm applies the denoising independently to each $Y_oU_oV_o$ component. This version is therefore slightly better than the version online in [**95**], which does not use the oracle step.

Algorithm 17 DCT denoising algorithm. DCT coefficients lower than 3σ are canceled in the first step and a Wiener filter is applied in the "oracle" second step. The color DCT denoising algorithm applies the current strategy independently to each $Y_oU_oV_o$ component.

Input: noisy image $\tilde{u}$, σ noise standard deviation.
Optional: prefiltered image $\hat{u}_1$ for "oracle" estimation.
Output: output denoised image.

Set parameter $\kappa = 8$: size of patches.
Set parameter $h = 3\,\sigma$: threshold parameter.

for each pixel **i** **do**
    Select a square reference patch $\tilde{P}$ around **i** of size $\kappa \times \kappa$.
    if $\hat{u}_1$ **then**
        Select a square reference patch P_1 around **i** in $\hat{u}_1$.
    end if
    Compute the DCT transform of $\tilde{P}$.
    if $\hat{u}_1$ **then**
        Compute the DCT transform of P_1.
    end if
    if $\hat{u}_1$ **then**
        Modify DCT coefficients of $\tilde{P}$ as

$$\tilde{P}(i) = \tilde{P}(i)\frac{P_1(i)^2}{P_1(i)^2 + \sigma^2}$$

    else
        Cancel coefficients of $\tilde{P}$ with magnitude lower than h.
    end if
    Compute the inverse DCT transform obtaining $\hat{P}$.
    Compute the aggregation weight $w_{\tilde{P}} = 1/\#\{\text{number of non-zero } DCT \text{ coefficients}\}$.
end for

for each pixel **i** **do**
    Aggregation: recover the denoised value at each pixel **i** by averaging all values at **i** of all denoised patches $\hat{Q}$ containing **i**, weighted by $w_{\tilde{Q}}$.
end for

FIGURE 1. Top: original and noisy images with an additive Gaussian white noise of standard deviation 25. Below, from top to bottom and left to right: crop of denoised images by sliding DCT thresholding filter and incrementally adding use of a $Y_oU_oV_o$ color system, uniform aggregation, variance based aggregation and iteration with the "oracle" given by the first step. The corresponding PSNR are 26.85, 27.33, 30.65, 30.73, 31.25.

CHAPTER 7

Detailed analysis of the Non-Local Means and the Non-local Bayes methods

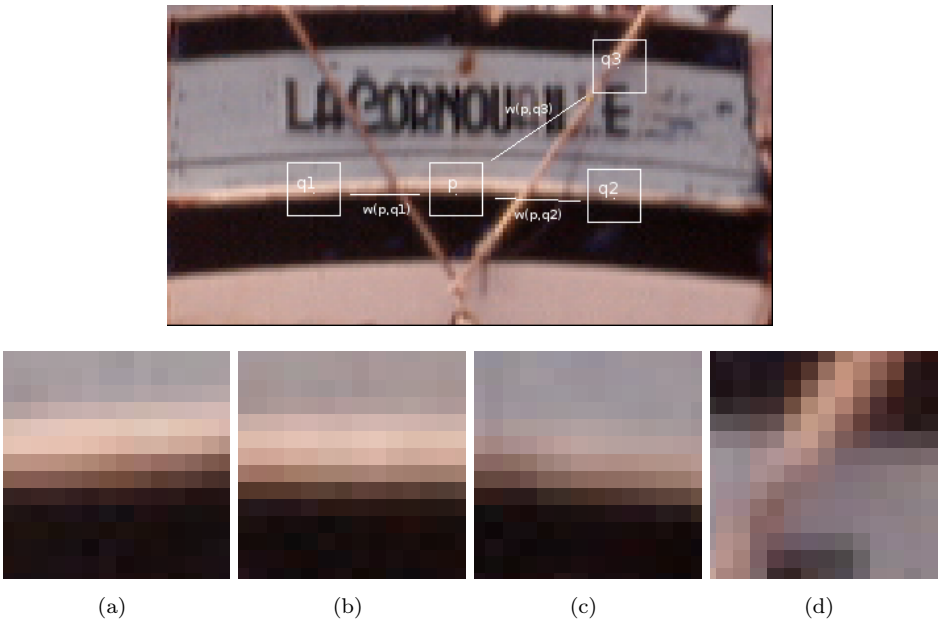
In this chapter, we give a detailed description and analysis of two denoising methods for which reliable faithful implementations are available: the Non-local means and Non-local Bayes methods. The algorithmic descriptions of both methods is provided. We shall extend this non-local Bayes algorithm in our final Chapter 8 on the Noise Clinic.

1. Non-local means

The Non-local means (NL-means) algorithm tries to take advantage of the redundancy of most natural images. The redundancy, or self-similarity hypothesis is that for every small patch in a natural image one can find several similar patches in the same image, as illustrated in figures 1 and 2. This similarity is true for patches whose centers stand at a one pixel distance of the center of the reference patch. In that case the self-similarity boils down to a local image regularity assumption. Such a regularity is guaranteed by Shannon-Nyquist's sampling conditions, which require the image to be blurry. In a much more general sense inspired by neighborhood filters, one can define as "neighborhood of a pixel **i**" any set of pixels **j** in the image such that a patch around **j** looks like a patch around **i**. All pixels in that neighborhood can be used for predicting the value at **i**, as was first shown in [**102**] for the synthesis of texture images. This self-similarity hypothesis is a generalized periodicity assumption. The use of self-similarities is actually well-known in information theory from its foundation. In his 1948 Mathematical Theory of Communication, Shannon [**25**] analyzed the local self-similarity (or redundancy) of natural written language, and gave probably the first stochastic text synthesis algorithm. The Efros-Leung texture synthesis method adapted this algorithm to images, and NL-Means [**103**] seems to be first adaptation of the same idea to denoising[1]

NL-means denoises a square reference patch $\tilde{P}$ around **i** of dimension $\kappa \times \kappa$ by replacing it by an average of all similar patches $\tilde{Q}$ in a square neighborhood of **i** of size $\lambda \times \lambda$. To do this, a normalized Euclidean distance between $\tilde{P}$ and $\tilde{Q}$, $d(\tilde{P}, \tilde{Q}) = \frac{1}{\kappa^2}\|\tilde{P} - \tilde{Q}\|^2$ is computed for all patches $\tilde{Q}$ is the search neighborhood. Then the weighted average is

[1]Nevertheless, some researchers have pointed out to us the report [**104**] as giving an early intuition that intuition could use signal redundancy. This very short paper describes an experiment in a few sentences. It suggests that region redundancy on both sides of an edge can be detected, and used for image denoising. Nevertheless, no algorithm is specified in this paper.



(a) (b) (c) (d)

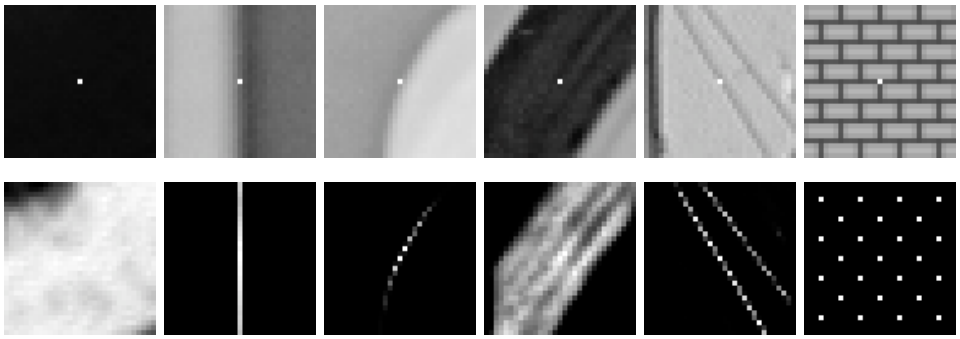
FIGURE 1. $q1$ (a) and $q2$ (c) have a large weight because their similarity windows are similar to that of p (b). On the other side, for $q3$ (d) weight $w(p, q3)$ is much smaller because the intensity grey values in the similarity windows are very different.

$$\hat{P} = \frac{\sum_{\tilde{Q}} \tilde{Q} e^{-\frac{d(\tilde{P},\tilde{Q})^2}{h^2}}}{\sum_{\tilde{Q}} e^{-\frac{d(\tilde{P},\tilde{Q})^2}{h^2}}}.$$

The thing that helps NL-means over the neighborhood filters is the concentration of the noise law, as the number of pixels increases. Because the distances are computed on many patch samples instead of only one pixel, the fluctuations of the quadratic distance due to the noise are reduced.

Related attempts: [**105**] proposed a "universal denoiser" for digital images. The authors prove that this denoiser is universal in the sense "of asymptotically achieving, without access to any information on the statistics of the clean signal, the same performance as the best denoiser that does have access to this information". In [**106**] the authors presented an implementation valid for binary images with an impulse noise, with excellent results. Awate and Whitaker [**107**] also proposed a method whose principles stand close to NL-means, since the method involves comparison between patches to estimate a restored value. The objective of the algorithm is to denoise the image by decreasing the randomness of the image.

A consistency theorem for NL-means. NL-means is intuitively consistent under stationarity conditions, namely if one can find many samples of every image detail. It can be proved [**32**] that if the image is a fairly general stationary and mixing random process, for every pixel **i**, NL-means



(a) (b) (c) (d) (e) (f)

FIGURE 2. On the right-hand side of each pair, one can see the weight distribution used to estimate a centered patch of the left image by NL-means. (a) In flat zones, the weights are uniformly distributed, NL-means acts like a low pass isotropic filter. (b) On straight edges, the weights are distributed in the direction of the edge (like for anisotropic filters). (c) On curved edges, the weights favor pixels belonging to the same contour. (d) In a flat neighborhood, the weights are distributed in a grey-level neighborhood (exactly like for neighborhood filters). In the cases of (e) and (f), the weights are distributed across the more similar configurations, even though they are far away from the observed pixel. This behavior justifies the "non local" appellation.

converges to the conditional expectation of $\mathbf{i}$ knowing its neighborhood, which is the best Bayesian estimate.

NL-means as an extension of previous methods. A Gaussian convolution preserves only flat zones, while contours and fine structure are removed or blurred. Anisotropic filters instead preserve straight edges, but flat zones present many artifacts. One could think of combining these two methods to improve both results. A Gaussian convolution could be applied in flat zones, while an anisotropic filter could be applied on straight edges. Still, other types of filters should be designed to specifically restore corners, or curved edges, or periodic texture. Figure 2 illustrates how NL-means chooses the right weight configuration for each sort of image self-similarity.

Algorithm 18 NL-means algorithm (parameter values for κ, λ are indicative).

Input: noisy image $\tilde{u}$, σ noise standard deviation.
Output: output denoised image.
Set parameter $\kappa = 3$: size of patches.
Set parameter $\lambda = 31$: size of research zone for which similar patches are searched.
Set parameter $h = 0.6\,\sigma$: bandwidth filtering parameter.
for each pixel **i** **do**
    Select a square reference patch $\tilde{P}$ around **i** of dimension $\kappa \times \kappa$.
    Set $\hat{P} = 0$ and $\hat{C} = 0$.
    for each patch $\tilde{Q}$ in a square neighborhood of **i** of size $\lambda \times \lambda$ **do**
        Compute the normalized Euclidean distance between $\tilde{P}$ and $\tilde{Q}$, $d(\tilde{P}, \tilde{Q}) = \frac{1}{\kappa^2}\|\tilde{P} - \tilde{Q}\|^2$.
        Accumulate $\tilde{Q} e^{-\frac{d(\tilde{P},\tilde{Q})^2}{h^2}}$ to $\hat{P}$ and $e^{-\frac{d(\tilde{P},\tilde{Q})^2}{h^2}}$ to $\hat{C}$.
    end for
    Normalize the average patch $\hat{P}$ by dividing it by the sum of weights $\hat{C}$
end for
for each pixel x **do**
    Aggregation: recover the denoised value at each pixel **i** by averaging all values at **i** of all denoised patches $\hat{Q}$ containing **i**
end for

2. Non-local Bayesian denoising

It is apparent that Equation (11) given in Chapter 5,

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

gives by itself a denoising algorithm, provided we can compute the patch expectations and patch covariance matrices. We shall now explain how the Non-local Bayes algorithm proposed in [**47, 91**] does it. Let $\mathcal{P}(\tilde{P})$ be the set of patches $\tilde{Q}$ similar to the patch $\tilde{P}$, which have obtained with a suitably chosen tolerance threshold, so that we can assume that they represent noisy versions of the patches similar to P. Then, by the law of large numbers, we have

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P}) - 1}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\Big(\tilde{Q} - \overline{\tilde{P}}\Big)\Big(\tilde{Q} - \overline{\tilde{P}}\Big)^t, \qquad \overline{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\tilde{Q}. \tag{16}$$

Nevertheless, the selection of similar patches at the first step is not optimal and can be improved in a second estimation step where the first step estimation is used as oracle. Thus, in a second step, where all patches have been denoised at the first step, all the denoised patches can be used again to obtain an estimation $\mathbf{C}_{\hat{P}_1}$ for $\mathbf{C}_P$, the covariance of the cluster containing P, and a new estimation of $\overline{\tilde{P}}$, the average of patches similar to $\tilde{P}$. Indeed, the patch similarity is better estimated with the denoised patches. Then it follows from (10) and (11) that we can obtain a second better denoised patch,

$$\hat{P}_2 = \overline{\tilde{P}}^1 + \mathbf{C}_{\hat{P}_1}\left[\mathbf{C}_{\hat{P}_1} + \sigma^2\mathbf{I}\right]^{-1}(\tilde{P} - \overline{\tilde{P}}^1) \tag{17}$$

where

$$\mathbf{C}_{\hat{P}_1} \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1) - 1}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\left(\hat{Q}_1 - \overline{\tilde{P}}^1\right)\left(\hat{Q}_1 - \overline{\tilde{P}}^1\right)^t, \qquad \overline{\tilde{P}}^1 \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\tilde{Q}. \tag{18}$$

We write the denoised patches $\overline{\tilde{P}}$ in (16) and $\overline{\tilde{P}}^1$ in (17). Indeed, in (17), the denoised version of $\tilde{P}$ computed as the average of noisy patches $\tilde{Q}$ whose denoised patch is similar to $\hat{P}_1$.

In short, the estimates (11) and (17) appear equivalent, but they are not in practice. $\mathbf{C}_{\hat{P}_1}$, obtained after a first denoising step, is a better estimation than $\mathbf{C}_{\tilde{P}}$. Furthermore, $\overline{\tilde{P}}^1$ is a more accurate mean than $\overline{\tilde{P}}$. It uses a better evaluation of patch similarities. All above quantities being computable from the noisy image, we obtain the two step algorithm 19.

As pointed out in [**47, 91**], the Nonlocal Bayes algorithm only is an interpretation (with some generic improvements like the aggregation) of the PCA based algorithm. This paper has a self-explanatory title: "Two-stage image denoising by principal component analysis with local pixel grouping." It is equivalent to apply a PCA on the patches similar to $\tilde{P}$, followed by a Wiener filter on the coefficients of $\tilde{P}$ on this PCA, or to apply formula (11) with the covariance matrix of the similar patches. Indeed the PCA computes nothing but the eigenvalues of the empirical covariance

Algorithm 19 Non local Bayes image denoising

1: **Input**: noisy image
2: **Output**: denoised image
3: **for** all patches $\tilde{P}$ of the noisy image **do**
4: Find a set $\mathcal{P}(\tilde{P})$ of patches $\tilde{Q}$ similar to $\tilde{P}$.
5: Compute the expectation $\overline{\tilde{P}}$ and covariance matrix $\mathbf{C}_{\tilde{P}}$ of these patches by

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

6: Obtain the first step estimation:

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

7: **end for**
8: Obtain the pixel value of the basic estimate image $\hat{u}_1$ as an average of all values of all denoised patches $\hat{Q}_1$ which contain $\mathbf{i}$.
9: **for** all patches $\tilde{P}$ of the noisy image **do**
10: Find a new set $\mathcal{P}_1(\tilde{P})$ of noisy patches $\tilde{Q}$ similar to $\tilde{P}$ by comparing their denoised "oracular" versions Q_1 to P_1.
11: Compute the new expectation $\overline{\tilde{P}}^1$ and covariance matrix $\mathbf{C}_{\hat{P}_1}$ of these patches:

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

12: Obtain the second step patch estimate

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

13: **end for**
14: Obtain the pixel value of the denoised image $\hat{u}(\mathbf{i})$ as an average of all values of all denoised patches $\hat{Q}_2$ which contain $\mathbf{i}$.

matrix. Thus, the method in [**91**] gets its Bayesian interpretation. A study on the compared performance of local PCA versus global PCA for TSID is actually proposed in [**70**].

CHAPTER 8

The "Noise Clinic": a universal denoiser

Arguably several thousands papers are dedicated to image denoising. Most papers assume a fixed noise model, mainly white Gaussian or Poissonian (Chapters 1 and 2). This assumption is only valid for raw images. Yet in most images handled by the public and even by scientists, the noise model is imperfectly known or unknown. End users only dispose of the result of a complex image processing chain (Chapter 3) effectuated by uncontrolled hardware and software (and sometimes by chemical means). For such images, it is possible to estimate from a single image a noise model which is simultaneously signal and frequency dependent, as we showed in Chapter 4. In this chapter we propose a multiscale denoising algorithm adapted to this broad noise model, based on the NL-Bayes denoising algorithm (Chapter 7). This leads to a blind denoising algorithm which we demonstrate on real JPEG images and on scans of old photographs for which the formation model is unknown. This algorithm is finally compared to the unique state of the art previous blind denoising method, based on Gaussian Scale Mixtures In The Wavelet Domain [**14**].

1. Introduction

1.1. Motivations. Blind denoising is the conjunction of a thorough noise estimation method followed by the application of an adapted denoising method. To cope with the broad variety of observed imaging noises, the noise model must be far more comprehensive than the usual white Gaussian noise. Our lead example will be JPEG images from digital CCD or CMOS cameras, where the initial signal dependent white Poisson noise has undergone nonlinear transforms, linear filters and a quantization of its DCT coefficients. After such alterations, a signal, frequency and scale dependency is a minimal assumption for the remaining noise. This requires dealing with a noise model depending on hundreds of parameters, in contrast with the usual one-parameter Gaussian white noise and the two-parameter Poisson noise. A flexible denoising method must also be conceived to cope with this signal, scale, and frequency dependent noise model.

To be useful to all image users, who generally have only access to the end result of a complex processing chain, blind denoising must be able to cope with both raw and preprocessed images of all sorts. The archives of the online executions at the IPOL journal of six classic denoising methods, namely DCT denoising [**108**], TV denoising [**109**], K-SVD [**110**], NL-means [**111**], BM3D [**112**] and NL-Bayes [**84**] are replete with such puzzling noisy images. IPOL users are in principle requested to upload noiseless images, to which the noise is added on line to test the performance of each algorithm. Yet, as one can observe in this public archive, the demand for a blind denoiser is so

strong that more than 10000 noisy images have been unduly uploaded. This shows how necessary "blind" methods are, for diffusing image processing techniques in science and technology.

1.2. Antecedents. We found only a few references on blind denoising approaches: Portilla [**37**], [**14**], Rabie [**113**] and Liu, Freeman, Szeliski and Kang [**52**]. Portilla's method is an adaptation of the famous BLS-GSM algorithm, which models wavelet patches at each scale by a Gaussian scale mixture (GSM), followed by a Bayesian least square (BLS) estimation for wavelet patches. This method is in principle adapted to homogeneous, Gaussian or mesokurtic noise. Yet, according to the author, the GSM model provides an automatic way to separate noise from signal. Indeed, for natural images, a GSM captures for the wavelet coefficients both high kurtosis marginals and a positive covariance between neighbor coefficient amplitudes. These coefficients are not shared by Gaussian or lower kurtosis noise sources. Then, for each wavelet subband a correlated Gaussian model can be used to estimate the noise and a correlated GSM is used for the signal. This algorithm is fully automatic, and will be compared to our results in Chapter 6.3. Our proposed solution shares many features with Portilla's method. Our noise model is nonetheless more general, being signal dependent, and our patch model is local, while the GSM wavelet patch model is global. (A recent local version of BLS-GSM [**38**] obtains a better performance than BLS-GSM.)

Liu, Freeman, Szeliski and Kang [**52**] proposed a unified denoising framework for JPEG images with two tasks in view: 1) automatic estimation and 2) removal of colored noise from a single image. These steps are performed by involving a piecewise smooth image model and a segmentation. The authors introduce the so called "noise level functions" (NLF) to estimate the noise level as a function of the image grey level. The obtained noise curve by their algorithm is an estimate of an upper bound of the real NLF, done by fitting a lower envelope to the standard deviations of per-segment image variances. In their denoising procedure, the chrominance of the colored noise is significantly removed by projecting pixel values onto a line fitted to the RGB values in each segment. Then, a Gaussian conditional random field is constructed to obtain the underlying clean image from the noisy input. Unfortunately no code is available for this complex procedure.

The method proposed by Rabie [**113**] seems less effective and works only for Gaussian noise. Here the blind denoising filter is based on the theory of robust statistics. The denoising part is done by minimizing a stationary cost function. For an adaptive window around the pixel of interest, noise pixels are seen as outlier pixels and rejected according to the Lorentzian robust estimator. The noise is basically estimated over a flat area of the noisy image. "Optimal-size" adaptive window are used to obtain the largest area containing relatively uniform structures around each pixel of interest. The uniformity is based on local signal variance estimate. This method seems less general than Portilla's method, since it can only deal with a signal-independent Gaussian noise. Observing the results shown in [**113**], indicates that this method mainly works on images with large homogeneous areas. An entropy-based noise level estimator has been proposed in [**114**], which may work for any sort of noise. Unfortunately it delivers a noise level but not a noise model. So we could not use it for noise estimation. Our denoising method will be based on a noise signal and frequency noise estimator proposed by Colom et al. [**10**], relying on a Ponomarenko et al. general

principle [**83**] to build a noise patch model. This method is proved in the aforementioned reference to estimate accurately the variances of DCT coefficients of noise patches in a JPEG image. We shall see that it can be easily extended to cope with a scale dependency.

2. A generalized nonlocal Bayesian algorithm

Most denoising methods in the literature focus on Gaussian white noise, which is a reasonable simplification of the problem, since for example Poisson noise can be transformed into approximately white Gaussian noise by the Anscombe transform [**85**]. In this chapter we show that one of them, the NL-Bayes method, designed for Gaussian white noise, can be extended to deal with a signal, scale and frequency dependent noise. NL-Bayes only requires the knowledge of a local Gaussian patch model and of a Gaussian noise model. It is therefore possible to extend the noise model to make obtain a denoising method compatible with a scale and signal dependent.

Like other patch based denoising methods, NL-Bayes denoises all noisy square patches extracted from the noisy image $\tilde{u}$ and then obtains the final denoised image $\hat{u}$ by replacing every image pixel value by an average of the denoised values obtained for this pixel in all denoised patches containing it. We shall denote by $\tilde{P}$ a reference patch extracted from the image, and by $\mathcal{P}(\tilde{P})$ a set of patches $\tilde{Q}$ similar to the reference patch $\tilde{P}$. Assuming that $\tilde{Q}$ follows a Gaussian model, a first basic estimation of the denoised patch P can be obtained [**78**] by

$$P^{\mathbf{basic}} = \overline{\tilde{P}} + [\mathbf{C}_{\tilde{P}} - \mathbf{C}_n]\, \mathbf{C}_{\tilde{P}}^{-1} \left(\tilde{P} - \overline{\tilde{P}}\right) \tag{19}$$

where

- $\overline{\tilde{P}}$ is the empirical average of the patches similar to $\tilde{P}$:

$$\overline{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})} \sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})} \tilde{Q}; \tag{20}$$

- $\mathbf{C}_n$ is the covariance matrix of the noise;
- $\mathbf{C}_{\tilde{P}}$ is the empirical covariance matrix of the patches similar to $\tilde{P}$, which may be obtained by

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P}) - 1} \sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})} \left(\tilde{Q} - \overline{\tilde{P}}\right)\left(\tilde{Q} - \overline{\tilde{P}}\right)^t. \tag{21}$$

For pure Gaussian signal-independent noise with variance σ^2, we simply have $\mathbf{C}_n = \sigma^2\mathbf{I}$. The above estimate would the optimal Bayesian estimate, if $\mathbf{C}_{\tilde{P}}$ and $\overline{\tilde{P}}$ were the true covariance matrix and expectation of the patches similar to $\tilde{P}$. In a second step, all the denoised patches obtained after the previous first step estimation can be reused by a classic Wiener argument to obtain a better unbiased estimation $\mathbf{C}_{\tilde{P}}^{\mathbf{basic}}$ for the covariance of the 3D group containing P. Similarly, a new estimation $\overline{\tilde{P}}^{\mathbf{basic}}$ of the average of patches similar to P can be obtained. This leads to a second Wiener-Bayes estimate

$$P^{\mathbf{final}} = \overline{\tilde{P}}^{\mathbf{basic}} + \mathbf{C}_{\tilde{P}}^{\mathbf{basic}} \left[\mathbf{C}_{\tilde{P}}^{\mathbf{basic}} - \mathbf{C}_n\right]^{-1} \left(\tilde{P} - \overline{\tilde{P}}^{\mathbf{basic}}\right). \tag{22}$$

2.0.0.1. Adaptation to Signal-Dependent Noise. As formulas (19) and (22) show, the above Bayesian principle is compatible with a patch noise model $\mathbf{C}_n$ depending on each patch $\tilde{P}$. The above formulas only require a good estimate of the covariance matrix of the noise associated with each group of similar patches. The algorithm computing this matrix is given in Chapter 3. The noise model being signal dependent, for each intensity $\mathbf{i}$ in the range intensity $[0, 255]$ of the image a noise covariance matrix $\mathbf{C}_{n\mathbf{i}}$ will be available. The noise model for each group of patches similar to $\tilde{P}$ will depend on $\tilde{P}$ through their mean $\mathbf{i}$. The reference intensity for the current 3D group $\mathcal{P}(\tilde{P})$ must therefore be estimated to apply formulas (19) and (22) with the appropriate noise covariance matrix. This intensity is simply estimated as the average of all pixels contained in $\mathcal{P}(\tilde{P})$.

2.0.0.2. Local Correction of the Covariance Matrix. The denoising performance strongly depends on the noise covariance matrices estimation. If the matrices $\{\mathbf{C}_{n\mathbf{i}}\}_{\mathbf{i}\in[0,255]}$ are not accurate enough, denoising can cause ugly artifacts, particularly in the first step. The noise estimation procedure from the image is always at risk of an overestimation, particularly when the image is small or when it contains a uniform texture which becomes indistinguishable from colored noise. If $\mathbf{C}_n$ is overestimated, then (19) risks adding "negative noise" to the image, because of the $-\mathbf{C}_n$ term in this equation. Thus, a conservative estimation strategy must be applied on the first Bayesian step to avoid noise overestimation artifacts. This strategy ensures that the noise variances are always smaller than the noisy patch variances. This sanity check based on the diagonal values of both $\mathbf{C}_{\tilde{P}}$ and $\mathbf{C}_n$ covariance matrices leads to the following more conservative estimate of the diagonal elements of the patch covariance matrix used in (19):

$$\forall p \in [\![0, \kappa^2 - 1]\!], \mathbf{C}_{\tilde{P}}(p,p) = \max\left(\mathbf{C}_{\tilde{P}}(p,p), \mathbf{C}_n(p,p)\right). \tag{23}$$

2.0.0.3. Homogeneous Area Detection. The original NL-Bayes algorithm **[78]** has a statistical test to determine if a 3D group belongs to a homogeneous area, and in this case the estimation of all patches is replaced by the global mean over all pixels contained in the 3D group. This criterion is merely based on the comparison of the empirical standard deviation of all pixels of $\mathcal{P}(\tilde{P})$ with σ^2.

In our generalization of this algorithm, σ doesn't exist since $\mathbf{C}_n \neq \sigma^2\mathbf{I}$. So this criterion must be adapted to better take into account $\mathbf{C}_n$ in the following way:

- First, compute the difference of the traces of both covariance matrices for each channel c,

$$\delta_c = \mathrm{Tr}(\mathbf{C}_{\tilde{P}}) - \mathrm{Tr}(\mathbf{C}_n). \tag{24}$$

- Denote by $\hat{\tilde{Q}}$ a first estimation of $\tilde{Q}$ obtained by (19). Then the basic estimate is $\forall \tilde{Q} \in \mathcal{P}(\tilde{P})$,

$$Q^{\mathbf{basic}} = \begin{cases} \overline{\overline{\tilde{P}}} & \text{if } \delta_c < \alpha \mathrm{Tr}(\mathbf{C}_n) \\ \hat{\tilde{Q}} & \text{if } \delta_c > \beta \mathrm{Tr}(\mathbf{C}_n) \\ t\hat{\tilde{Q}} + (1-t)\overline{\overline{\tilde{P}}} & \text{otherwise.} \end{cases} \tag{25}$$

where

$$t = \frac{\delta_c - \alpha \mathrm{Tr}(\mathbf{C}_n)}{\beta \mathrm{Tr}(\mathbf{C}_n) - \alpha \mathrm{Tr}(\mathbf{C}_n)}$$

and

$$\tilde{\tilde{P}} = \frac{1}{\#\mathcal{P}(\tilde{P})\kappa^2} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \sum_{p=1}^{\kappa} \sum_{q=1}^{\kappa} \tilde{Q}(p,q)$$

The thresholds $(\alpha,\ \beta)$ are chosen equal to $\left(-\frac{1}{3}, \frac{1}{3}\right)$. This (optional) correction which generally increases the PSNR is only used for the first step of the finest scale of the multiscale algorithm.

3. Obtaining the covariance matrix of noise patches

Colom et al., [**10**], proposed an adaptation of the Ponomarenko et al. [**83**] method estimating a frequency dependent noise to estimate noise in JPEG images. Given a patch size $\kappa \times \kappa$, the method extracts from the image a set with fixed cardinality of sample blocks with very similar patches in DCT space, which are therefore likely to contain only noise. These noise blocks are transformed by a DCT, and an empirical standard deviation of their DCT coefficients is computed. This gives a noise model that is proved in [**10**] to be accurately consistent with noise observed in JPEG images. This algorithm computes for every intensity $\mathbf{i}$ with a multi-frequency noise estimate given by a $\kappa^2 \times \kappa^2$ matrix

$$\mathbf{M_i} := \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}} \left(\mathcal{D}N_{\mathbf{i}}\right)^t\right) \tag{26}$$

where:

- $\mathcal{D}$ is the $\kappa^2 \times \kappa^2$ matrix of the discrete cosine transform (DCT) ;
- $N_{\mathbf{i}}$ denotes the $\kappa \times \kappa$ stochastic noise patch model at intensity $\mathbf{i}$.

3.1. Are noise covariances negligible in the block DCT space? The method of the preceding chapter only estimates the variances of the DCT coefficients of noise blocks and not their covariances. The covariance matrices are therefore assumed to be diagonal, which amounts to assume that the DCT decorrelates the noise. A formal argument can be given in favor of this assumption. Assume that the initial image noise was white Gaussian, and that the image has undergone a symmetric, real, periodic linear filter H. Then this filter corresponds to applying a diagonal operator to the image in the DCT frequency domain. Thus the noise covariance of the filtered noise remains diagonal in the DCT domain. Yet, this argument is only valid for a *global* image DCT. Here, because we need a signal dependent noise model, we are estimating it on *local* DCTs applied to each block. It is therefore no more true that the blocks have undergone a periodic convolution filter. Thus, it cannot be *exactly* true that after the application of a global linear filter, the noise block DCTs have a diagonal covariance. To check nonetheless the quantitative validity of this assumption, we tested three different filters applied to a white noise:

- $\mathbf{H}_1$ with coefficients $\frac{1}{4}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ supported by the pixels $(-\frac{1}{2}, -\frac{1}{2})$, $(\frac{1}{2}, \frac{1}{2})$, $(\frac{1}{2}, -\frac{1}{2})$, $(-\frac{1}{2}, \frac{1}{2})$;

κ	4	6	8	16
mean $\{\|C_{i,j}\|\}_{i\neq j}$	0.83	0.48	0.31	0.10
mean $\{\|C_{i,j}\|\}_{i=j}$	24.89	25.31	24.95	24.73
median $\{\|C_{i,j}\|\}_{i\neq j}$	0.04	0.03	0.02	0.01
median $\{\|C_{i,j}\|\}_{i=j}$	19.42	17.14	16.28	14.41

TABLE 1. Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_1$.

κ	4	6	8	16
mean $\{\|C_{i,j}\|\}_{i\neq j}$	0.48	0.28	0.19	0.06
mean $\{\|C_{i,j}\|\}_{i=j}$	14.59	13.95	14.45	14.23
median $\{\|C_{i,j}\|\}_{i\neq j}$	0.010	0.008	0.005	0.002
median $\{\|C_{i,j}\|\}_{i=j}$	6.75	4.50	3.77	2.35

TABLE 2. Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_2$.

- $\mathbf{H}_2$ the centered filter with coefficients $\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$;
- $\mathbf{H}_3$ the centered filter with coefficients $\frac{1}{88}\begin{pmatrix} 1 & 2 & 4 & 8 & 4 & 2 & 1 \\ 2 & 4 & 8 & 16 & 8 & 4 & 2 \\ 1 & 2 & 4 & 8 & 4 & 2 & 1 \end{pmatrix}$.

The noise image $\tilde{u}$ was a 256×256 Gaussian white noise with mean 128, and standard deviation $\sigma = 20$. After convolution, we extracted N distinct $\kappa \times \kappa$ patches $\{P_n\}_{n\in N}$ from the image and a 2D normalized DCT was applied on them. Finally, their empirical $\kappa^2 \times \kappa^2$ covariance matrix $\mathbf{C}$ was computed as

$$\begin{aligned} &\forall (p,q),(i,j) \in [\![0,\kappa-1]\!]^2, \\ &\mathbf{C}(p,q,i,j) = \frac{1}{N}\sum_{n=1}^{N} \hat{P}(p,q)\hat{P}(i,j) \\ &-\frac{1}{N^2}\left(\sum_{n=1}^{N}\hat{P}(p,q)\right)\left(\sum_{n=1}^{N}\hat{P}(i,j)\right) \end{aligned} \tag{27}$$

These covariances matrices can be visualized by the absolute value of their coefficients $|C_{i,j}|$, normalized in $[0,1]$ so that the largest coefficient is set equal to 1, and the smallest equal to 0. The following color code is used in the visualization: a coefficient appears in blue if it is near 0; in green if it is near 0.5 and in red if it is near 1. The results for various patch sizes are shown in Figure 1. This illustration and the quantitative tables 1, 2 and 3 confirm that the block DCT noise covariance matrices are nearly diagonal.

So from now on, only variance coefficients will be considered in DCT space.

3.2. Covariance matrix filtering. Since the noise covariance matrices can only be estimated for sparse bins in the intensity range, an interpolation must be applied to obtain a noise

κ	4	6	8	16
mean $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.22	0.15	0.10	0.04
mean $\{\lvert C_{i,j}\rvert\}_{i=j}$	9.28	8.94	9.04	8.51
median $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.020	0.016	0.010	0.003
median $\{\lvert C_{i,j}\rvert\}_{i=j}$	3.32	2.86	2.44	1.73

TABLE 3. Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_3$.

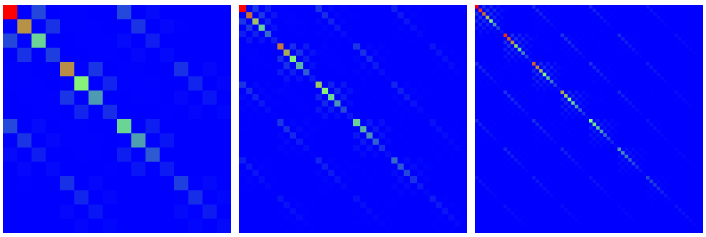


FIGURE 1. Visualization of the noise covariance matrices in DCT space after applying filter $\mathbf{H}_1$ to illustrate that it is almost diagonal. From left to right, top to bottom patch size $\kappa = 4, 6, 8$.

covariance matrix of the noise for each given intensity. The covariance matrices must be smoothed before such an interpolation. This can be obtained by a regularization of the covariance matrices in DCT space before applying the inverse DCT to get back a covariance matrix in the image domain. We found that a robust regularization could be performed in the following two steps:

(1) For each frequency independently, perform a linear interpolation between the bin values to obtain a noise curve for this frequency, giving the variance as a function of the signal $\mathbf{i}$. Smooth this curve by applying a sliding average;
(2) For every bin, replace each matrix coefficient by the median of its four neighbors and itself.

Since the filtering is channel independent, the pseudo-code only describes the filtering for one channel.

3.2.0.1. Getting Back to the Space Domain. For a given intensity $\mathbf{i}$, the covariance matrix of the noise is by definition

$$\mathrm{Cov}(N_{\mathbf{i}}) = \mathbb{E}\left(N_{\mathbf{i}} N_{\mathbf{i}}^t\right)$$

which leads to

$$\begin{aligned} \mathcal{D}\mathrm{Cov}(N_{\mathbf{i}})\mathcal{D}^t &= \mathcal{D}\mathbb{E}\left(N_{\mathbf{i}} N_{\mathbf{i}}^t\right)\mathcal{D}^t \\ &= \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}} N_{\mathbf{i}}^t \mathcal{D}^t\right) \\ &= \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}} \left(\mathcal{D}N_{\mathbf{i}}\right)^t\right) \\ &= \mathbf{M}_{\mathbf{i}} \end{aligned} \tag{28}$$

thanks to equation (26). Since $\mathcal{D}^{-1} = \mathcal{D}^t$, then from equation (28) we get

$$\mathrm{Cov}(N_{\mathbf{i}}) = \mathcal{D}^t \mathbf{M}_{\mathbf{i}} \mathcal{D}. \tag{29}$$

4. The multiscale algorithm

4.1. Why a multiscale algorithm? Classic denoising algorithms such as BM3D (Dabov et al. [**36**]), NL-means (Buades et al. [**86**]), K-SVD (Mairal et al. [**115**], [**116**]), Wiener filters applied on DCT (Yaroslavsky et al. [**117**], [**118**]) or on wavelet transform (Donoho et al. [**119**]) and the total variation minimization (Rudin et al. [**120**]) achieve good results for moderate noise ($\sigma \leq 20$). Yet for larger noise artifacts inherent to each method (and different for each method) start appearing. In particular all keep an often disturbing low frequency noise. A natural idea to deal with low frequency noise is to involve a coarse to fine multiscale procedure, which promises three improvements:

(1) in the patch-based methods, it favors a better patch comparison, because the patch low frequencies are denoised *before* grouping them by similarity for denoising their higher frequencies;
(2) at coarse scales the noise decreases by zoom out, and state-of-the-art algorithms work better;
(3) subsampling the image before denoising amounts to enlarge the size of the neighborhood on which the denoising is performed, thus permitting to grab and remove low frequency noise on larger regions.

A still stronger argument in favor of a multiscale procedure is that in most images submitted by users, the main bulk of the noise is contained in the low frequencies. This is explainable by several factors. In accurately scanned old photographs, the chemical noise is over-sampled and its grain has low frequency components. In JPEG images, compression has strongly attenuated high frequency noise components, but the low frequency components after the third octave are intact.

FIGURE 2. A multiscale process is required to remove the low frequency noise. This is particularly apparent in the flat image regions. From left to right: Noisy image ($\sigma = 30$), result of the "Classic NL-Bayes", result of the multiscale (three scales) NL-Bayes.

To define a coarse to fine multiscale structure, we proceed by a classic oversampled wavelet denoising strategy [**121**]. The image is convolved by a Haar "mother wavelet", which is nothing

but a box-filter $\mathbf{F}$ where each lower scale pixel is the mean of four samples in the higher scale. This cumulates the advantage of dividing the noise standard deviation by two and of maintaining the independence of the samples after down-sampling. By this process a white noise remains white after subsampling. A classic objection to this wavelet method is that the sub-sampled image is aliased and cannot be up-sampled after denoising. The classic wavelet method avoids this obstacle by denoising simultaneously the three wavelet components obtained by convolving the image with the three Haar wavelets, before reconstructing the finer scale. Yet when dealing with patch based methods, it is better to keep all frequency components together to perform a better nonlocal patch comparison. For this reason the proposed multiscale algorithm keeps and processes four channels that are partly redundant. The four channels are obtained by moving the sub-sampling grid by respectively $(0,0)$, $(1,0)$, $(0,1)$, $(1,1)$. In that way there is enough information for up-sampling after denoising the denoised images at the lower scale.

The above method is multiscale but does not take advantage of the sub-sampling in the lower scales to increase the algorithm speed. A normal multiscale algorithm is only $1+\frac{1}{4}+\frac{1}{16}+\cdots=\frac{4}{3}$ more complex than the single scale algorithm. Instead a multiscale algorithm keeping all sub-images when sub-sampling will be twice to five times slower, depending on the number of scales involved, (by default two). Yet, the redundancy of this denoising at lower scales notably increases the restoration quality. This is particularly important, as *any denoising error on a down-sampled image is amplified by a four-factor after upsampling.*

4.2. The mean sub-sampling method. We shall denote by s the current dyadic scale of the multiscale algorithm. For the particular case of white noise, the aim of the sub-sampling is to obtain from $\tilde{u}_s$ an image $\tilde{u}_{s+1}$ where the standard deviation of the noise has been divided by two compared to the noise contained in $\tilde{u}_s$. To get this result, one can use a filter $f(i,j)$ satisfying

$$\sum_{i,j} f(i,j) = 1 \quad \text{and} \quad \sum_{i,j} f(i,j)^2 = \frac{1}{4}.$$

The simplest filter coping with these conditions is the average filter $\mathbf{F}$, defined by

$$\mathbf{F}(i,j) = \begin{cases} \frac{1}{4} & \text{if } (i,j) \in [(0,0),(0,1),(1,0),(1,1)], \\ 0 & \text{otherwise.} \end{cases}$$

which averages each group of four adjacent neighboring pixels. There are four different filter+sub-sample results, as shown in figure 3. Moreover if the image $\tilde{u}_s$ is well-sampled, so is $\tilde{u}_s * \mathbf{F}$. Thus, the difference image is not aliased. Since all sub-sampled images are available, the noise estimation can work with the same amount of samples at every scale, which favors a good precision on the noise estimation at lower scales. All sub-sampled images must also be denoised. To avoid handling them separately, we introduce here a new procedure to process them jointly in a single image, while avoiding creating artificial borders. The four sub-sampled images are regrouped in one mosaic image, as shown in figure 4. The boundaries of the sub-images are in that way better denoised, because they are included in a smooth larger image.

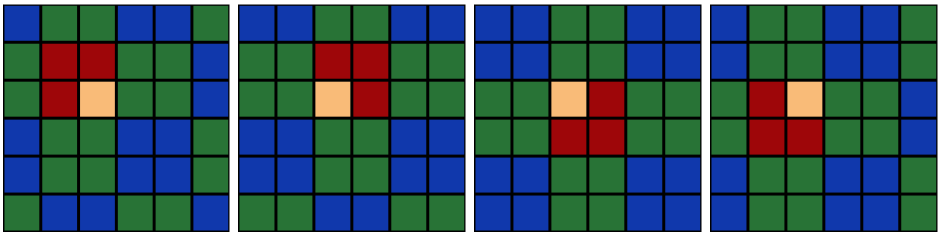


FIGURE 3. Four different ways to average red neighbors of the yellow reference pixel.

FIGURE 4. Left: mosaic of the scale 1 sub-images. Right: mosaic of the scale 2 sub-images, The input image has scale 0.

4.3. The mean up-sampling method. The aim of the up-sampling is to go back to the upper scale, after denoising the four sub-images obtained by sub-sampling as seen in Chapter 4.2. The four sub-images $\tilde{u}_1$, $\tilde{u}_2$, $\tilde{u}_3$ and $\tilde{u}_4$ have their pixel center (resp. in red, purple, green and blue in figure 5) located at the center of four pixels of $\tilde{u}$ (in black in figure 5). Thus they are shifted by $\pm\frac{1}{2}$ in both coordinate directions. The reconstruction of the pixels of $\tilde{u}$ (see the example of the pixel in yellow in figure 5) will be done by averaging their four neighbors, each one belonging to each sub-image.

4.4. Noise estimation. If the input noisy image had pure Gaussian noise, then after each sub-sampling the noise should be divided by two and remain white. For raw images it is the case, since (almost) no alteration nor transformations are applied to the original noisy pixels. Then the noise is a Poisson random process, which can be approximated by a signal-dependent Gaussian noise.

However, the proposed algorithm must deal with all kinds of noisy images. A large majority of them are JPEG images where JPEG has quantized DCT coefficients, making the energy decrease as the frequency increases. In such images the noise increases at lower scales, as illustrated in Figure 6, which are the noise curves of the image shown in Figure 16. This figure displays average

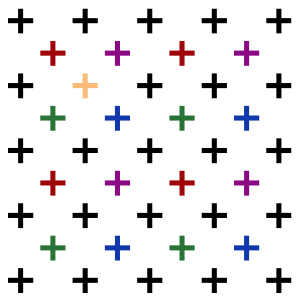


FIGURE 5. Position of the center of pixels in the original image $\tilde{u}$ in black, in the four sub-images $\tilde{u}_1$ in red, $\tilde{u}_2$ in purple, $\tilde{u}_3$ in green and $\tilde{u}_4$ in blue. The yellow pixel will be reconstructed by averaging the top left red pixel, the top right purple pixel, the bottom left green pixel and the bottom right blue pixel of its four pixel neighborhood.

noise curves for high and low frequencies respectively, in the three scales noise estimation from a JPEG image. The low-frequency noise is not altered by JPEG and becomes a high-frequency noise after three[1] subsampling operations.

In our redundant noise estimation, the noise covariance matrices are estimated at each dyadic scale. Chapter 4.2 explains how the noise estimation is applied on the mosaic image composed of all sub-images. Then for every scale the same number of samples is available, which allows the noise estimation to retain a decent accuracy even at coarse scales. At each given scale, all sub-images of the mosaic are denoised with the same set of noise covariance matrices.

The whole coarse to fine multiscale procedure is summarized in Algorithm 20. During the sub-sampling the four sub-images are kept and assembled in a mosaic to be denoised together. It follows that for each scale, the mosaic keeps the original image size. Thus the complexity for the whole algorithm is approximately equal to N times the complexity of the one scale algorithm. In the sequel we shall call our proposed algorithm the "Noise Clinic" as it combines a diagnose of the image illness with an immediate cure.

5. Validation

Blind denoising is designed mainly for images where the image history is unknown and no ground truth available. But we can test the denoising performance of the Noise Clinic after simulating a whole image processing chain on a Poisson noisy image for which the ground truth is available. One of the worst possible noise distortion is provided by the image processing chain applied in the camera hardware and generally ending with JPEG compression. This chain includes nonlinear corrections on the raw image, followed by some denoising, demosaicing, gamma-correction, white balance and JPEG compression, namely the quantization of local block DCT coefficients. To see to which extent the method works, we started with perturbations consistent with our noise model

[1]Since JPEG transform is based on the 8×8 DCT transform, after three subsamplings the 8×8 pixels patches become a single pixel. Thus, at the third scale the noise is only high-frequency and uncorrelated.

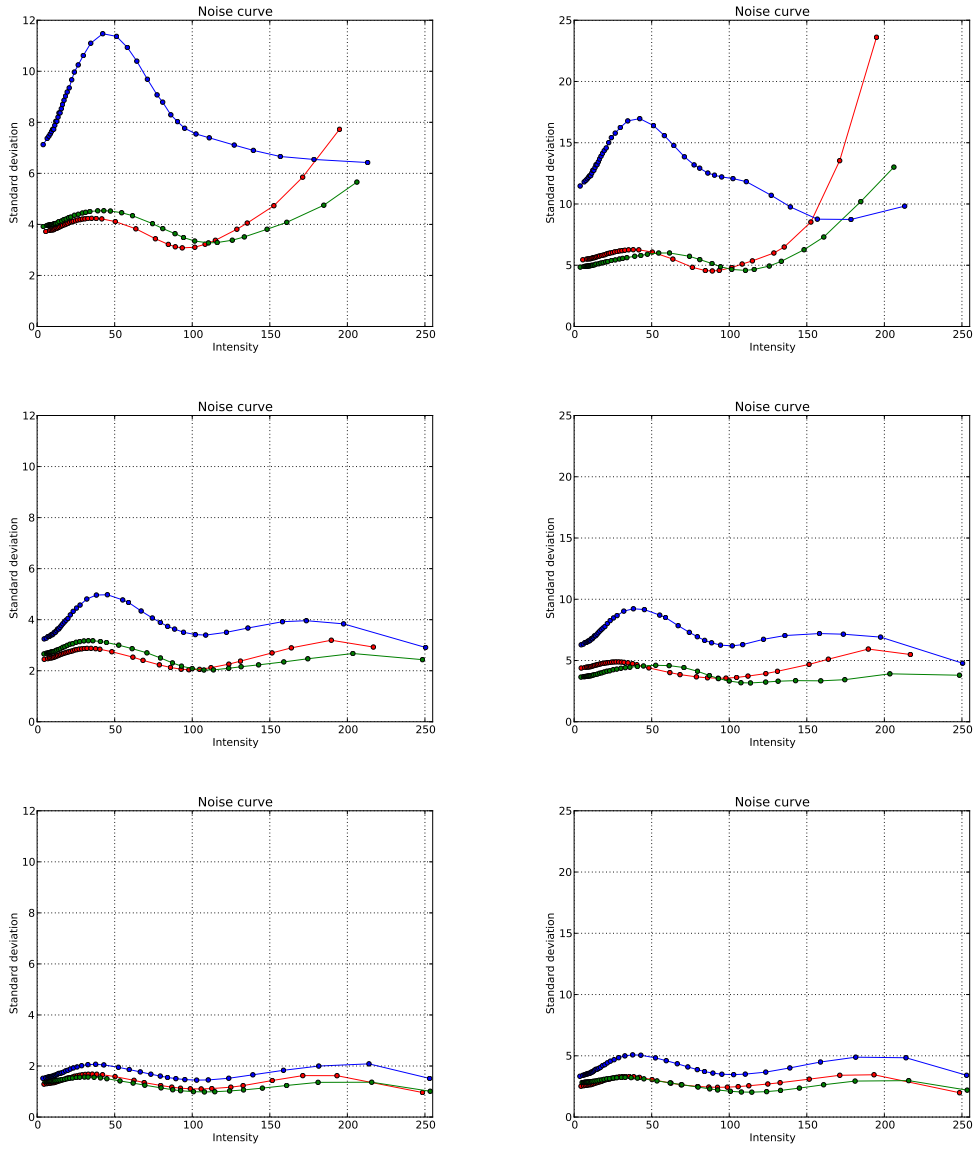


FIGURE 6. Average noise curves for a typical JPEG-encoded image (shown in Figure 16). From left to right: low frequencies, high frequencies. From top to bottom: scale 2, scale 1, scale 0. Instead of being divided by two at each scale (as it should happen with white noise), the noise grows in lower scales, where JPEG has not removed it.

and then simulated a typical camera image processing chain ending with JPEG compression. We first obtained a noise-free raw image u_{raw} by subsampling a high quality outdoor image. Then a Poisson noisy $\tilde{u}_{\text{raw}}$ was simulated from it. Four validation experiments were performed.

First, we computed a reference denoised version of the image:

- the Noise Clinic was directly applied on $\tilde{u}_{\text{raw}}$ to get $\hat{u}_{\text{raw}}$;

Algorithm 20 Noise Clinic

Input : Noisy image $\tilde{u}_0$
Input : Number of scales N
Output : Denoised image $\hat{u}_0$
Part 1 : Builds the image scale pyramid and records the difference images
for each scale $s = 1$ to $N - 1$ **do**
  Let $\{\tilde{u}^k_{s-1}\}_{k\in[\![1,4^{s-1}]\!]}$ be the set of noisy subsampled images obtained at the previous scale. (For scale $s = 1$, it is $\tilde{u}_0$);
  for $k = 1$ to 4^{s-1} **do**
    Downsample $\tilde{u}^k_{s-1}$ into 4 sub-images : $\{\tilde{u}^{4(k-1)+i}_s\}_{i\in[\![1,4]\!]}$ as described in Chapter 4.2;
    Save difference images for this scale :

$$\tilde{d}^k_{s-1} = \tilde{u}^k_{s-1} - \mathbf{U}\left(\{\tilde{u}^{4(k-1)+i}_s\}_{i\in[\![1,4]\!]}\right)^k.$$

  end for
  if $s = N - 1$ **then**
    Set $\{\tilde{v}^k_{N-1}\}_k = \{\tilde{u}^k_{N-1}\}_k$
    Build the noisy mosaic image $\tilde{m}_{N-1}$ from the set of sub-images $\{\tilde{v}^k_{N-1}\}_{k\in[\![1,4^{N-1}]\!]}$.
  end if
end for
Part 2 : Estimates noise and denoises bottom-up in the pyramid
for $s = N - 1$ to 0 **do**
  Estimate the noise covariance matrices on the mosaic $\tilde{m}_s$ as explained in Chapter 3;
  Denoise the noisy mosaic image $\tilde{m}_s$ with the NL-Bayes extension of Chapter 2, using the noise covariance matrices $\{\mathcal{D}^t\mathbf{M_i}\mathcal{D}\}_\mathbf{i}$ to obtain $\hat{m}_s$;
  if $s > 0$ **then**
    for $k = 1$ to 4^{s-1} **do**
      Up-sample $\{\hat{u}_{s,4(k-1)+i}\}_{i\in[\![1,4]\!]}$ and add the saved details $\tilde{d}^k_{s-1}$ to get $\tilde{v}^k_{s-1}$ as described in Chapter 4.3
    end for
    Construct the mosaic image of the next scale $\tilde{m}_{s-1}$ from the set of sub-images $\{\tilde{v}^k_{s-1}\}$.
  else
    $\hat{u}_0 = \hat{u}^1_0$
  end if
end for

- a white balance and a gamma correction were applied on u_{raw}, $\tilde{u}_{\text{raw}}$ and $\hat{u}_{\text{raw}}$ to get u_{rgb}, $\tilde{u}_{\text{rgb}}$ and $\hat{u}_{\text{rgb}}$.

Those images will be used as reference, to see how other parts of the image processing chain (such as the demosaicing and the JPEG compression) impact the result of the denoising. Table 4 shows RMSEs between the noisy and denoised images and the reference one. One can also remark that

$\tilde{u}_{\text{rgb}}$	$\hat{u}_{\text{rgb}}\|^{s2}$	$\hat{u}_{\text{rgb}}\|^{s3}$	$\hat{v}_{\text{rgb}}$
8.62	3.63	3.65	6.46

TABLE 4. RMSE between noisy/denoised images and corresponding reference image (u_{raw}) when two and three scales are used. $\hat{v}_{\text{rgb}}$ denotes the result of Blind BLS-GSM for this experiment.

$\tilde{u}_d$	$\hat{u}_d\|^{s2}$	$\hat{u}_d\|^{s3}$	$\hat{v}_d$
8.64	4.84	4.84	6.43

TABLE 5. RMSE between noisy/denoised images and corresponding reference image (u_d) when two and three scales are used for the demosaicking experiment. $\hat{v}_d$ denotes the result of Blind BLS-GSM for this experiment.

the best result of the denoising (both in term of RMSEs and visual aspects) is obtained when the Noise Clinic is applied directly before any transformation.

Second, a demosaicing algorithm was added to the image processing chain before calling the denoising part:

- extract the mosaic[2] of the noise-free image: $u_m = \text{Mosaic}(u_{\text{raw}})$;
- do the same for the noisy image: $\tilde{u}_m = \text{Mosaic}(\tilde{u}_{\text{raw}})$;
- apply a classic demosaicing method[3] on both images, followed by a white balance and a gamma correction to get u_d and $\tilde{u}_d$;
- finally apply the Noise Clinic on $\tilde{u}_d$ to get $\hat{u}_d$.

Table 5 shows RMSEs for this experiment. One may notice that after a demosaicing the noise is no more white, and some structures appears in the noise. These structures are preserved and sometimes enhanced by the denoising algorithm, since it is seen as structure and not as noise. This explains why RMSEs are less favorable than when the denoising is directly applied on the raw images.

Third, a complete image processing chain was simulated to obtain a final JPEG compressed image:

- apply a JPEG compression of quality 92 over both u_d and $\tilde{u}_d$ to get u_{jpeg} and $\tilde{u}_{\text{jpeg}}$;
- apply the Noise Clinic to get $\hat{u}_{\text{jpeg}}$.

Table 6 shows RMSEs for this experiment. Of course, as JPEG compression creates more artifacts and structured noise, results are worse than with the first two experiments. This only means that the denoising should be applied as soon as possible in the whole image processing chain. However,

[2]The mosaic image is obtained by keeping only the Bayer (R Gr Gb B) over a group of four pixels instead of all RGB values.

[3]The demosaicing algorithm used in this experiment was Self-similarity Driven Demosaicking algorithm [**122**], available on IPOL.

$\tilde{u}_{\text{jpeg}}$	$\hat{u}_{\text{jpeg}}\|^{s2}$	$\hat{u}_{\text{jpeg}}\|^{s3}$	$\hat{v}_{\text{jpeg}}$
8.70	5.34	5.53	6.30

TABLE 6. RMSE between noisy/denoised images and corresponding reference image (u_{jpeg}) for the *JPEG* experiment, with compression quality of 92. $\hat{v}_{\text{jpeg}}$ denotes the result of Blind BLS-GSM for this experiment.

$\tilde{u}_f$	$\hat{u}_{f_1}\|^{s2}$	$\hat{u}_{f_1}\|^{s3}$	$\hat{u}_{f_2}\|^{s2}$	$\hat{u}_{f_2}\|^{s3}$	$\hat{v}_{f_1}$	$\hat{v}f_2$
1.58	0.75	0.82	0.75	0.75	1.24	1.28

TABLE 7. RMSE between noisy/denoised images and corresponding reference image (u_f) for the *filtered* experiment. $\hat{v}_{f_1}$ and $\hat{v}_{f_2}$ denote results of Blind BLS-GSM for this experiment.

results are not very far from the ideal case, which confirms the interest and the strength of the Noise Clinic.

Fourth, the filter $\mathbf{H}_2$ seen in Chapter 3.1 was used to get:

- a reference filtered image: $u_f = \mathbf{H}_2 * u_{\text{raw}}$;
- a noisy filtered image: $\tilde{u}_f = \mathbf{H}_2 * \tilde{u}_{\text{raw}}$;
- the result of the Noise Clinic of the noisy filtered image: $\hat{u}_{f_1} = NC(\mathbf{H}_2 * \tilde{u}_{\text{raw}})$;
- the filtered result of the Noise Clinic of the noisy image: $\hat{u}_{f_2} = \mathbf{H}_2 * NC(\tilde{u}_{\text{raw}})$.

Table 7 shows RMSEs associated to this experiment. Of course after this filtering, there only remains low frequency noise, which explains why RMSEs values are better than in the ideal case. However, the Noise Clinic is still able to give good results.

Figure 7 (resp. 8 and 9) shows results associated of the raw experiment (resp. demosaicking and JPEG).

Figure 10 (resp. 11 and 12) shows a comparison between the Noise Clinic and Blind BLS-GSM for the raw experiment (resp. demosaicking and JPEG).

6. Results

6.1. Detailed results. In this chapter we applied the blind denoising to real noisy images for which no noise model was available. To illustrate the algorithm structure and its action at each scale, we present for each experiment the noisy input image and for each scale:

- the noisy image where noise has already been removed at coarser scales;
- the denoised image at this scale;
- the difference image = noisy - denoised at this scale;
- the average noise curve over high frequencies;
- the average noise curve over low frequencies.

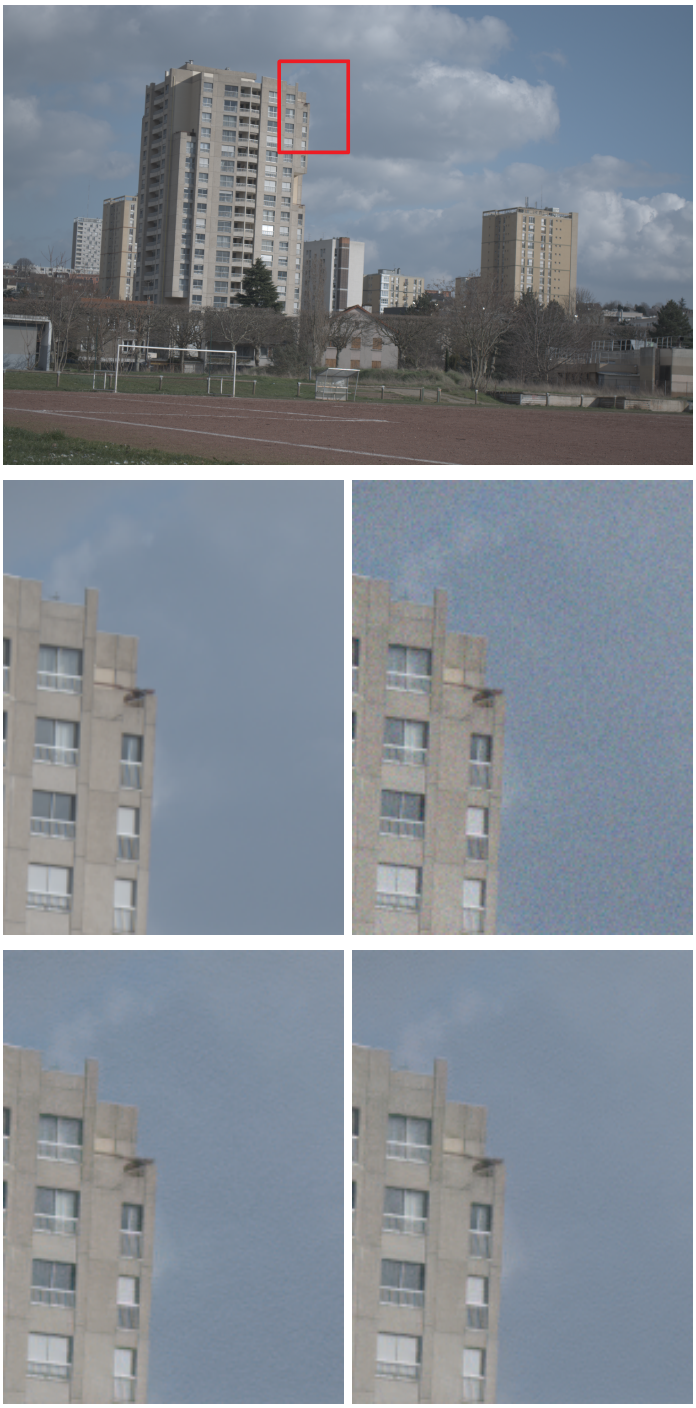


FIGURE 7. Visual results of the reference (first) experiment. From top to bottom, and left to right: full noise-free image, crop of the noise-free image u_{rgb}, crop of the noisy image $\tilde{u}_{\mathrm{rgb}}$, crop of the result of the Noise Clinic using two scales $\hat{u}_{\mathrm{rgb}}|^{s2}$ and crop of the result of the Noise Clinic using three scales $\hat{u}_{\mathrm{rgb}}|^{s3}$.

FIGURE 8. Visual results of the demosaicking (second) experiment. From top to bottom, and left to right: crop of the noise-free image u_d, crop of the noisy image $\tilde{u}_d$, crop of the result of the Noise Clinic using two scales $\hat{u}_d|^{s2}$ and crop of the result of the Noise Clinic using three scales $\hat{u}_d|^{s3}$.

For each scale larger than 1, the subsampled images are up-sampled to keep the original image size. Similarly, the noisy image shown at each scale is the sum of the upsampled version of the denoised sub-images of the previous scale and of the still noisy difference image kept in reserve. In other terms this image contains the remaining noise at the current scale; the noise at coarser scales has in principle already been removed. Visual results are shown in Figure 13. The corresponding noise curves are presented in Figure 14. The experiments made on JPEG photographs from unknown sources are obviously noisy but, as the noise curves illustrate, the noise is not white and is signal dependent. This is easily detected by the fact that the noise curves are not flat and that they are not divided by two from a scale to the next, as they should if the noise were white. A typical fact of JPEG images is that the noise increases at the lower scales. This confirms the necessity of a multiscale algorithm.

FIGURE 9. Visual results of the JPEG (third) experiment. From top to bottom, and left to right: crop of the noise-free image u_{jpeg}, crop of the noisy image $\tilde{u}_{\text{jpeg}}$, crop of the result of the Noise Clinic using two scales $\hat{u}_{\text{jpeg}}|^{s2}$ and crop of the result of the Noise Clinic using three scales $\hat{u}_{\text{jpeg}}|^{s3}$.

6.2. Influence of the number of scales. Theoretically any number of scales could be used. Indeed at a very coarse scale the noise should be almost null and estimated as such, so that no denoising eventually would occur at very coarse scales. In practice however, some structure of the image may be confused with noise in the noise estimation step. Indeed the noise estimation method is tight on very large images on which pure noise samples in large numbers can be found [**83**]. After several subsamplings, the image becomes too small, and the risk of confusing texture with noise increases. In consequence applying a blind denoising on a small image is increasingly at risk of removing detail when the scale increases. Thus, it is almost always better to use a minimal number of scales, in most cases not more than two. However, we found that for some images with large low frequency noise it is sometimes better to use up to five scales. From that point of view our "blind denoising" is not fully blind and requires an user evaluation of the number of scales involved.

FIGURE 10. Visual comparison of the reference (first) experiment. From left to right: crop of the result of the Noise Clinic using three scales $\hat{u}_{\mathrm{rgb}}|^{s3}$ and crop of the result of the Blind BLS-GSM algorithm $\hat{v}_{\mathrm{rgb}}$.

FIGURE 11. Visual comparison of the demosaicking (second) experiment. From left to right: crop of the result of the Noise Clinic using three scales $\hat{u}_d|^{s3}$ and crop of the result of the Blind BLS-GSM algorithm $\hat{v}_d$.

Nevertheless our default value is two, and works on a large majority of the images. Illustrations of the use of the "right" number of scales are presented in Figure 15 .

For the "Palace" image in Figure 15, five scales are needed to obtain a noise-free result because of the huge low-frequency noise. In the difference image using five scales one can see that some image structure has been included in the noise. Yet, this low frequency loss is harmless, being undetectable in the resulting denoised image.

6.2.0.1. Result on typical low-light JPEG image . The amount of noise is directly related to the amount of light during the acquisition. Images as shown in Figure 16, taken in a bar with low light conditions are typically very difficult to denoise, even if we had directly access to the RAW

FIGURE 12. Visual comparison of the JPEG (third) experiment. From left to right: crop of the result of the Noise Clinic using three scales $\hat{u}_{\text{jpeg}}|^{s3}$ and crop of the result of the Blind BLS-GSM algorithm $\hat{v}_{\text{jpeg}}$.

image, due to the huge amount of noise. One can observe big colored spots caused by demosaicing. JPEG compression ends up creating structured noise. The big colored spots are well attenuated by blind denoising, but the structure created by JPEG is partly left. This is easily explained. These artifacts present sharp recurrent structures which are necessarily confused with signal in an algorithm based on image self-similarity.

6.2.0.2. Results on Old Photographs . Scanned old photographs form a vast image corpus for which the noise model can't be anticipated. The noise is chemical, generally with big grain and further altered by the scanning and JPEG encoding. Figures 17 and 18 show results obtained by the Noise Clinic over this kind of noisy images.

6.3. Comparison to one of the very few available blind denoising algorithms. We end this experimental chapter with a comparison of the Noise Clinic with blind BLS-GSM introduced in [**14**] and [**37**], a state-of-the-art blind denoising algorithm. The comparison was performed on several images with various noise models. BLS-GSM also is a multiscale algorithm modeling wavelet coefficient patches at each scale and making a global sophisticated Bayesian estimation of them as a Gaussian mixture. NL-Bayes instead has a simpler, but local patch Gaussian model. The global patch model in BLS-GSM has to be more complex to cope with the global patch variability.

In Figure 19 noisy images present strongly structured periodic noise, which is remarkably removed by the blind BLS-GSM algorithm, whereas our blind denoising keeps it and even reenforces it. However one can argue that this structured noise may be seen as a repetitive texture belonging to the image and therefore must be treated as detail and not as noise.

In Figure 20 the noise is more "normal" and closer to what can be expected from a natural image, and our blind denoising performs better. Blind BLS-GSM manages to remove some noise, but a slightly structured noise still remains, appearing in horizontal strips.

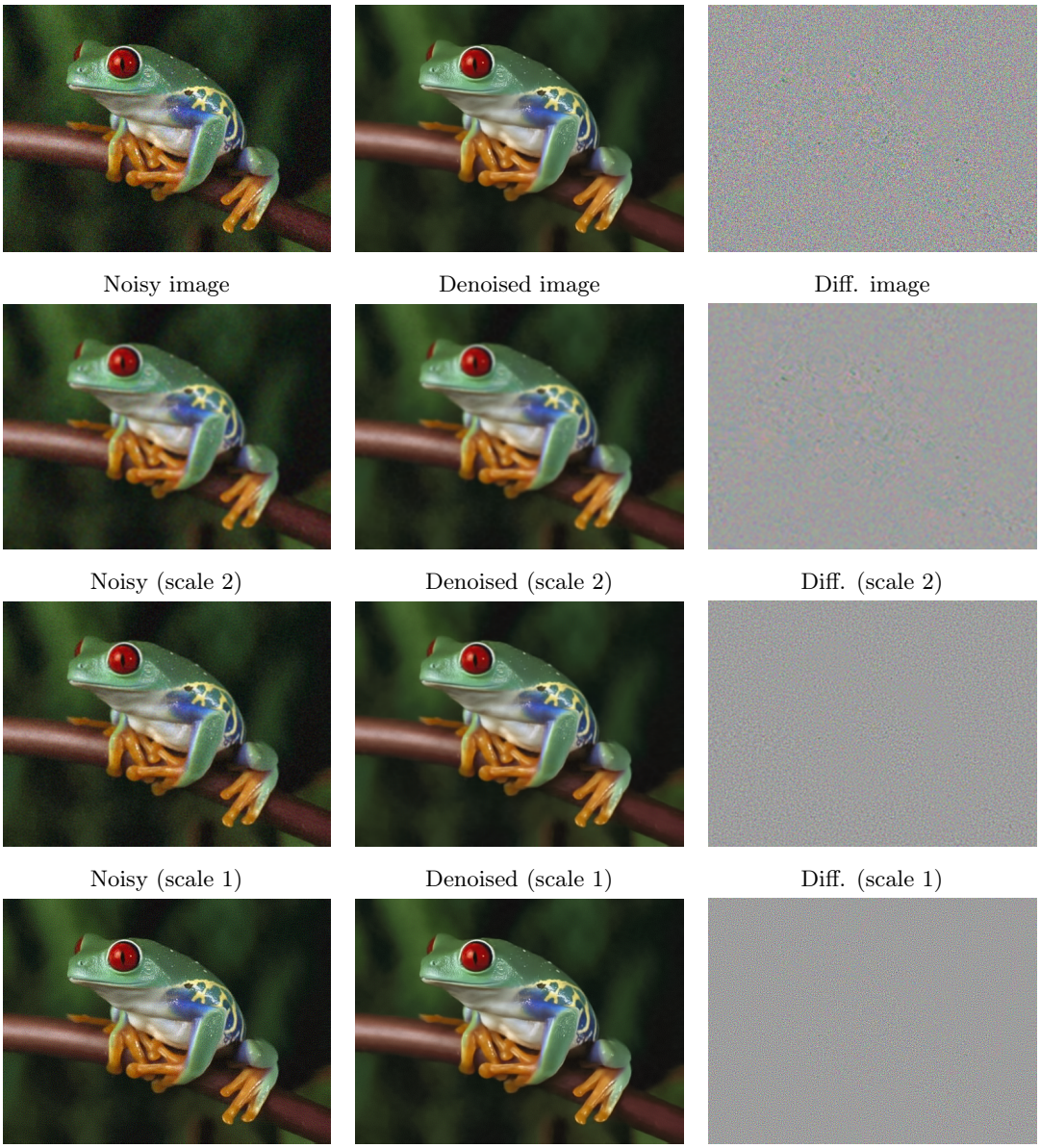
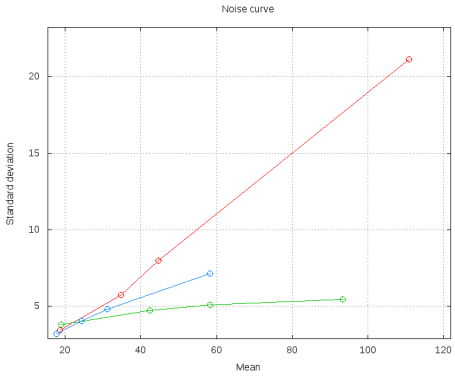


FIGURE 13. Illustration of blind denoising of a JPEG image, the "Frog" image. It is advised to zoom in the high quality .pdf to see detail.

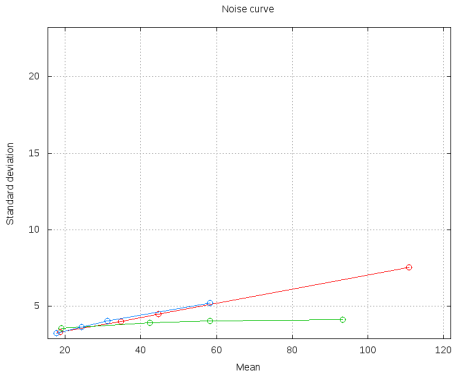
Figures 21, 22 and 23 show comparisons for low-light JPEG image and old Photographs presented in Chapters 6.2.0.1 and 6.2.0.2

7. Discussion

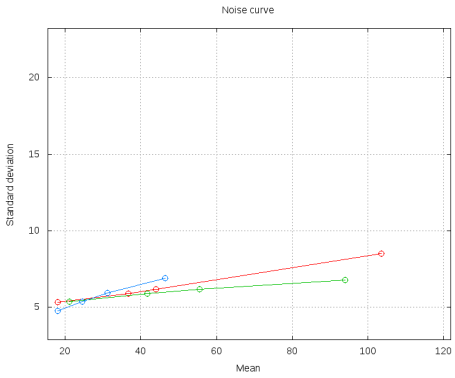
Blind denoising can be performed with minimal assumptions on the nature of the noise. We observed good results on almost any natural image, even if it had been modified by destructive applications such as JPEG compression or chemical processes. Particularly in old photographs,



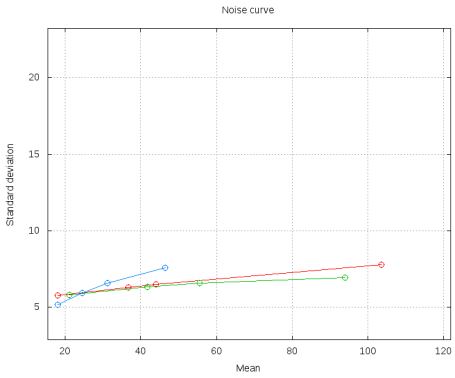
Low frEquation av. curve (s. 2)



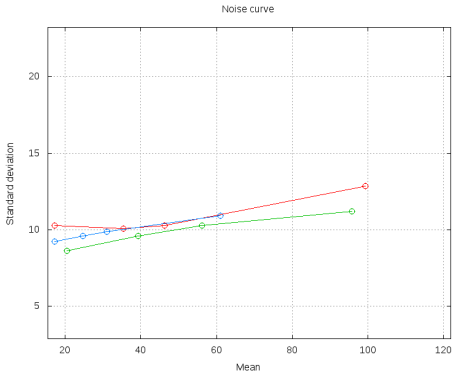
High frEquation av. curve (s. 2)



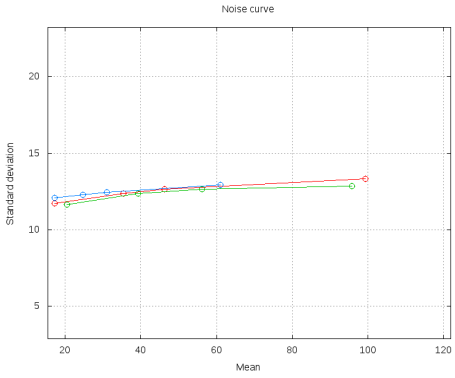
Low frEquation av. curve (s. 1)



High frEquation av. curve (s. 1)



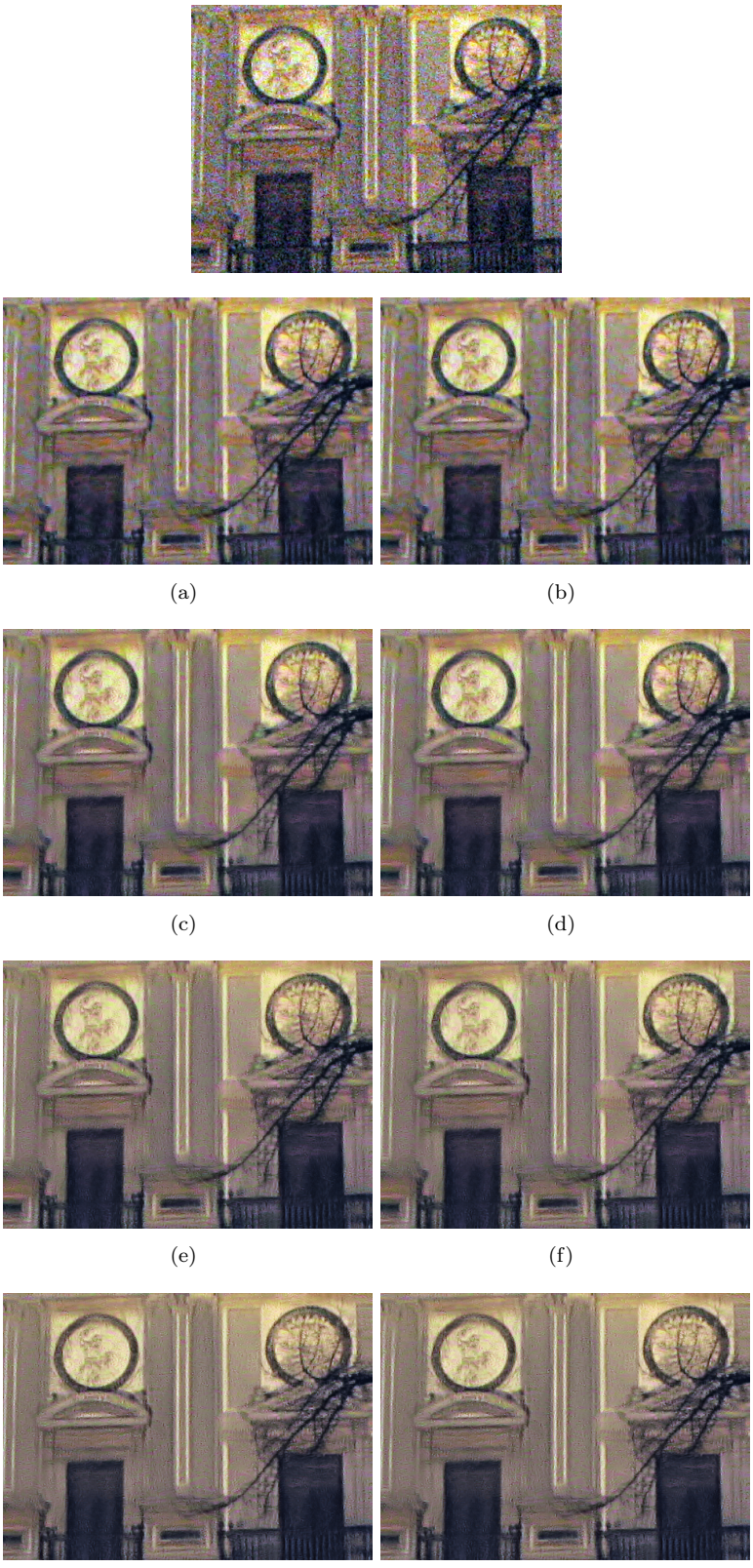
Low frEquation av. curve (s. 0)



High frEquation av. curve (s. 0)

FIGURE 14. Noise estimation of the "Frog" image: The noise in this image is clearly colored: it increases with descending octaves instead of being divided by two, as it should if it were white.

noise can acquire a thick grain which is only efficiently denoised at low scales. This method does not apply to impulse or multiplicative noise and should be extended to such alterations. Also our local noise estimation procedure did not detect the strength of the fully structured noise present



(a) (b)

(c) (d)

(e) (f)

(g) (h)

FIGURE 15. Blind denoising when varying the number of scales on “Palace”.

FIGURE 16. Blind denoising on "Bar", using three scales. From left to right, top to bottom : input noisy image, crop of the noisy image, crop of the output denoised image, crop of the difference image.

in the third infrared image of Figure 19. The case of a globally frequency dependent noise is of course better treated by Portilla's method which assumes a global noise model.

We wrote that the proposed method was "signal, scale and frequency" dependent. In fact as indicated by the preceding caveat, the method estimates and processes noise frequencies in the DCT of small blocks. So these frequency coefficient are far less precise than global image frequencies. Furthermore they are scale dependent, since we applied a dyadic subsampling procedure. Since at each dyadic scale, frequencies are estimated for blocks with at least 4×4 size, it follows that these scale dependent frequencies overlap. This leads to a redundant denoising since left-over noise at a coarse scale can be estimated again, and removed again at the overlapping finer dyadic scale. This redundancy of estimators is particularly necessary for such a complex noise model. The fact that JPEG images can be denoised in that way was far from granted. Indeed, it is impossible to really model noise in JPEG images, which are the result of a chain of nonlinear operators. It can be argued that our noise signal, frequency and scale dependent noise estimation is not yet general enough to cope with such alterations. This objection is definitely valid for block artifacts apparent in strong JPEG compression. Thus, strongly compressed images where blocking effects dominate remain beyond our scope.

FIGURE 17. Blind denoising on "Marilyn", using two scales. From left to right, top to bottom : input noisy image, crop of the noisy image, crop of the output denoised image, crop of the difference image.

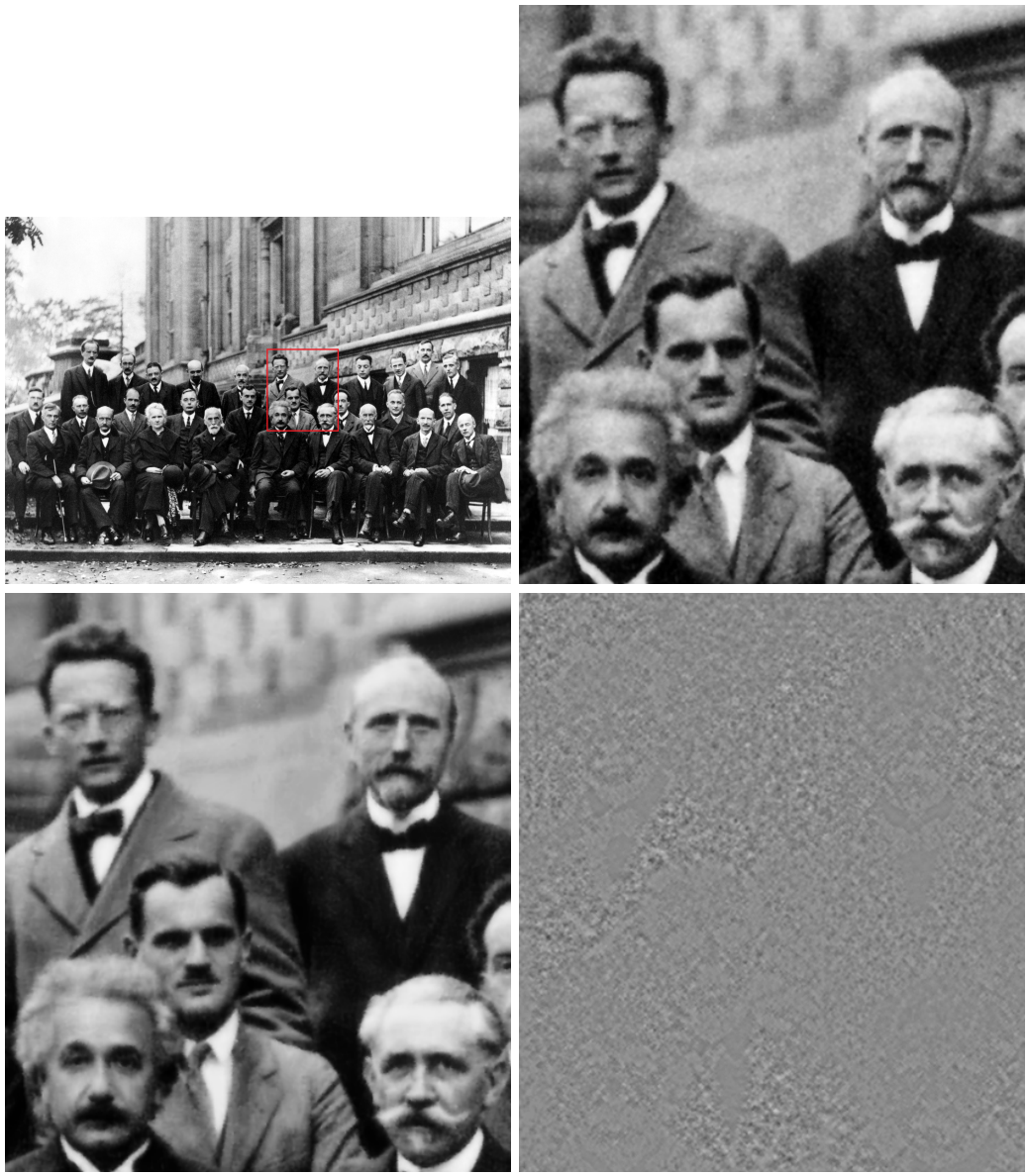


FIGURE 18. Blind denoising on "Solvay conference, 1927", using three scales. From left to right, top to bottom : input noisy image, crop of the noisy image, crop of the output denoised image, crop of the difference image.

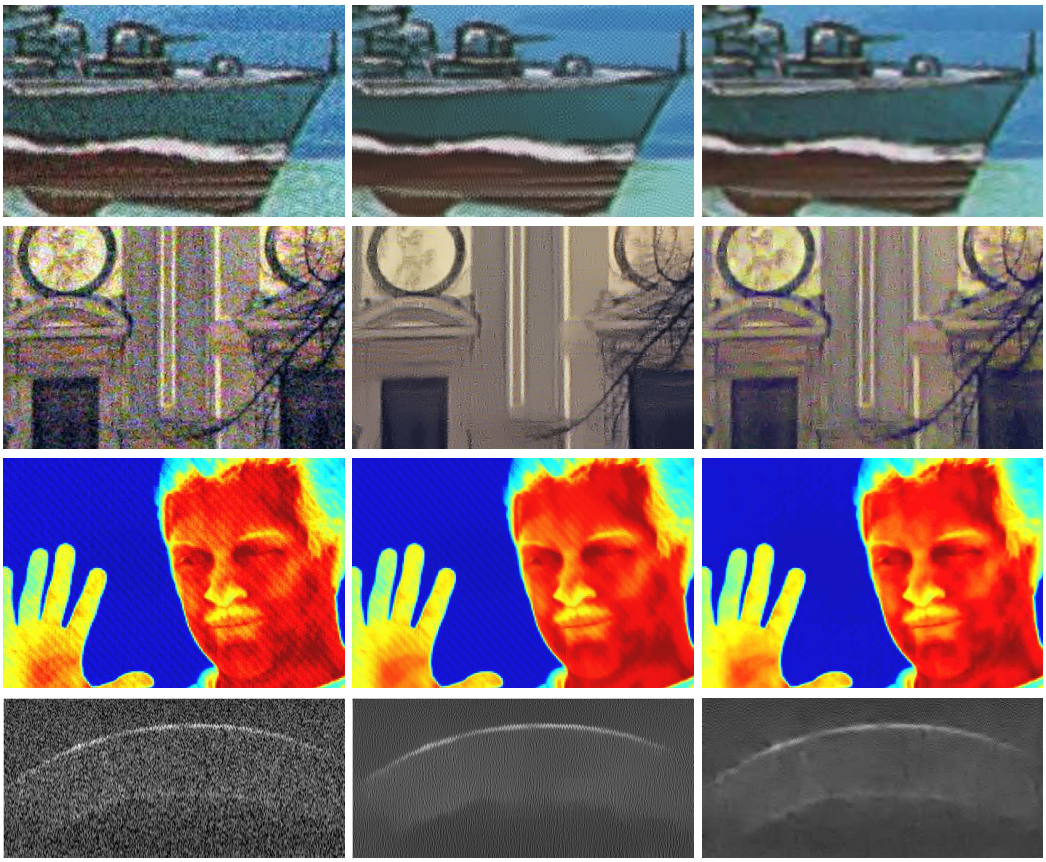


FIGURE 19. Results of our blind denoising and of Blind BLS-GSM on several images from [**14**]. From left to right: Noisy image, result of the Noise Clinic, result of the Blind BLS-GSM algorithm. It is advised to zoom in by a 300% factor the digital document to examine details.

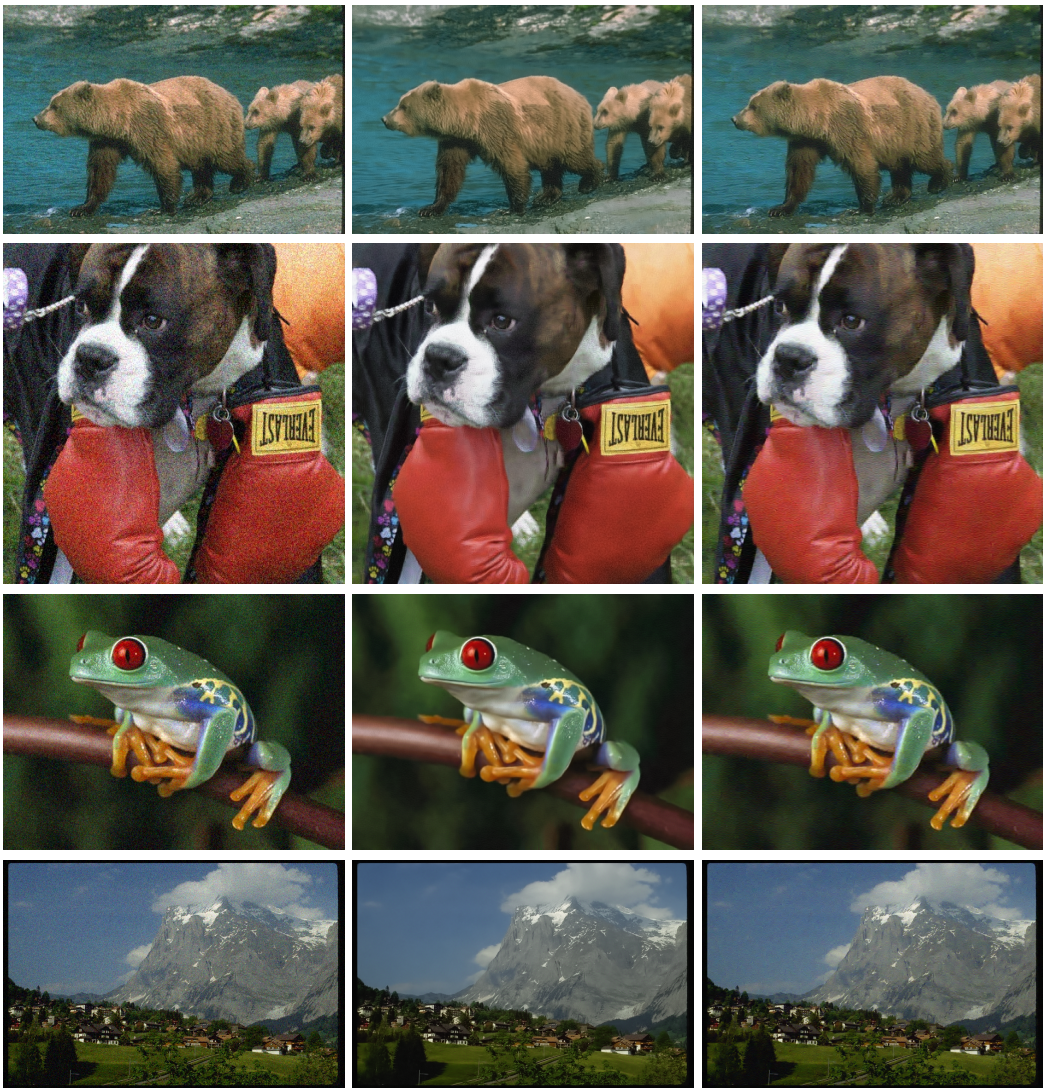


FIGURE 20. Comparing our blind denoising with Blind BLS-GSM on several images. It is advised to zoom in by a 400% factor the digital document to examine details. From left to right: Noisy image, result of the Noise Clinic, result of the Blind BLS-GSM algorithm.

FIGURE 21. Blind denoising on "Bar". From left to right: crop of the result of the Noise Clinic by using three scales and crop of the result of the Blind BLS-GSM algorithm.

FIGURE 22. Blind denoising on Marilyn". From left to right: crop of the result of the Noise Clinic by using two scales and crop of the result of the Blind BLS-GSM algorithm.

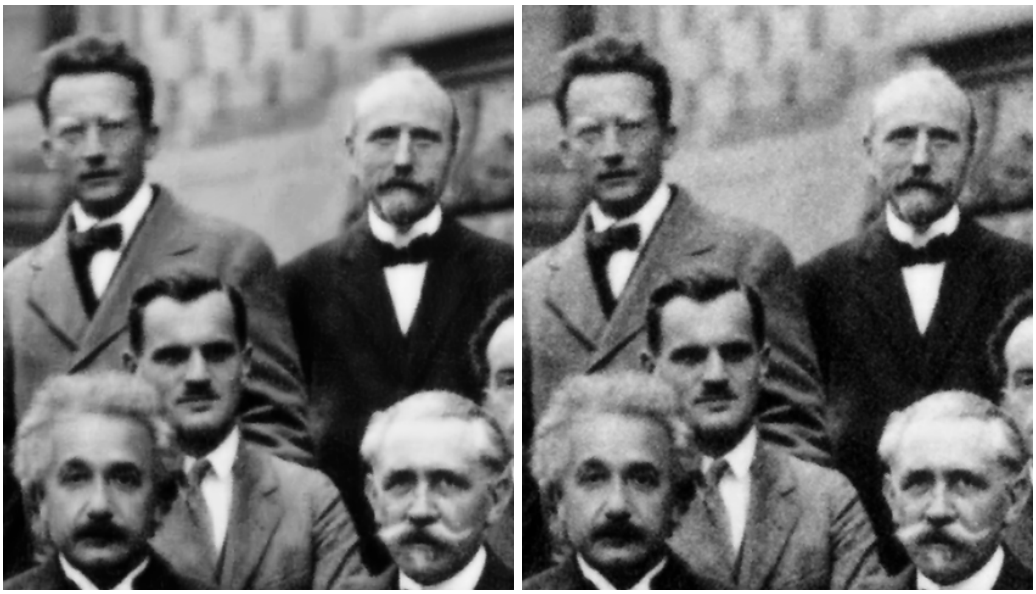


FIGURE 23. Blind denoising on "Solvay conference, 1927". From left to right: crop of the result of the Noise Clinic by using three scales and crop of the result of the Blind BLS-GSM algorithm.

Part 3

REPRODUCIBLE RESEARCH CONTRIBUTIONS

This part contains a fully detailed and reproducible account of the three main basic noise estimation methods that we extended in our work to reach more and more general noise estimation methods, culminating in the SFD noise model estimation. We present in order a general method to make any homoscedastic noise estimation method into a signal-dependent noise estimation (Chapter 9).

The three methods detailed here are:

- *The Ponomarenko et al. method (Chapter 10), which estimates the standard deviation of the noise using the high-frequency coefficients on a low quantile of the blocks in the image whose energy measured in the low-frequency coefficients is minimal.*
- *The Percentile method (Chapter 11), which estimates the standard deviation of the noise from the blocks of a high-passed version of the noisy image, using only the blocks whose standard deviation is under a small quantile.*
- *The PCA method (Chapter 12), which estimates the variance of the noise from the eigenvalues of the covariance matrix of the blocks of the noisy image.*

In Chapter 13 these three methods are evaluated in different scenarios.

CHAPTER 9

How to adapt homoscedastic noise estimators to signal-dependent noise

Most block-based homoscedastic noise estimators can be easily adapted to deal with signal-dependent noise. Even if the signal-dependent noise model by itself is not sufficient to characterize the correlated noise (see Chapters 3 and 4), yet it is useful to obtain the noise curves of raw images, where the noise is only-signal dependent. Section 1.1 explains in detail the procedure to adapt a block-based noise estimator to measure signal-dependent noise.

Once the noise curve has been obtained, it might happen that some of its control points exhibit an overestimation for some particular intensities. This happens when all blocks in a small intensity range belong to a texture. Section 1.2 proposes a filtering algorithm to minimize the overestimation in the noise curve. Another problem that block-based noise estimators are expected to find is the presence of saturated points in the image. This special points appear at extreme intensities as isolated control points that distort the noise curve when interpolation is performed in between. Section 1.3 presents a procedure to avoid taking into account completely saturated pixels in the noise estimation.

The techniques presented in this chapter are general, in the sense that they can be applied to almost any block-based noise estimator. In Chapters 10, 11, and 12, we apply them to the Ponomarenko et al., Percentile and PCA methods, namely. The articles presented in this third part were published in the Image Processing On Line (IPOL) journal. It publishes image processing and image analysis algorithms, described in accurate literary form, coupled with code. This way, scientists are allowed to check directly the published algorithms online with any uploaded image. It also promotes reproducible research, and the establishment of a state of the art verifiable by all, and on any image.

1. General techniques for adapt to signal-dependent noise estimation

1.1. Extension to signal-dependent noise. Most noise estimation methods found in the literature assume that the noise in the image is additive, signal-independent, and Gaussian. Note that in this context *uniform* means that the variance of the Gaussian noise is fixed and it does not depend on the intensity of the pixels of the ideal image. This assumption is not realistic because of the quantum nature of light itself and the way a CCD or CMOS detector responds to light. It is well-known that the emission of photons by a body follows a Poisson distribution. This distribution can be approximated by a Gaussian distribution when the number of photons is large enough. For very dark regions of the image this assumption does not hold. We consider

an image pixel $\tilde{\mathbf{U}}(x, y)$ as a Poisson variable with variance and mean $\mathbf{U}(x, y)$. The Poisson noise has therefore a standard deviation of $\sqrt{\mathbf{U}(x, y)}$. (An image is nothing but a noise whose mean would be the ideal image.) This noise adds up to a thermal noise and to an electronic noise which are approximately additive and white. On a motionless scene with constant lighting, the expected value $\mathbf{U}$ can be approximated by simply accumulating photons for a long exposure time, and then by taking the temporal average of this photon count. Any noise estimation algorithm assuming that the noise is uniform is unrealistic. Fortunately, most block-based methods are easily adapted to signal-dependent noise.

For a signal-dependent noise, a "noise curve" must be established. This noise curve associates with each image value $\mathbf{U}(x, y)$ an estimation of the standard deviation of the noise associated with this value. Thus, for each block in the image, its mean must be computed and will give an estimation of a value in $\mathbf{U}$. The measurement of the variation of the block (for example, its variance) will also be stored. The means are classified into a disjoint union of variable intervals or bins, in such a way that each interval contains a large enough number of elements. These measurements allow for the construction of a list of block variances whose corresponding means belong to the given bin. To find the number of samples/bin that minimizes the committed RMSE of the obtained noise curve compared to the ground-truth noise curve, we simulated signal-dependent noise with variance $\sigma^2 = 1 + 2U$ on a set of noise-free images (Figure 1) of 1080×808 pixels each. Then, the mean RMSE along all the images in the set was computed depending on the number of bins used. The number of bins that minimizes the RMSE depends on the method. For the Ponomarenko et al. and Percentile methods, the required number of samples/bin is 42000. Figure 2 shows the RMSE depending on the number of bins for the PCA method. The minimum for the PCA method is attained when using 5 bins. However, since the error committed when using 8 bins is not much worse than the error using 5 bins and a noise curve with 8 control points is more informative that the noise curve with 5 bins, we decided to use 8 bins for images of 1080×808 pixels. Therefore, the number of samples/bin is $\frac{1080 \times 808}{8} = 109080$. Experimental results with other noise-free natural images refined the minimum to 112000 samples/bin for the PCA method.

Therefore, it is possible to apply the generic noise estimator to each set of blocks associated with a given bin. In this way, an estimation of the noise for the intensities inside the limits of the bin is obtained. Because the set of bins is disjoint and there is no gap between bins, is it possible to deduce by interpolation a curve that relates the means of the blocks with their standard deviation, hence obtaining a signal-dependent *noise curve*. The intensity associated to each bin is given by the mean of the block at the percentile. The algorithmic description of the function building this histogram of block means can be found in Algorithm 21. This algorithm works as follows:

(1) It takes as input the number of bins that will be used ("bins" variable), the input data (the variances of the blocks, "data" variable), the associated intensities of the input data (the means of the blocks, "datal" variable) and the total number of samples ("N" variable). The algorithm stores at the variable "samples_per_bin" the integer value of N/bins. In general, samples_per_bin = 112000 samples/bin. Since the last bin contain the remaining samples, it may contain less than samples_per_bin samples.

FIGURE 1. Set of noise-free images used to determine empirically the optimal number of bins that the algorithm should use, as a function of the size of the image. Each image is 1080×808 pixels.

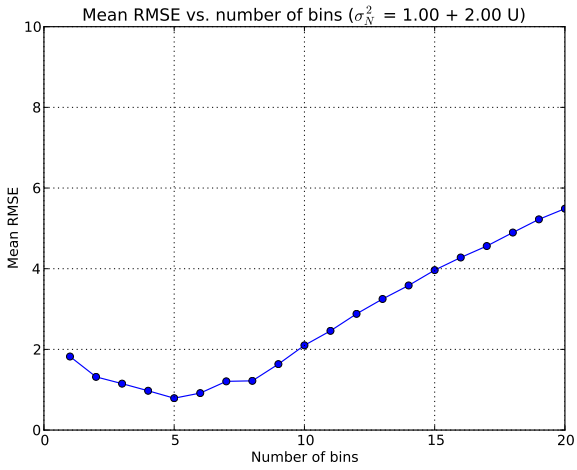


FIGURE 2. Mean RMSE of the estimation of the signal-dependent noise with variance $\sigma^2 = 1 + 2U$ along all the noise-free test images in Figure 1, for the PCA method.

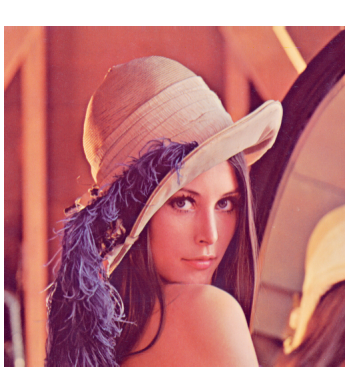
(2) It returns for each bin b its intensity bounds ("limits_begin[b]" and "limits_end[b]" variables), the list of variances that belong to bin b ("data_bins[b]" variable) and the list of intensities (block means) that belong to bin b ("datal_bins[b]" variable).
(3) For each bin b, the algorithm fills the data_bins[b] and datal_bins[b] buffers with the variances and intensities of the blocks, sorted by their mean.
(4) The lower and upper intensity bounds of the current bin b are stored into the variables limits_begin[b] and limits_end[b]. Then, the next bin is processed.

1.2. Filtering the noise curve. Optionally, the noise curve obtained on real images can be filtered. Indeed, it may present peaks when some given gray level interval contains mostly means of blocks belonging to a highly textured region. In this case, the measured block variance would be caused by the signal itself and not by the noise and the noise variance would be overestimated.

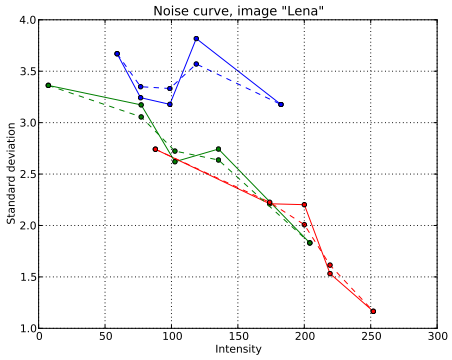
Given the i-th control point of the noise curve $(\hat{\mu}_i, \hat{\sigma}_i)$ a closed intensity interval centered at this bin is considered, that is, $[\hat{\mu}_i - D, \hat{\mu}_i + D]$. For each intensity μ inside the interval (assuming that μ starts at $\hat{\mu}_i - D$ and it is incremented with a step of 0.05 while it is less or equal to $\hat{\mu}_i + D$), it is obtained the interpolated standard deviation that corresponds to each intensity μ. In order to avoid an excessive interpolation, if $\hat{\mu}_i - D < \hat{\mu}_0$ for the i-th bin, then the diameter D is changed to the value $\hat{\mu}_b - \hat{\mu}_0$. In the same way, if $\hat{\mu}_i + D > \hat{\mu}_{B-1}$ (being B the number of bins), then the diameter D is changed to the value $\hat{\mu}_{B-1} - \hat{\mu}_b$. Since each $\hat{\mu}_i$ can be seen as an oscillation (given by the RMSE) around the ideal value, averaging the noise curve inside the interval $[\hat{\mu}_i - D, \hat{\mu}_i + D]$ for each control point attenuates the oscillations and puts them closer to the ground-truth. Once the oscillations have been attenuated, it might happen that a control point corresponds to a peak caused by a texture. In that case, the action taken is to compute the average inside the interval $[\hat{\mu}_i - D, \hat{\mu}_i + D]$ and to substitute the standard deviation $\hat{\mu}_i$ of the i-th control point by the average only if it is *lower* than the average of the intensities in the interval. The filtering procedure is iterated five times. In the first three iterations the control points are allowed to go up and down, thus canceling the oscillations around the ideal value. In the next two iterations the points are only allowed to go down, to attenuate the overestimation of the noise because of textures. The simple strategy presented here performs properly for most natural images and in general not more than five filtering iterations are needed to get a reliable estimation of the noise. Applying more than five iterations does not improve the results significantly and for certain images it could produce noise curves that are excessively smooth. A diameter $D = 7$ is recommended. The pseudo-code of the filtering is detailed in Algorithm 24. It uses Algorithm 23 to interpolate the standard deviation corresponding to a given intensity. Algorithm 23 uses Algorithm 22 to get the corresponding standard deviation by a simple affine transformation.

Figure 3 shows the noise curve for the test image *Lena*. The non-filtered curve is drawn with solid lines and the filtered curve (five iterations) with dashed lines, using $D = 7$, $p = 0.5\%$, $w = 8$ and 6 bins. Note that the peak in the blue channel has decreased.

1.3. Discarding saturated pixels. When the number of photons measured by the CCD or CMOS detector during the exposure time is too high, its output may get saturated, and therefore underestimated. When the signal saturates the output of the CCD or CMOS detector, the measured variance in the saturated areas of the image is zero. Figure 4 shows an image with some saturated pixels. If the saturated pixels are taken into account when measuring the noise, the noise curve is no more reliable. Since the intensity of the saturated pixels is much higher than the intensity of most of the pixels in the image, there is usually a large gap between the values of normal non saturated pixels and the saturated ones and the noise curve will interpolate the standard deviation values. Of course, the information given by the noise curve inside this gap is



(a)



(b)

FIGURE 3. (a): test image *Lena* used to compare the noise curves with and without filtering; (b): noise curve for *Lena*. The non-filtered curve is drawn with solid lines and the filtered curve (five iterations) with dashed lines, using $D = 7$, $p = 0.5\%$, $w = 8$ and 6 bins. Note that the peak in the blue channel is corrected after the filtering.

FIGURE 4. Image with saturated pixels.

not correct at all. In general, the strategy used to discard saturated pixels is to avoid the blocks that contain a group of four connected exactly equal pixels, in any of the channels. This is useful not only to discard saturated pixels, but also to avoid processing blocks whose pixels have suffered other types of alterations that can be detected by finding these special blocks. For example, JPEG encoding with a high compression factor sets to zero the value of the high-frequency coefficients in many of the 8×8 blocks. Among other undesired effects like lower frequency artifacts and blocking patterns, it can also create smooth zones, since high-frequency coefficients of the DCT of the block were set to zero by the JPEG encoder. In natural images that have not been highly compressed, the probability of finding a set of four connected pixels sharing exactly the same value is very low, because of noise and texture. Nevertheless, we were unable to reproduce this problem using the PCA algorithm, which does not seem to be affected by saturated pixels in raw images. However, the option is left available to the user in the demo. The pseudo-code can be found in Algorithm

25. This algorithm checks if a pixel belong to a group of 2×2 pixels whose intensity is the same (in practice, with a difference between them below $\varepsilon = 10^{-3}$). The blocks that contain at least one invalid pixels are not taken into account by the algorithm.

Algorithm 21 Algorithm classifying blocks by their means.

CLASSIFY_BY_MEAN - Splits the input elements into disjoint *bins* according to the mean of the elements trying that each bin has the same cardinality. **Input bins**: number of bins. **Input data**: list of input data elements. **Input N**: number of elements/bin. **Input datal**: list of means of the input elements. **Output limits_begin[b]**: the lower intensity bound for bin b. **Output limits_end[b]**: the upper intensity bound for bin b. **Output data_bins[b]**: list of elements at bin b. **Output datal_bins[b]**: list of means of the elements at bin b.

samples_per_bin = $\lfloor N/\text{bins} \rfloor$
limits_begin = zeros(bins)
limits_end = zeros(bins)
num_elements = zeros(bins)
data_bins = array(bins)
datal_bins = zeros(bins)
buffer = array(N)
bufferl = zeros(N)
▷ Sort data by datal
indices = argsort(datal, N)
▷ Min and max
min_datal = datal[indices[0]]
max_datal = datal[indices[N-1]]
lim0 = min_datal
elements_count = 0
bin = 0
for idx = 0 ... N **do**
    if idx == N **then**
        finished_loading = true
    else
        lim1 = datal[indices[idx]]
        finished_loading = ¬ (bin == bins - 1) ∧ (elements_count ≥ samples_per_bin)
    end if
end for
if finished_loading **then**
    data_bins[bin] ← buffer
    datal_bins[bin] ← bufferl
▷ Update limits and number of elements of the bin
    limits_begin[bin] = lim0
    limits_end[bin] = lim1
    num_elements[bin] = elements_count
▷ Prepare for the next element
    lim0 = lim1
    bin = bin + 1
    elements_count = 0
else ▷ Keep loading...
    buffer[elements_count] = data[indices[idx]]
    bufferl[elements_count] = datal[indices[idx]]
    elements_count = elements_count + 1
end if
limits_end[bins-1] = max_datal

Algorithm 22 Obtain a corresponding standard deviation by an affine transformation.

AFFINE. **Input** (μ_c, σ_c): current control point. **Input** (μ_e, σ_e): endpoint control point. **Input** μ: intensity of the control points whose standard deviation is wanted. **Output** σ: standard deviation attributed to the intensity μ.

$\varepsilon = 10^{-6}$
if $|\mu_c - \mu_e| < \varepsilon$ **then** ▷ Avoid dividing by zero
    $s = 0$
else
    $s = \frac{\sigma_c - \sigma_e}{\mu_c - \mu_e}$
end if
$\sigma = (\mu - \mu_e)s + \sigma_e$

Algorithm 23 Interpolates an affine standard deviation from of the points of the given noise curve.

INTERPOLATION. **Input** (μ_c, σ_c): known control points. **Input** μ: the intensity of the point whose interpolated standard deviation is wanted. **Output** σ: the interpolated standard deviation of the point whose intensity is μ.

▷ Find the nearest control point
$i = \text{argmin}_i\,(\mu_c[i] - \mu|)$
$m = \mu_c[i]$
if $\mu < m$ **then** ▷ on the right of μ
    if $i = 0$ **then** ▷ Treat boundary
        $i = 1$
        $m = \mu_c[i]$
    end if
    $m_1 = \mu_c[i-1]$
    $m_2 = m$
    $s_1 = \sigma_c[i-1]$
    $s_2 = \sigma_c[i]$
else ▷ on the left of μ
    $N = \text{len}(\mu_c)$
    if $i \geq N - 1$ **then** ▷ Treat boundary
        $i = N - 2$
        $m = \mu_c[i]$
    end if
    $m_1 = m$
    $m_2 = \mu_c[i+1]$
    $s_1 = \sigma_c[i]$
    $s_2 = \sigma_c[i+1]$
end if
    return AFFINE$(m_1, s_1, m_2, s_2, \mu)$

Algorithm 24 Filters a noise curve.

FILTER_CURVE Input (μ_c, σ_c): list of control points to be filtered. **Input** D: diameter. **Input** allow_up: allow the points to go up and down. Otherwise, they are only allowed to go down. **Output** σ_c^o: returned list filtered standard deviations

B = len(μ_c)
$\sigma_c^o \leftarrow \emptyset$
for $b = 0 \dots B-1$ **do**
    mu_current, std_current = $\mu_c[b], \sigma_c[b]$
    left = mu_current $- D$
    right = mu_current $+ D$
    ▷ Adjust the diameter for the points near the boundary
    if left $< \mu_c[0]$ **then**
        dist = $\mu_c[b] - \mu_c[0]$
        left = mu_current - dist
        right = mu_current + dist
    else
        if right $> \mu_c[B-1]$ **then**
            dist = $\mu_c[B-1] - \mu_c[b]$
            left = mu_current - dist
            right = mu_current + dist
        end if
    end if
    ▷ Add the interpolated control points inside the interval [left, right]
    sum_window = 0
    L = 0
    for μ = left ... right (with step $\Delta = 0.05$) **do**
        sum_window += INTERPOLATION(μ_c, σ_c, μ)
        L += 1
    end for
    std_new /= L
    if allow_up **then**
        std_filtered = std_new
    else
        std_filtered = std_new **if** std_new < std_current **else** std_current
    end if
    $\sigma_c^o \leftarrow$ std_filtered
end for
return σ_c^o

Algorithm 25 Algorithm for the detection of groups of four equal pixels.

DETECT_EQUAL_PIXELS - Creates a mask of valid pixels. **Input I**: input image. **Input** N_x: width of **I**. **Input** N_y: height of **I**. **Input** w: block side. **Input** num_channels: number of channels of **I**. **Output** mask: mask of VALID/INVALID pixels.

$\varepsilon = 10^{-3}$
for $i = 0 \dots N_x - 1$ **do**
  for $j = 0 \dots N_y - 1$ **do** ▷ Check if the pixel is not too close to the image boundary
    if $i < N_x - w + 1 \wedge j < Ny - w + 1$ **then**
      for c = 0 ... num_channels - 1 **do**
        u = **I**.get_channel(c) ▷ Look if the 2×2 block is constant
        pixel_status = (INVALID if c == 0 else mask[x,y]) ▷ Try to validate pixel
        if $|u[i,j] - u[i+1,j]| > \varepsilon \vee |u[i+1,j] - u[i,j+1]| > \varepsilon \vee |u[i,j+1] - u[i+1,j+1]| > \varepsilon$ **then**
          pixel_status = VALID
        end if
        mask[i, j] = pixel_status
      end for
    else
      mask[i, j] = INVALID
    end if
  end for
end for

CHAPTER 10

The Ponomarenko et al. method

In the article An Automatic Approach to Lossy Compression of AVIRIS Images *N.N. Ponomarenko et al. propose a new method to specifically compress AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) images. As part of the compression algorithm, a noise estimation is performed with a proposed new algorithm based on the computation of the variance of overlapping* 8×8 *blocks. The noise is estimated on the high-frequency orthonormal DCT-II coefficients of the blocks. To avoid the effect of edges and textures, the blocks are sorted according to their energy measured on a set of low-frequency coefficients. The final noise estimation is obtained by computing the median of the variances measured on the high-frequency part of the spectrum of the blocks using only those whose energy (measured on the low-frequencies) is low. A small percentile of the total set of blocks (typically the* 0.5%*) is used to select those blocks with the lower energy at the low-frequencies. Although the method measures uniform Gaussian noise, it can be easily adapted to deal with signal-dependent noise, which is realistic with the Poisson noise model obtained by a CCD or CMOS detector in a digital camera.*

1. Noise Estimation Method

1.1. Notation and Terminology. This chapter prepares the detailed description of the noise estimation algorithm given in Section 1.2 by fixing its notation and terminology.

- $\mathbf{U}$: the noiseless ideal image.
- $\tilde{\mathbf{U}}$: the discrete noisy image of $\mathbf{U}$.
- N_x, N_y: the width and height of $\tilde{\mathbf{U}}$ in pixels.
- $\tilde{\mathbf{U}}(x, y)$: the gray-level value of $\tilde{\mathbf{U}}$ at pixel (x, y), $x \in [0, N_x - 1]$ and $y \in [0, N_y - 1]$.
- $\mathbf{W}(x, y)$: a $w \times w$ pixels block in $\tilde{\mathbf{U}}$, $\mathbf{W}(x, y) = \{\tilde{\mathbf{U}}(x + i, y + j) : i \in [0, w - 1], j \in [0, w - 1], x \in [0, N_x - w + 1], y \in [0, N_y - w + 1]\}$.
- w: the side of the overlapping $w \times w$ pixels blocks $\mathbf{W}(x, y)$.
- M: the total number of overlapping blocks. $M = (N_x - w + 1)(N_y - w + 1)$.
- $\mathbf{D}_{x,y}$: the result of applying the orthonormal 2D DCT-II to a block $\mathbf{W}(x, y)$. Its coefficients are $\mathbf{D}_{x,y}(i, j)$ and the transform is defined as

$$\mathbf{D}_{x,y}(i,j) = Q_w(i)Q_w(j) \sum_{n_x=0}^{w-1} \sum_{n_y=0}^{w-1} \mathbf{W}(x + n_x, y + n_y) \cos\left[\frac{\pi}{w}\left(n_x + \frac{1}{2}\right) i\right] \cos\left[\frac{\pi}{w}\left(n_y + \frac{1}{2}\right) j\right],$$

with $x \in [0, N_x - w - 1], y \in [0, N_y - w - 1], i \in [0, w - 1], j \in [0, w - 1]$ and $Q_N(k)$ is a normalization factor

$$Q_N(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 0 \\ \sqrt{\frac{2}{N}}, & k \neq 0 \end{cases} .$$

From now on the DCT operator will refer specifically to this definition of the orthonormal 2D DCT-II.

- $\mathbf{W}_m$: a $w \times w$ pixels block in $\tilde{\mathbf{U}}$ according to its index m in a list of overlapping blocks $\mathbf{W}_m = \mathbf{W}(x, y)$ where $y = \left\lfloor \frac{m}{N_x - w + 1} \right\rfloor, x = m - y(N_x - w + 1), m \in \{0, 1, \ldots, M-1\}$.
- $\mathbf{D}_m(i, j) = \mathrm{DCT}(\mathbf{W}_m)$ where m is the index of the block.
- T: threshold used by the function $\delta(i, j)$ that labels the coefficients of the transformed blocks $\mathbf{D}_{x,y}(i, j)$ as *low-frequency* coefficients (see Section 1.2.2 for more details).

1.2. The Algorithm.

1.2.1. *Step 1: Computing the Set of Transformed Blocks* $\{\boldsymbol{D}_m(i, j)\}$. From an image $\tilde{\mathbf{U}}$ of width N_x and height N_y, corrupted with additive white Gaussian noise of variance σ^2, it is extracted a set of $M = (N_x - w + 1)(N_y - w + 1)$ (overlapping) $w \times w$ blocks $\{\mathbf{W}_m\}$, where m is the index of the block, $m \in \{0, 1, \ldots, M-1\}$. Many noise estimation algorithms **[4, 60, 6, 5]** compute local estimates of the noise variance in small blocks that are used for a final statistical estimation (median, average, percentile, ...). Unlike other methods that pre-filter the image before extracting noise variance information from the blocks **[57]**, the Ponomarenko et al. method measures the variance directly on $\mathbf{W}_m$. The DCT of each of these blocks is computed and gives the set $\{\mathbf{D}_m\}$ of transformed blocks. The DCT coefficients in each block are denoted by $\mathbf{D}_m(i, j)$ where m is the index of the block and $0 \leq i, j < w$ is the frequency pair associated to that coefficient.

1.2.2. *Step 2: Defining a Function to Label the Low/High Frequency Coefficients.* The algorithm labels coefficients of the transformed blocks as belonging to low or medium/high frequencies. A coefficient corresponds to a low frequency if and only if $\delta(i, j) = 1$. If not, it is labeled as belonging to the medium/high frequencies set, where δ is defined by

$$\delta(i, j) = \begin{cases} 1, & (i + j < T) \wedge (i + j \neq 0), \rightarrow \text{low frequencies} \\ 0, & (i + j \geq T) \vee (i + j = 0) \rightarrow \text{medium/high frequencies.} \end{cases}$$

where T is a given threshold, and $\wedge$ and $\vee$ stand for the *AND* and *OR* logical operators, namely. Note that this function does not label the mean of the block term $(i + j = 0)$ as a low-frequency.

1.2.3. *Step 3: Estimating the Block Empirical Variance only with the Low-Frequency Coefficients.* Given the set of transformed blocks $\{\mathbf{D}_m\}$ with $m = 0, 1, \ldots, M-1$ the set of (empirical) variances associated to the low-frequency coefficients of the block m is defined as

$$\mathbf{V}_m^L = \frac{1}{\theta} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} [\mathbf{D}_m(i, j)]^2 \, \delta(i, j),$$

where $\theta = \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} \delta(i, j)$ is the adequate normalization factor to get a mean.

1.2.4. *Step 4: Computing the Empirical Variance of the High-Frequency Coefficients.* The set of transformed blocks $\{\mathbf{D}_0, \ldots, \mathbf{D}_m, \ldots, \mathbf{D}_{M-1}\}$ is rewritten with respect to the corresponding value of $\mathbf{V}_m^L$ in ascending order. The block that gives the lowest low-frequency variance will be noted as $\mathbf{D}_{(0)}$, the next with the lowest low-frequency variance as $\mathbf{D}_{(1)}$ and so on. The block with the highest low-frequency variance is $\mathbf{D}_{(M-1)}$. Given the list of sorted blocks $\{\mathbf{D}_{(m)}\}$, the noise variance estimate associated with the high-frequency coefficient at (i, j) is defined by

$$\mathbf{V}^H(i,j) = \frac{1}{K} \sum_{k=0}^{K-1} \left[\mathbf{D}_{(k)}(i,j)\right]^2,$$

where $i+j \geq T$ and $K = \lfloor pM \rfloor, p < 1$ is the position of the p-quantile in the list $\{\mathbf{D}_{(m)}\}_{m\in[0,M-1]}$. Note that this empirical variance estimate is made with the list of the K transformed blocks whose empirical variance as measured in their low-frequencies is lowest. It is understood that these blocks are likely to contain only noise. Thus their high frequencies are good candidates to estimate the noise. Noise in high and low frequencies is uncorrelated and since most of the energy of the ideal image is concentrated in the low and medium frequency coefficients (because of the sparsity of most natural images), one can assume that $\mathbf{V}^H(i, j)$ gives an accurate estimation of the noise variance. However, if the image is highly textured, those high-frequency coefficients might give a variance that is explained by the textures of the image and not by the noise.

1.2.5. *Step 5: Choosing the Best K and Obtaining the Final Noise Estimate.* The final noise estimation is given by the median of the variance estimates $\mathbf{V}^H(i, j)$,

$$\hat{\sigma} := \sqrt{\text{median}_{i,j}\left(\{\mathbf{V}^H(i,j) \mid i+j \geq T\}\right)}.$$

However, the values in the list $\{\mathbf{V}^H(i, j)\}$ depend on the value of the quantile K. Ponomarenko et al. [**16**] propose to use the following adaptive strategy to find out the best value for K:

(1) Set $K = \sqrt{M}$. The original setting is $K = M/512$, because the algorithm is designed to work with AVIRIS images of size 512×677. In order to be able to use any size of image, we propose to set $K = \sqrt{M}$.
(2) Compute an upper bound of noise variance as $A = 1.3\mathbf{V}_{K/2}^L$.
(3) Determine a new $K = m_{\min}$, where $m_{\min}$ is the value of m that minimizes $|A - \mathbf{V}_m^L|$.
(4) Repeat seven times the steps 2 and 3.
(5) Set $A = A/5$.

Nevertheless, we found that fixing directly a small percentile equal to 0.5% of the set of variances gives more accurate and reliable results than the above procedure. This is the only place where our implementation differs from the original algorithm. The complete algorithmic description of the original method is summarized in algorithm 26. The modified version of the algorithm that uses a fixed percentile $p = 0.5\%$ instead of the iterations to find the value of K is given in algorithm 27.

For a review of several noise estimation methods we refer the reader to the *Secrets of image denoising cuisine* [**1**] and *Estimation of noise in images: an evaluation* [**55**] articles.

Algorithm 26 Pseudo-code for the Ponomarenko et al. noise estimation algorithm.

PONOMARENKO - Returns the standard deviation of the white Gaussian noise of the input image.

Input $\tilde{\mathbf{U}}$: noisy image.

Output $\tilde{\sigma}$: estimated standard deviation of its noise.

$w = 8$.
$T = 9$.
$N_x = \text{width}(\tilde{\mathbf{U}})$
$N_y = \text{height}(\tilde{\mathbf{U}})$
$M = (N_x - w + 1)(N_y - w + 1)$ ▷ number of (overlapping) blocks in $\tilde{\mathbf{U}}$
$\mathbf{W} \leftarrow$ all M possible $w \times w$ (overlapping) blocks in $\tilde{\mathbf{U}}$.
$\mathbf{D} \leftarrow \text{DCT}(\mathbf{W})$. ▷ 2D orthonormal DCT-II of the $w \times w$ blocks in $\mathbf{W}$
$\delta[i,j] = 1$ if $(i + j < T) \wedge (i + j \neq 0)$ else 0, $\forall (i,j) \in [0, w-1]^2$.
$\theta = \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} \delta[i,j]$.
$\mathbf{V}_m^L = \frac{1}{\theta} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} [\mathbf{D}_m(i,j)]^2 \, \delta[i,j]$
$K = \sqrt{M}$.
for $n = 1 \ldots 7$ **do**
    $A = 1.3\mathbf{V}_{K/2}^L$.
    $K = \text{argmin}_m \left(|A - \mathbf{V}_m^L| \right)$.
end for
$K = K/5$.
$\mathbf{I} = \text{sort}_m(\mathbf{V}_m^L)$. ▷ $\mathbf{I}$ contains the sorting indices
$\mathbf{V}^H[i,j] = \frac{1}{K} \sum_{k=0}^{K-1} \mathbf{D}^2_{\mathbf{I}[k]}(i,j)$ ▷ $\mathbf{V}^H[i,j]$ is defined only for those $[i,j]$ such that $i + j \geq T$.
$\hat{\sigma} = \sqrt{\text{median}_{i,j} \left(\mathbf{V}^H(i,j) \right)}$

2. Online Demo

2.1. Example: *traffic* Image. The results of this example can be reproduced by adding noise with parameters $A = 0$ and $B = 0.5$ to the *traffic* image. The rest of the parameters are the default parameters of the demo. Figure 1 shows the input noiseless image *traffic* before adding signal dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$.

Figure 2 shows the noise estimated for the three first scales of the signal-dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$ added to the *traffic* image. Because the noise was added to a noise-free image, we can compute the RMSE for the different scales S_0, S_1 and S_2, and the corresponding errors are 0.15, 0.18 and 0.16, respectively. Note that, as expected, the noise standard deviation is divided by approximately two when down-scaling the image by the same ratio.

Algorithm 27 Pseudo-code for the Ponomarenko et al. noise estimation algorithm (using a fixed percentile).

PONOMARENKO WITH FIXED PERCENTILE - Returns the standard deviation of the white Gaussian noise of the input image.

Input $\tilde{\mathbf{U}}$: noisy image.

Output $\tilde{\sigma}$: estimated standard deviation of its noise.

$w = 8$.

$T = 9$.

$p = 0.005$.

$N_x = \text{width}(\tilde{\mathbf{U}})$

$N_y = \text{height}(\tilde{\mathbf{U}})$

$M = (N_x - w + 1)(N_y - w + 1)$ ▷ number of (overlapping) blocks in $\tilde{\mathbf{U}}$

$\mathbf{W} \leftarrow$ all M possible $w \times w$ (overlapping) blocks in $\tilde{\mathbf{U}}$.

$\mathbf{D} \leftarrow \text{DCT}(\mathbf{W})$. ▷ 2D orthonormal DCT-II of the $w \times w$ blocks in $\mathbf{W}$

$\delta[i,j] = 1$ if $(i + j < T) \wedge (i + j \neq 0)$ else 0, $\forall (i,j) \in [0, w-1]^2$.

$\theta = \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} \delta[i,j]$.

$K = pM$. ▷ Get the p-quantile position in the list of variances. Typically $p = 0.005 \Rightarrow$ the 0.5%-percentile.

$\mathbf{V}_m^L = \frac{1}{\theta} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} [\mathbf{D}_m(i,j)]^2 \, \delta[i,j]$.

$\mathbf{I} = \text{sort}_m(\mathbf{V}_m^L)$. $\mathbf{I}$ contains the sorting indices.

$\mathbf{V}^H[i,j] = \frac{1}{K} \sum_{k=0}^{K-1} \mathbf{D}^2_{\mathbf{I}[k]}(i,j)$, ▷ $\mathbf{V}^H[i,j]$ is defined only for those $[i,j]$ such that $i + j \geq T$

$\hat{\sigma} = \sqrt{\text{median}_{i,j} \left(\mathbf{V}^H(i,j) \right)}$

FIGURE 1. Noise free input image *traffic* before adding noise with variance $\sigma^2 = 0.5\mathbf{U}$.

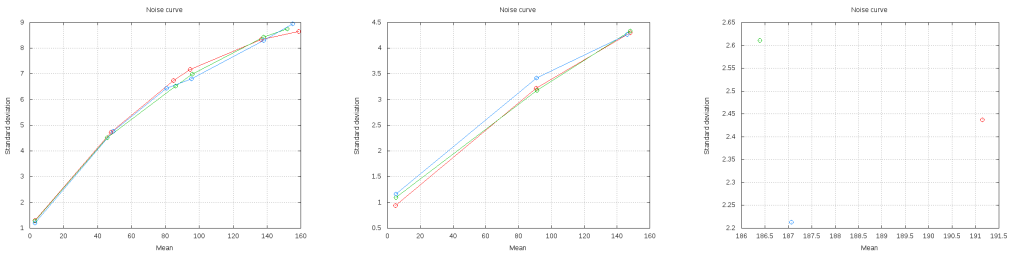


FIGURE 2. The noise estimated with the IPOL Ponomarenko et al. demo, for the three first scales of the signal-dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$ added to the *traffic* image. From left to right: scales S_0 (original), S_1 and S_2. Note that the noise standard deviation is approximately divided by two when down-scaling.

CHAPTER 11

The Percentile method

Given a white Gaussian noise signal $\boldsymbol{N}_\sigma$ on a sampling grid, its variance σ^2 can be estimated from a small $w \times w$ pixels sample. However, in natural images we observe $\tilde{\boldsymbol{U}} = \boldsymbol{U} + \boldsymbol{N}_\sigma$, the combination of the geometry of the scene that is photographed and the added noise. In this case, estimating directly the standard deviation of the noise from $w \times w$ samples of $\tilde{\boldsymbol{U}}$ is not reliable since the measured standard deviation is not explained just by the noise but also from the geometry of $\boldsymbol{U}$. The Percentile method tries to estimate the standard deviation σ from $w \times w$ blocks of a high-passed version of $\tilde{\boldsymbol{U}}$ by a small p-percentile of these standard deviations. The idea behind is that edges and textures in a block of the image increase the observed standard deviation but they never make it decrease. Therefore, a small percentile (0.5%, for example) in the list of standard deviations of the blocks is less likely to be affected by the edges and textures than a higher percentile (50%, for example). The 0.5%-percentile is empirically proven to be adequate for most natural, medical, and microscopy images. The Percentile method is adapted to deal with signal-dependent noise, which is realistic with the Poisson noise model obtained by a CCD or CMOS detector in a digital camera.

1. Introduction

The Percentile method [**1**] assumes the signal-dependent noise model. The output of the Percentile method is a noise curve, that is, a function that relates the intensity of the image with a noise standard deviation. The Percentile method tries to estimate the standard deviation σ from $w \times w$ blocks of a high-passed version of $\tilde{\mathbf{U}}$ by a small p-percentile of these standard deviations. The idea behind is that edges and textures in a block of the image increase the observed standard deviation but they never make it decrease. Therefore, a small percentile (0.5%, for example) in the list of standard deviations of the blocks is less likely to be affected by the edges and textures than a higher percentile (50%, for example). However, it might happen that for a certain intensity interval all the samples belong to textures and edges. In that case, the variance measured is explained by the geometry of the image and not the by noise. In order to minimize that effect, the noise curves can be filtered as explained in Section 1.2 of Chapter 9.

For a review of several noise estimation methods we refer the reader to the work of Lebrun et al. [**1**] and Olsen [**55**].

2. Noise Estimation Method

2.1. Notation and Terminology. This chapter prepares the detailed description of the noise estimation algorithm given in Chapter 2.2 by fixing its notation and terminology.

- $\mathbf{U}$: the noiseless ideal image.
- $\tilde{\mathbf{U}}$: the noisy input image.
- N_x, N_y: the size of $\tilde{\mathbf{U}}$. N_x and N_y are always odd. If they are not, the leftmost column or the bottom row are first removed from $\tilde{\mathbf{U}}$.
- $\tilde{\mathbf{U}}_c$: the discrete noisy image $\tilde{\mathbf{U}}$ after cropping $(s-1)/2$ columns and rows from each of the four sides of $\tilde{\mathbf{U}}$ using the function $\mathrm{CROP}_{(s-1)/2}$, where s is odd. The details of this function are given in algorithm 28. The size of $\tilde{\mathbf{U}}_c$ is therefore $(N_x-(s-1))\times(N_y-(s-1))=(N_x-s+1)\times(N_y-s+1)$.
- $\mathbf{R}$: the operator used to obtain the discrete filter $\mathbf{F}$ with support $s\times s$, with s odd.
- $\mathbf{F}$: the discrete filter with support $s\times s$ used to high-pass the noisy image, obtained from operator $\mathbf{R}$.
- $\mathbf{F}(x,y)$: the value of the discrete filter $\mathbf{F}$ at position $(x,y), x\in[0,s-1], y\in[0,s-1]$.
- $\wedge$: logical conjunction (*and* operator).
- $\vee$: logical disjunction (*or* operator).
- $\neg$: logical negation (*not* operator).
- $\tilde{\mathbf{U}^f}$: the cropped high-passed version of $\tilde{\mathbf{U}}$: $\tilde{\mathbf{U}^f} := \mathrm{CROP}_{s-1}\left[(\tilde{\mathbf{U}}*\mathbf{F})\right]$, where $*$ is the discrete convolution operator:

$$\tilde{\mathbf{U}^f}(x,y) := \mathrm{CROP}_{s-1}\left[(\tilde{\mathbf{U}}*\mathbf{F})(x,y)\right] = \mathrm{CROP}_{s-1}\left[\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\mathbf{F}_z(i,j)\tilde{\mathbf{U}}_z(x-i,y-j)\right] =$$

$$= \mathrm{CROP}_{s-1}\left[\sum_{i=0}^{s-1}\sum_{j=0}^{s-1}\mathbf{F}(i,j)\tilde{\mathbf{U}}_z(x-i,y-j)\right], \tag{30}$$

where $\mathbf{F}_z(x,y) := \begin{cases}\mathbf{F}(x,y), & x\in[0,s-1]\wedge y\in[0,s-1]\\ 0 \text{ otherwise}\end{cases}$ extends $\mathbf{F}$ with zeros,

$\tilde{\mathbf{U}}_z(x,y) := \begin{cases}\tilde{\mathbf{U}}(x,y), & x\in[0,N_x-1]\wedge y\in[0,N_y-1]\\ 0 \text{ otherwise}\end{cases}$ extends $\tilde{\mathbf{U}}$ with zeros and the CROP_{s-1} function removes $s-1$ columns or rows at each of the four boundaries of the filtered image to avoid boundary effects. Since the convolution $(\tilde{\mathbf{U}}*\mathbf{F})(x,y)$ is defined for $x\in[0,N_x+s-2]\wedge y\in[0,N_x+s-2]$, the cropped image $\tilde{\mathbf{U}^f}(x,y)$ is defined for $x\in[0,N_x-s]\wedge y\in[0,N_y-s]$. The Fast Fourier Transform (FFT) algorithm **[123]** is used to speed-up the computation of the convolution; the details are given in Chapter 2.2.1.

- $\tilde{\mathbf{U}}_c(x,y)$: the gray-level intensity of the pixel of $\tilde{\mathbf{U}}_c$ at column x and row y, $x\in[0,N_x-s]$, $y\in[0,N_y-s]$.

- $\tilde{\mathbf{U}^f}(x,y)$: the gray-level intensity of the pixel of $\tilde{\mathbf{U}^f}$ at column x and row y, $x \in [0, N_x - s]$, $y \in [0, N_y - s]$.
- w: the side of the overlapping $w \times w$ pixels blocks $\mathbf{W}(x,y)$.
- M: the total number of overlapping blocks. $M = (N_x - w - s + 1)(N_y - w - s + 1)$.
- $\mathbf{W}(x,y)$: a $w \times w$ pixels block in $\tilde{\mathbf{U}}_c$, $\mathbf{W}(x,y) = \{\tilde{\mathbf{U}}_c(x+i, y+j) : i \in [0, w-1], j \in [0, w-1]\}$.
- $\mathbf{W}^f(x,y)$: a $w \times w$ pixels block in $\tilde{\mathbf{U}^f}$, $\mathbf{W}^f(x,y) = \{\tilde{\mathbf{U}^f}(x+i, y+j) : i \in [0, w-1], j \in [0, w-1]\}$.
- $\mathbf{W}_m$: a $w \times w$ pixels block in $\tilde{\mathbf{U}}_c$ according to its index m in a list of overlapping blocks $\mathbf{W}_m = \mathbf{W}(x,y)$ where $y = \left\lfloor \frac{m}{N_x - s + 1} \right\rfloor, x = m - y(N_x - s + 1), m \in [0, M-1]$.
- $\mathbf{W}^f_m$: a $w \times w$ pixels block in $\tilde{\mathbf{U}^f}$ according to its index m in a list of overlapping blocks $\mathbf{W}^f_m = \mathbf{W}^f(x,y)$ where $y = \left\lfloor \frac{m}{N_x - s + 1} \right\rfloor, x = m - y(N_x - s + 1), m \in [0, M-1]$.
- $\mathbf{V}$: the list of M variances from the blocks $\{\mathbf{W}^f_m\}$. $\mathbf{V}[m]$ corresponds to the variance of the block $\mathbf{W}^f_m$.
- $\mathbf{T}$: the list of M means from the blocks $\{\mathbf{W}_m\}$. $\mathbf{T}[m]$ corresponds to the mean of the block $\mathbf{W}_m$.
- $\mathbf{IV}n$: the index of the element at the position n in the list $\mathbf{V}$ after sorting the elements of $\mathbf{V}$ in ascending order.
- p: the (small) p-percentile.
- $\hat{\sigma}^2$: biased variance at the p-percentile of $\mathbf{V}$.
- $\tilde{\sigma}^2$: final unbiased variance at the p-percentile of $\mathbf{V}$.

2.2. The Algorithm.

2.2.1. *Step 1: Pre-filtering the Input Noisy Image.* In order to get rid of deterministic tendencies due to signal structure, the image is first pre-filtered with a high-pass filter $\mathbf{F}$ implemented as a discrete stencil with support $s \times s$. This stencil corresponds to a discretization of a certain operator $\mathbf{R}$, typically a differential operator or a waveform. Convolving the image with such a filter removes smooth variations inside the blocks, which increases the number of blocks where noise dominates and on which the variance estimate will be reliable. Mastis proposed a similar approach **[57]**, where operator $\mathbf{F}$ writes as a simple subtraction of the average or median to each 7×7 block. For the Percentile method a filter based on the DCT with support 7×7 is proposed. Given an $s \times s$ block in the image $\tilde{\mathbf{U}}$ at position (x,y), its orthonormal 2D DCT-II is

(31)
$$\mathrm{DCT}\left(\tilde{\mathbf{U}}(x,y)\right)(i,j) := Q_s(i)Q_s(j) \sum_{n_x=0}^{s-1} \sum_{n_y=0}^{s-1} \tilde{\mathbf{U}}(x+n_x, y+n_y) \cos\left[\frac{\pi}{s}\left(n_x + \frac{1}{2}\right) i\right] \cos\left[\frac{\pi}{s}\left(n_y + \frac{1}{2}\right) j\right]$$

with $x \in [0, N_x - s - 1], y \in [0, N_y - s - 1], i \in [0, s-1], j \in [0, s-1]$ and $Q_N(k)$ is the normalization factor

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & -8 & 2 & 0 \\ 1 & -8 & 20 & -8 & 1 \\ 0 & 2 & -8 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 3 & -12 & 3 & 0 & 0 \\ 0 & 3 & -24 & 57 & -24 & 3 & 0 \\ 1 & -12 & 57 & -112 & 57 & -12 & 1 \\ 0 & 3 & -24 & 57 & -24 & 3 & 0 \\ 0 & 0 & 3 & -12 & 3 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

FIGURE 1. From left to right, the discrete stencils associated Δ, $\Delta\Delta$ and $\Delta\Delta\Delta$ discrete operators, respectively.

$$Q_N(k) := \begin{cases} \frac{1}{\sqrt{N}}, & k = 0 \\ \sqrt{\frac{2}{N}}, & k \neq 0. \end{cases} \tag{32}$$

Filter $\mathbf{F}$ is made by taking the normalized product of cosines that correspond to the highest frequency $[s-1, s-1]$, that is,

$$\mathbf{F}(n_x, n_y) := \frac{2}{s} \cos\left[\pi \left(n_x + \frac{1}{2}\right) \frac{s-1}{s}\right] \cos\left[\pi \left(n_y + \frac{1}{2}\right) \frac{s-1}{s}\right], \ (n_x, n_y) \in [0, s-1]^2. \tag{33}$$

The filter $\mathbf{F}$ presented here was empirically proven to give the best results. However, other typical differential operators can be used, like directional derivatives, the Δ (Laplace) operator, or its iterations $\Delta\Delta$, $\Delta\Delta\Delta$, all implemented as discrete stencils. Figure 1 shows the discrete stencils associated to these operators.

The filtered image $\tilde{\mathbf{U}^f}$ is obtained by cropping the discrete convolution, $\text{CROP}_{s-1}\left[(\tilde{\mathbf{U}} * \mathbf{F})(x, y)\right]$. Note that this cropping operation avoids the boundary effects of the convolution. To speed-up the computation, the Fast Fourier Transform (FFT) algorithm is used:

(1) Consider the signal

$$\tilde{\mathbf{U}}_z(x, y) := \begin{cases} \tilde{\mathbf{U}}(x, y), \ x \in [0, N_x - s] \wedge y \in [0, N_y - s] \\ 0 \text{ otherwise.} \end{cases} \tag{34}$$

$\tilde{\mathbf{U}}_z(x, y)$ is defined for $x \in [0, N_x + s - 2] \wedge y \in [N_y + s - 2]$.

(2) Consider the signal

$$\mathbf{F}_z(x, y) := \begin{cases} \mathbf{F}(x, y), \ (x, y) \in [0, s-1]^2 \\ 0 \text{ otherwise.} \end{cases} \tag{35}$$

$\mathbf{F}_z(x, y)$ is defined for $x \in [0, N_x + s - 2] \wedge y \in [N_y + s - 2]$.

(3) Compute the FFT of $\tilde{\mathbf{U}}_z(x, y)$: $\text{FFT}[\tilde{\mathbf{U}}_z(x, y)]$.

(4) Compute the FFT of $\mathbf{F}_z(x, y)$: $\text{FFT}[\mathbf{F}_z(x, y)]$.

(5) Compute the point-wise product of the FFTs:

$$\text{FFT}[\tilde{\mathbf{U}}_z(x, y)] \times \text{FFT}[\mathbf{F}_z(x, y)].$$

(6) Compute the inverse FFT of the point-wise product:

$$\text{FFT}^{-1}(\text{FFT}[\tilde{\mathbf{U}}_z(x,y)] \times \text{FFT}[\mathbf{F}_z(x,y)]).$$

(7) Crop (see algorithm 28) the result to get $\tilde{\mathbf{U}^f}$, the cropped and low-pass filtered version of the noisy input image $\tilde{\mathbf{U}}$:

$$\tilde{\mathbf{U}^f}(x,y) := \text{CROP}_{s-1}\left[\text{FFT}^{-1}\left(\text{FFT}[\tilde{\mathbf{U}}_z(x,y)] \times \text{FFT}[\mathbf{F}_z(x,y)]\right)\right]. \tag{36}$$

Algorithm 28 Crops the boundary of the input image.

CROP$_b$ - Crops the boundary of the input image.
Input I: input image.
Input N_x: width of **I**.
Input N_y: height of **I**.
Input b: width of the boundary that will be removed at each of the four sides of **I**.
Output V: the cropped image
$\mathbf{V} = \text{zeros}(N_x - 2b, N_y - 2b)$
for $y = b \dots N_y - b - 1$ **do**
  for $x = b \dots N_x - b - 1$ **do**
    $\mathbf{V}[x-b, y-b] = \mathbf{I}[x,y]$
  end for
end for

2.2.2. *Step 2: Computing the Sample Variances from $\tilde{\boldsymbol{U}^f}$ and the Means from $\tilde{\boldsymbol{U}}$.* The size of the filtered noisy image $\tilde{\mathbf{U}^f}$ is $N_x - s + 1 \times N_y - s + 1$ pixels. Therefore, there are $M = (N_x - w - s + 1)(N_y - w - s + 1)$ overlapping blocks of size $w \times w$ pixels. Each overlapping block is referred to as $\{\mathbf{W}_m^f\}$, where m is the index of the block, $m \in \{0, 1, \dots, M-1\}$. Many noise estimation algorithms [**4, 60, 6, 5**] compute local estimates of the noise variance in small blocks that are used for a final statistical estimation (median, average, percentile, ...).

The empirical variance of each block $\mathbf{W}_m^f$ is computed as

$$\text{Var}\mathbf{W}_m^f := \frac{1}{w^2 - 1} \sum_{x=0}^{w-1} \sum_{y=0}^{w-1} [\mathbf{W}_m^f(x,y) - \bar{\mathbf{W}_m^f}]^2 \tag{37}$$

where $\bar{\mathbf{W}_m^f} = \frac{1}{w^2} \sum_{x=0}^{w-1} \sum_{y=0}^{w-1} \mathbf{W}_m^f(x,y)$ is the mean of the block. Let $\mathbf{V}$ be the list of variances of the blocks $\{\mathbf{W}_m^f\}$, $\mathbf{V}[m] := \text{Var}\mathbf{W}_m^f$. The corresponding means from $\{\mathbf{W}_m\}$ are stored in the list $\mathbf{T}[m] = \bar{\mathbf{W}}_m = \frac{1}{w^2} \sum_{x=0}^{w-1} \sum_{y=0}^{w-1} \mathbf{W}_m(x,y)$, $m \in \{0, 1, \dots, M-1\}$. The mean of each patch will be needed when extending the method to signal-dependent noise (Section 1.1 of Chapter 9), and therefore is stored at this stage of the method.

2.2.3. *Step 3: Obtaining a (biased) Noise Variance Estimation from $\boldsymbol{V}$ Using p.* Once the list $\mathbf{V}$ is built a biased noise variance estimation is obtained by the p-percentile. Set $\mathbf{I}Vn := \text{SORTED}\,(\mathbf{V})\,[n]$ as the function that given a list of real numbers, sorts them in ascending order and returns the sorting indices. For example, if $n = 0$ then $\mathbf{I}V0$ is the index of the minimum in

FIGURE 2. Left: Noise-free test image *computer* (see Section 1 of Chapter 13) after adding homoscedastic white Gaussian noise of $\sigma = 10$. Right: pure homoscedastic white Gaussian noise of $\sigma = 10$ image. Both images have the same 704×469 size.

the list $\mathbf{V}$ and therefore $\mathbf{V}[\mathbf{I}V0]$ is that minimum. In this step, a biased estimation of the variance is obtained by the small p-percentile and is given by

$$\hat{\sigma}^2 = \mathbf{V}\left[\mathbf{I}V\left\lfloor \frac{p}{100}M + \frac{1}{2}\right\rfloor\right], \tag{38}$$

where M is the cardinal of $\mathbf{V}$. We obtain a list of variances $\mathbf{I}Vn$ of $\mathbf{V}$ sorted in ascending order, $\mathbf{V}[\mathbf{I}Vn]$ with $n \in [0, M-1]$. Since in general the variance of the signal (geometry of the image) is higher than the variance of the noise, small percentiles of $\mathbf{V}$ are related more to the noise than to the signal. To illustrate it, figure 2 shows the noise-free test image *computer* after adding homoscedastic white Gaussian noise of $\sigma = 10$, and an image of pure homoscedastic Gaussian noise of $\sigma = 10$. Figure 3 shows the values of $\mathbf{V}[\mathbf{I}Vn]$ depending on n and using 21×21 blocks without any filtering. We refer to Section 1 of Chapter 13 for the details about how the noise-free images are obtained. Only for small percentiles the estimation of the variance on the *computer* image is close to the estimation on pure noise because of the effect of the image edges and textures. The Percentile method tries to avoid the effect of edges and textures by considering the variances under a very low percentile of the block variance histogram.

2.2.4. *Step 4: Correcting the Biased Estimation $\hat{\sigma}^2$ to Obtain the Final $\tilde{\sigma}^2$ Estimation.* When a percentile different from the median ($p = 0.5$) is used, the estimation of the variance obtained is biased by the percentile. Figure 4 shows the values of $\mathbf{V}[\mathbf{I}Vn]$ depending on n using 21×21 blocks without any filtering in the image of pure white Gaussian noise (figure 2, right). If the percentile is under the median of the distribution an underestimation of the variance of the noise is obtained; on the other hand, if it is over the median, the result is an overestimation. The median is attained at position $n = \frac{(N_x - w + 1)(N_y - w + 1)}{2} = \frac{(704-21+1)(469-21+1)}{2} = 153558$. Since only a small percentile gives a reliable biased estimation of the noise (see Chapter 2.2.3), the estimation is lower than the real average block variance and it has to be corrected in order to get an estimation close to the median. The correction consists of multiplying the biased estimation $\hat{\sigma}$ by a factor. This correction only depends on the percentile, block size and on the chosen operator used to filter the image. Figure 5 shows the correction $\mathbf{C}_{w,\mathbf{R}}(p) = (\hat{\sigma} - \tilde{\sigma})_{w,\mathbf{R}}$ vs. the direct estimation $\hat{\sigma}$ learned

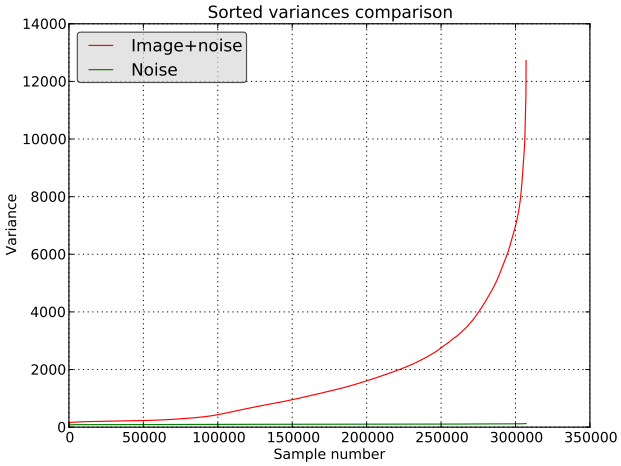


FIGURE 3. Values of $\mathbf{V}[\mathbf{I}Vn]$ depending on n using 21×21 blocks without any filtering in the noise-free test image *computer* after adding homoscedastic white Gaussian noise of $\sigma = 10$ (red) and for pure homoscedastic white Gaussian noise of $\sigma = 10$ image (green). Only a part of the values is shown. The estimation in the *computer* image is only reliable when the percentile of $\mathbf{V}$ is small.

with pure-noise images. The correction is linear with the observed $\hat{\sigma}$. As a matter of fact it can be easily proven that there exists a constant $k_{w,\mathbf{R}}$ such that $\hat{\sigma} = k_{w,\mathbf{R}}\mathbf{C}_{w,\mathbf{R}}(p) = k_{w,\mathbf{R}}(\hat{\sigma} - \tilde{\sigma})$ and then $(k_{w,\mathbf{R}} - 1)\hat{\sigma} = k_{w,\mathbf{R}}\tilde{\sigma}$. As a consequence,

$$\tilde{\sigma} = \frac{k_{w,\mathbf{R}} - 1}{k_{w,\mathbf{R}}}\hat{\sigma}. \tag{39}$$

Nevertheless, this constant $k_{w,\mathbf{R}}$ is not easy to calculate explicitly, but can be learned from simulations. To obtain it, a large image of 4320×3232 pixels with all pixels set to zero is used. Homoscedastic Gaussian noise with standard deviation σ is simulated and added to this image. Then, the noise is estimated from the noisy image using 200 bins, with a percentile $p \in \{0.01\%, 0.1\%, 0.5\%, 5\%, 10\%, 50\%\}$, a pre-filter operator $\mathbf{R}$, which can be chosen between the following: Identity (no filtering), Directional derivative, Laplace, Laplace (2 iterations), Laplace (3 iterations), Laplace (4 iterations), DCT with support 7×7, DCT with support 5×5, DCT with support 3×3 or the filter of the article Fast Noise Variance Estimation [**4**]. The size of the block is $w \times w$ with $w \in \{3, 7, 8, 21\}$. No curve filtering iterations are used. The averaged estimation along all the bins gives $\hat{\sigma}$.

The Fast Noise Variance Estimation method tries to avoid the influence of image structures (edges and textures) on the image when estimating the noise. To do it, it detects these structures using an operator based on the Laplacian and cancels them. It therefore considers two 3×3 Laplacian stencils L_1 and L_2 and computes their difference to obtain the noise estimation operator $L = 2(L_2 - L_1)$.

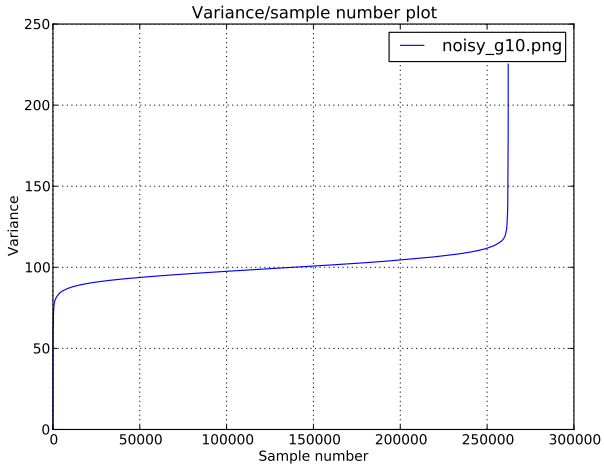


FIGURE 4. values of $\mathbf{V}[\mathbf{I}Vn]$ depending on n using 21×21 blocks without any filtering in the image of pure white Gaussian noise (figure 2, right). If the percentile is under the median one obtains an underestimated value; if it is above the median, an overestimation. The median is attained at position $n = \frac{(N_x - w + 1)(N_y - w + 1)}{2} = \frac{(704 - 21 + 1)(469 - 21 + 1)}{2} = 153558$.

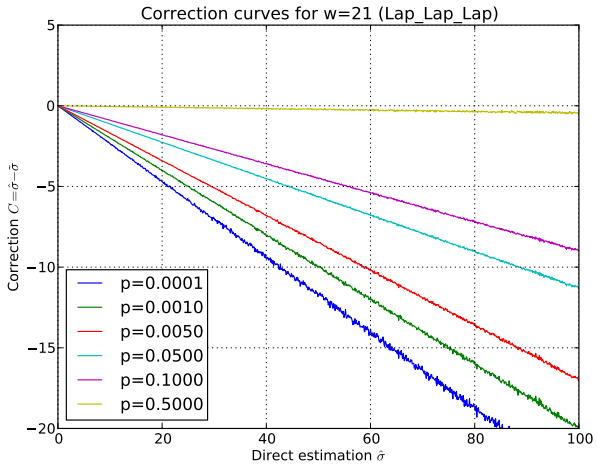


FIGURE 5. Corrections $\mathbf{C}_{w,\mathbf{R}}(p) = \hat{\sigma} - \tilde{\sigma}$ for several different percentiles, with 21×21 blocks and the $\mathbf{R} = \Delta\Delta\Delta$ operator, learnt on pure-noise patches.

For example, with $p = 0.5\%$, $w = 21$ and $\mathbf{R} = \Delta\Delta\Delta$, this empirical $\frac{k_{w,\mathbf{R}}}{k_{w,\mathbf{R}}-1}$ factor learned on pure noise is 1.208610869.

The complete algorithmic description of the Percentile method is summarized in algorithm 29.

Algorithm 29 Percentile noise estimation algorithm.

PERCENTILE - Returns the standard deviation of the image noise.
Input $\tilde{\mathbf{U}}$: discrete noisy image $\mathbf{U}$ after cropping.
Input N_x: (odd) width of the image before cropping.
Input N_y: (odd) height of the image before cropping.
Input $\mathbf{F}$: discrete filter with support $s \times s$.
Input s: support parameter of the discrete filter $\mathbf{F}$.
Input w: side of the $w \times w$ pixels blocks.
Output $\tilde{\sigma}$: estimated standard deviation of the image noise.
$\tilde{\mathbf{U}^f} = \mathrm{CROP}_{s-1}\left[\tilde{\mathbf{U}} * \mathbf{F}\right]$. ▷ Filter $\tilde{\mathbf{U}}$ with the discrete filter $\mathbf{F}$ according to formula (2.1).
$M = (N_x - w - s + 1)(N_y - w - s + 1)$. ▷ Number of overlapping blocks.
for $m = 0 \dots M - 1$ **do**
    $\mathbf{W}^f(x, y) = \{\tilde{\mathbf{U}^f}(x + i, y + j) : i \in [x, x + w - 1], j \in [y, y + w - 1]\}$
    $\mathbf{W}^f_m = \mathbf{W}^f(x, y)$ where $y = \left\lfloor \frac{m}{N_x - s + 1} \right\rfloor, x = m - y(N_x - s + 1), m \in [0, M - 1]$.
    $\mathbf{V}[m] = \mathrm{Var}\mathbf{W}^f_m = \frac{1}{s^2 - 1} \sum_{x=0}^{s-1} \sum_{y=0}^{s-1} [\mathbf{W}^f_m(x, y) - \bar{\mathbf{W}^f_m}]^2$. ▷ Compute sample variances of the blocks.
end for
$\mathbf{I}Vn = \mathrm{SORTED}\,(\mathbf{V})\,(n)\ \forall n \in [0, M - 1]$. ▷ Get ascending sorting indices.
$\hat{\sigma}^2 = \mathbf{V}\left[\mathbf{I}V\left\lfloor \frac{p}{100} M + \frac{1}{2} \right\rfloor\right]$. ▷ Get biased variance estimation.
$\hat{\sigma} = \sqrt{\hat{\sigma}^2}$. ▷ Get biased standard deviation.
$\tilde{\sigma} = \frac{k_{w,\mathbf{R}} - 1}{k_{w,\mathbf{R}}} \hat{\sigma}$. ▷ Obtain the final unbiased estimation by correcting $\hat{\sigma}$.

3. Optimal Parameters

The optimal parameter choice depends on the size of the image. Three possible sizes were fixed: $S_0 = 6M$, $S_1 = \frac{S_0}{4} = \frac{6M}{4}$, $S_2 = \frac{S_1}{4} = \frac{6M}{16}$, $S_3 = \frac{S_2}{4} = \frac{6M}{32}$ and $S_4 = \frac{S_3}{4} = \frac{6M}{128}$, where M stands for megapixels. Table 1 shows the choice of the parameters values according to the size of the image.

Image size	Percentile p	Block size $w \times w$	Pre-filter operator $\mathbf{R}$
S_0	0.005	21×21	DCT supp. 7×7
S_1	0.005	15×15	DCT supp. 7×7
S_2	0.005	15×15	DCT supp. 7×7
S_3	0.005	15×15	DCT supp. 7×7
S_4	0.005	5×5	$\Delta\Delta\Delta$

TABLE 1. Best percentile p, block size $w \times w$, and pre-filter operator $\mathbf{R}$ for the Percentile method according to the size of the image.

4. Online demo

4.1. Example: *traffic* Image. The results of this example can be reproduced by adding noise with parameters $A = 0$ and $B = 0.5$ to the *traffic* image. The rest of the parameters are

the default parameters of the demo. Figure 1 shows the input noiseless image *traffic* before adding signal dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$.

Figure 6 shows the noise estimated for the three first scales of the signal-dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$ added to the *traffic* image. Because the noise was added to a noise-free image, we can compute the RMSE for the different scales S_0, S_1 and S_2, and the corresponding errors are 0.15, 0.18 and 0.16, respectively. Note that, as expected, the noise standard deviation is divided by approximately two when down-scaling the image by the same ratio.

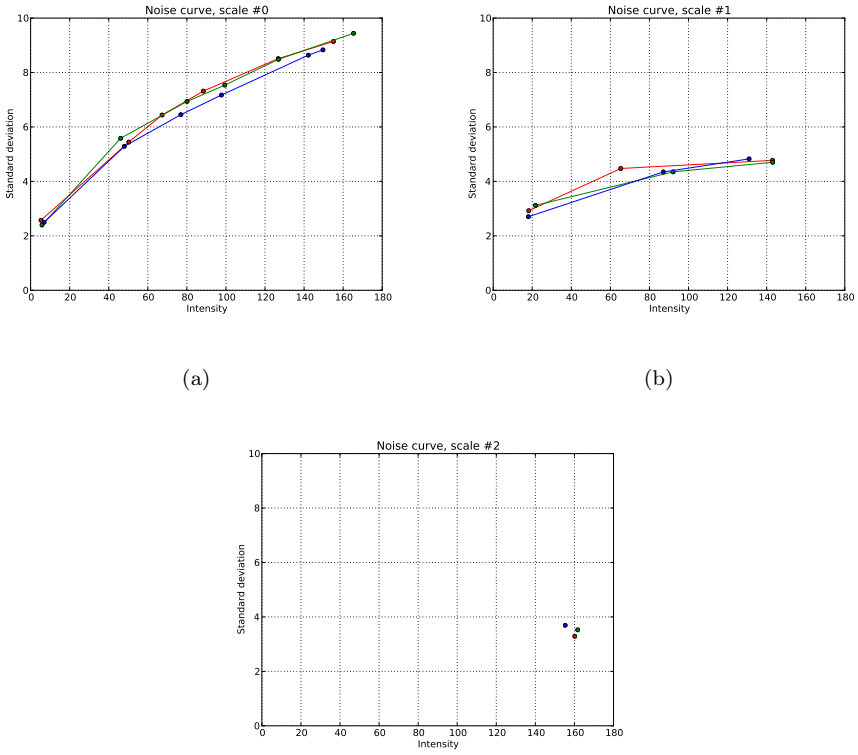


FIGURE 6. The noise estimated with the IPOL Percentile demo, for the three first scales of the signal-dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$ added to the *traffic* image. From left to right: scales S_0 (original), S_1 and S_2.

CHAPTER 12

The PCA method

In the article Image Noise Level Estimation by Principal Component Analysis **[3]**, *S. Pyatykh, J. Hesser, and L. Zheng propose a new method to estimate the variance of the noise in an image from the eigenvalues of the covariance matrix of the overlapping blocks of the noisy image. Instead of using all patches of the noisy image, the authors propose an iterative strategy to adaptively chose the optimal set containing the patches with lowest variance. Although the method measures uniform Gaussian noise, it can be easily adapted to deal with signal-dependent noise, which is realistic with the Poisson noise model obtained by a CCD or CMOS detector in a digital camera.*

1. Noise estimation method

Noise estimation is a necessary preliminary step for most image processing and computer vision algorithms. Most noise estimation algorithms start by applying a high-pass filter to the image or image patches. The energies of these high-passed image patches are used in order to estimate the noise standard deviation by some statistical criterion. Instead of using any fixed high-pass filter, S. Pyatykh, J. Hesser, and L. Zheng **[3]** decided to apply Principal Component Analysis in order to select an adapted representation for image patches. The energy of the patches with fewer variation are used in order to estimate the noise variance since they are supposed to be less affected by edges and textures **[124, 59]**. Computing a local estimate of the noise variance from small blocks than then infering the variance of the noise by applying some statistic (median, average, percentile, ...) is technique shared by most noise estimation methods **[4, 60, 6, 5]**. Unlike other methods that pre-filter the image before extracting noise variance information from the blocks **[57]**, the PCA method measures the variances directly from the noisy patches. For a review of several noise estimation methods we refer the reader to the "Secrets of image denoising cuisine" **[1]** and "Estimation of noise in images: an evaluation" **[55]** articles.

The algorithm proposed in **[3]** applies only for the estimation of white Gaussian noise. However, this assumptions is not realistic because of the way a CCD or a CMOS detector responds to light. The photon emission by a body follows a Poisson distribution which can be approximated by a Gaussian distribution when the number of photons is large enough. But the variance of this Gaussian is signal-dependent. For this reason, the PCA noise estimation algorithm is adapted (see Section 1 of Chapter 9) in order to estimate signal-dependent noise and get a noise curve that assigns an standard deviation of the noise for each gray level intensity of the input image.

1.1. Notation and terminology. This chapter prepares the detailed description of the noise estimation algorithm given in Section 3 by fixing its notation and terminology.

- $\mathbf{x}$: the discrete noise-free image of size $S_1 \times S_2$ pixels.
- $\mathbf{n}$: a discrete additive Gaussian noise of variance σ^2 and zero mean of size $S_1 \times S_2$ pixels.
- $\mathbf{y} = \mathbf{x} + \mathbf{n}$: the corrupted image resulting from adding the noise $\mathbf{n}$ to the noise-free image $\mathbf{x}$.
- $M = M_1 \times M_2$: the size in pixels of the overlapping blocks; $M_1 = M_2 = 5 \Rightarrow M = 25$.
- $Q(p)$: the p-quantile of the list of the variances of the overlapping blocks in $\mathbf{y}$.
- $N = (S_1 - M_1 + 1) \times (S_2 - M_2 + 1)$: the number of overlapping blocks.
- $\mathbf{X}$: matrix of samples of a random vector made from realizations $\mathbf{x}_i$, $i \in [1, N]$. It can be written in matrix notation as

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1M} \\ x_{21} & x_{22} & \dots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{NM} \end{pmatrix}$$

  where each row of the matrix contains a different blocks, that is, a realization $\mathbf{x}_i$ with $i \in [0, N]$.
- $\mathbf{Y}$: matrix of samples of a random vector made from realizations $\mathbf{y}_i$, $i \in [1, N]$.
- $\mathbf{N}$: matrix of samples of a random vector made from realizations $\mathbf{n}_i$, $i \in [1, N]$ such that $\mathbf{N} \sim \mathcal{N}_M(0, \sigma^2\mathbf{I})$ and $\operatorname{cov}(\mathbf{X}, \mathbf{Y}) = 0$, where $\mathcal{N}$ represents the normal distribution and $\mathbf{I}$ is the identity matrix.
- $S_{\mathbf{X}}$ and $S_{\mathbf{Y}}$: the empirical covariance matrices obtained from $\mathbf{X}$ and $\mathbf{Y}$, respectively.
- $\lambda_{\mathbf{X},1} \geq \lambda_{\mathbf{X},2} \geq \dots \lambda_{\mathbf{X},M}$: the eigenvalues of $S_{\mathbf{X}}$.
- $\lambda_{\mathbf{Y},1} \geq \lambda_{\mathbf{Y},2} \geq \dots \lambda_{\mathbf{Y},M}$: the eigenvalues of $S_{\mathbf{Y}}$.

2. Principal components on natural images

The authors claim **[3]** that information in a natural image patch can be represented by fewer values than the number of pixels in the patch. Formally, the authors claim that any patch lies in a sub-space $V_{M-m} \subset \mathbb{R}^M$ whose dimension $M - m$ is smaller than the number of pixels of the patch, M.

ASSUMPTION 1. *Let $\boldsymbol{X}$ be the matrix of samples made of N rows, each describing a noise free patch of size M. Then we assume that all realizations $\boldsymbol{x}_i$ lie in a sub-space $V_{M-m} \subset \mathbb{R}^M$ whose dimension $M - m$ is smaller that the dimension M of the $\boldsymbol{x}_i$. Therefore, $\boldsymbol{x}_i$ have zero variance along any direction orthogonal to V_{M-m}*

$$\lambda_{\boldsymbol{X},M-m-1} = \dots = \lambda_{\boldsymbol{X},M} = 0. \tag{40}$$

By assuming the previous result the authors show that for noisy patch observations $\mathbf{Y}$ the following results holds.

THEOREM 2. *Let $\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{n}$, where $\boldsymbol{n}$ is uniform Gaussian noise of variance σ^2 then*

$$\mathbb{E}\left|\lambda_{\boldsymbol{Y},i} - \sigma^2\right| \leq T\sigma^2/\sqrt{N}, \quad i = M - m + 1, \dots, M \tag{41}$$

where $\lambda_{\boldsymbol{Y},i}$ denote the eigenvalues of the covariance matrix $S_{\boldsymbol{Y}}$, N the number of rows in $\boldsymbol{Y}$ and T is a positive constant.

If the number of blocks is large enough, the variance of noise can be estimated from $\lambda_{\mathbf{Y},M}$. Indeed, taking $N \to \infty$ and the M-th eigenvector $\lambda_{\mathbf{Y},M}$ leads to

$$\lim_{N\to\infty} \mathbb{E}\left|\lambda_{\mathbf{Y},M} - \sigma^2\right| = 0. \tag{42}$$

This result provides the way to estimate the variance of the noise from the M-th eigenvector $\lambda_{\mathbf{Y},M}$ when the number of blocks N is large enough. In order to check that this approximation is correct the authors use inequality (41) with $i = M - m + 1$ and approximating σ^2 by λ_M

$$\lambda_{\mathbf{Y},M-m+1} - \lambda_{\mathbf{Y},M} < T\sigma^2/\sqrt{N}. \tag{43}$$

The authors fix the parameter $m = 7$ for blocks of $M_1 \times M_2 = 5 \times 5$ pixels and $T = 49$.

In order to impose patches to actually belong to a V_{M-m} subspace, the authors choose a subset of patches defined as those with smallest variance. The authors choose to use the variance of the blocks in $\mathbf{y}$ as a measure of the distance of a block $\mathbf{y}_i$ to $V_{M-m}, i \in [1, N]$. A simple strategy to select the subset of blocks for which equation (43) is verified consists in taking a small enough p-quantile $Q(p)$ of the list of variances of the noisy blocks and to set

$$B(p) := \{\mathbf{y}_i : s^2(\mathbf{y}_i) \leq Q(p),\ i \in [1, N]\}. \tag{44}$$

In practice, the authors propose an iterative strategy which reduces the quantile p while equation (43) is not satisfied.

3. Algorithm

The noise variance estimation algorithm (Algorithm 30) first obtains an upper bound of the variance of the noise. This upper bound is also used as initialization for the iterative procedure which refines the estimated variance, Function *GetNextEstimate* (in Algorithm 30). This function is iterated until the absolute difference between the new variance and the previous one is negligible.

The function *GetNextEstimate* uses a noise variance initialization, σ_{est}, in order to test inequality (43). The eigenvalues of the set of patches are computed and checked to see if they satisfy Equation (43) with $\sigma = \sigma_{est}$. The number of patches used for this estimation is reduced by decreasing p for the set of blocks $B(p)$ while inequality (43) does not hold. The iteration begins setting $p = 1$ and it is decremented by $\Delta p = 0.05$ after each iteration.

4. Optimizing the PCA computation

Computing the eigenvectors and the corresponding eigenvalues of the sample covariance matrix is a time consuming operation which is called several times in Function *GetNextEstimate*. The PCA is computed on the set $B(p)$ for different values of p. It is possible to use the nested structure of these sets in order to avoid recomputing the covariance matrix at each step.

Algorithm 30 PCA noise estimation.

PCA noise estimation - Compute noise variance.
Input y: list of blocks of the noisy image.
Output $\sigma^2_{\mathbf{est}}$: variance of the noise.

$i_{\max} = 10$
$\varepsilon = 0.001$
$\sigma^2_{\text{ub}} = 3.1\, Q(0.0005, \mathbf{y})$ ▷ Function $Q(p, \mathbf{y})$ gives the p-quantile of $\mathbf{y}$
$\sigma^2_{\text{est}} = \sigma^2_{\text{ub}}$
for i $= 1 \ldots i_{\max}$ **do**
  $\sigma^2_{\text{next}} = \text{GetNextEstimate}(\mathbf{y}, \sigma^2_{\text{est}}, \sigma^2_{\text{ub}})$
  if $\left|\sigma^2_{\text{est}} - \sigma^2_{\text{next}}\right| < \varepsilon$ **then**
   return σ^2_{est}
  end if
  $\sigma^2_{\text{est}} = \sigma^2_{\text{next}}$
end for
  return σ^2_{est}

GetNextEstimate - Refine current variance estimation
Input y: list of blocks of the noisy image.
Input $\sigma^2_{\mathbf{est}}$: current value of the estimated noise variance.
Input $\sigma^2_{\mathbf{ub}}$: upper bound of the noise variance.
Output $\sigma^2_{\mathbf{next}}$: next approximation of the estimated noise variance.

$p_{\min} = 0.06$
$\Delta p = 0.05$
$M = M_1 \times M_2 = 5 \times 5 = 25$
$m = 7$
$T = 49$
$p = 1$
$\sigma^2_{\text{next}} = 0$
while $p \geq p_{\min}$ **do**
  $\lambda_{\mathbf{Y},i} = \text{ApplyPCA}(B(p))$ ▷ $\lambda_{\mathbf{Y},i}$ is the i-th eigenvalue of $S_{\mathbf{Y}}$.
  $\sigma^2_{\text{next}} = \lambda_{\mathbf{Y},M}$
  if $\left(\lambda_{\mathbf{Y},M-m+1} - \lambda_{\mathbf{Y},M} < \frac{T\sigma^2_{\text{est}}}{\sqrt{|B(p)|}}\right) \wedge \left(\sigma^2_{\text{next}} \leq \sigma^2_{\text{ub}}\right)$ **then**
   return σ^2_{next}
  end if
  $p = p - \Delta p$
end while
  return σ^2_{next}

The sample covariance matrix can be written as

$$\frac{1}{|B(p)-1|}\left(\sum_{\mathbf{y}_i \in B(p)} \mathbf{y}_i \mathbf{y}_i^T - \frac{1}{|B(p)|}\sum_{\mathbf{y}_i \in B(p)} \mathbf{y}_i \sum_{\mathbf{y}_i \in B(p)} \mathbf{y}_i^T\right). \tag{45}$$

FIGURE 1. Raw image *IMG_0177*.

The number of operations needed to compute directly this matrix is proportional to $|B(p)|\,M^2$. Since the sets $B(p)$

$$B(p) = \{\mathbf{y}_i : s^2(\mathbf{y}_i) \leq Q(p),\ i \in [1, N]\}.$$

satisfy

$$B(1) \supset B(1-\Delta p) \supset B(1-2\Delta p) \supset B(1-3\Delta p) \supset \ldots,$$

then the set $B(1-j\Delta p)$ can be written as $B(1-j\Delta p) = B(1-(j+1)\Delta p) \cup \mathbf{Y}_j$ being the set $\mathbf{Y}_j := \{\mathbf{y}_i : Q(1-(j+1)\Delta p) < s^2(\mathbf{y}_i) \leq Q(1-j\Delta p)\}$.

Since for disjoint sets X_1 and X_2 one has $C_{X_1 \cup X_2} = C_{X_1} + C_{X_2}$, then

$$C_{B(1-j\Delta p)} = C_{B(1-(j+1)\Delta p)} + C_Y.$$

Let $C_X := \sum_{\mathbf{y}_i \in X} \mathbf{y}_i \mathbf{y}_i^T$ and $\mathbf{c}_X := \sum_{\mathbf{y}_i \in X} \mathbf{y}_i$. With this notation, Expression (45) can be written as

$$\frac{1}{|B(p)-1|}\left(C_{B(p)} - \frac{1}{|B(p)|}\mathbf{c}_{B(p)}\mathbf{c}_{B(p)}^T\right). \tag{46}$$

This strategy permits to use the covariance matrices of previous iteration, but not the eigenvalue decomposition. With Expression (46) the number of operations needed is proportional to M^2.

5. Online demo

5.1. Example: raw image IMG_0177 (ISO 1250, t=1/30s). The results of this example can be reproduced by estimating the noise of the raw image without adding any noise (set $A = B = 0$) and choosing the raw image IMG_0177 (Figure 1) The rest of the parameters are the default parameters of the demo. Figure 1 shows the input noisy image.

Figure 2 shows the noise estimated for the three first scales of the signal-dependent noise with variance $\sigma^2 = 0.5\mathbf{U}$ added to the IMG_0177 image. From left to right: scales S_0 (original), S_1 and S_2. Note that, as expected, the noise standard deviation is divided by approximately two when down-scaling the image by the same ratio.

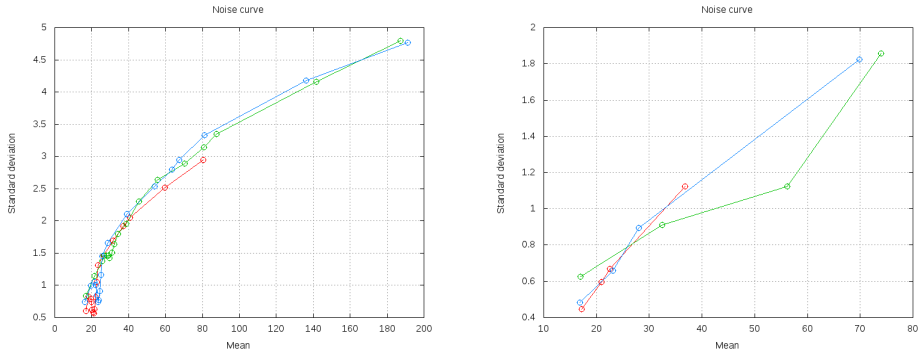


FIGURE 2. The noise estimated for the two first scales of the noisy raw image IMG_0177 (Figure 1) as shown in the IPOL PCA demo. Left: scale S_0. Right: scale S_1.

6. Appendix: proof of Theorem 2

The PCA noise estimation method is based on the fact that the noise variance can be estimated from $\lambda_{\mathbf{Y},M}$ when the condition of Theorem 2 holds and converge is reached (Algorithm 15). To justify completely this result, the proof of Theorem 2 the authors used is reproduced here. To proof Theorem 2 the Gerschgorin circle theorem has to be reviewed, since a lemma from it will be used in the demonstration.

THEOREM 3. *(Gerschgorin)* **[125]** *For a matrix $A \in \mathbb{C}^{M\times M}$ with entries a_{ij} let*

$$a_i := \sum_{j\neq i} |a_{i,j}|$$

and

$$G_i(A) := \{z \in \mathbb{C} : |z - a_{ii}| \leq a_i\}$$

Then all eigenvalues of A lie in $\bigcup_{i=1}^{M} G_i(A)$. Moreover, if m of the Gerschgorin disks $G_i(A)$ are isolated from the other $M-m$, then there are exactly m eigenvalues of A in their union.

LEMMA 1. *Let $A \in \mathbb{C}^{M\times M}$ be a random Hermitian matrix, λ an eigenvalue of A with multiplicity m and normalized eigenvectors $v_1, \ldots, v_m$ and $\tilde{A} = A + B$ a perturbed matrix. If there is a constant $\delta > 0$ such that the distance between λ and the other eigenvalues of A is always greater than δ, then there are exactly m eigenvalues $\tilde{\lambda}_k$ of $\tilde{A}$, which satisfy*

$$\left|\tilde{\lambda}_k - \lambda\right| \leq \max_{i=1,\ldots,m} \sum_{j=1}^{m} \left|\boldsymbol{v}_i^T B \boldsymbol{v}_j\right| + O(\|B\|^2)$$

Proof: let $\Lambda = Q^T A Q$ be the eigen-decomposition of A and $\mathbf{q}_i$ the columns of Q. Without loss of generality

$$\Lambda = \text{diag}\{\underbrace{\lambda, \ldots, \lambda}_{m}, \underbrace{\lambda', \ldots, \lambda'}_{M-m}\}$$

where λ' is an eigenvalue of A other than λ. Then the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m$ form the fist m columns of Q:

$$\mathbf{q}_i = \mathbf{v}_i, \ i = 1, \ldots, m.$$

Let us consider the matrix $\tilde{\Lambda}_a = D_a^{-1} Q^T \tilde{A} Q D_a$, where

$$D_a = \operatorname{diag}\{\underbrace{1, \ldots, 1}_{m}, \underbrace{a, \ldots, a}_{M-m}\}, \ a > 0.$$

The matrices $\tilde{A}$ and $\tilde{\Lambda}_a$ are similar and, hence, have the same eigenvalues.

$$\begin{aligned}
\tilde{\Lambda}_a &= D_a^{-1} Q^T (A + B) Q D_a = \\
&= \Lambda + D_a^{-1} Q^T B Q D_a = \\
&= \Lambda + \begin{bmatrix}
b_{11} & \cdots & b_{1m} & ab_{1\ m+1} & \cdots & ab_{1M} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
b_{m1} & \cdots & b_{mm} & ab_{m\ m+1} & \cdots & ab_{mM} \\
a^{-1}b_{m+1\ 1} & \cdots & a^{-1}b_{m+1\ m} & b_{m+1\ m+1} & \cdots & b_{m+1\ M} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
a^{-1}b_{M1} & \cdots & a^{-1}b_{Mm} & b_{M\ m+1} & \cdots & b_{MM}
\end{bmatrix}
\end{aligned}$$

where $b_{ij} = \mathbf{q}_i^T B \mathbf{q}_j$. Note that $|b_{ij}| \le \|\mathbf{q}_i\| \, \|B\| \, \left\|\mathbf{q}_j\right\| \le \|B\|$. The first m Gerschgorin disks of $\tilde{\Lambda}_a$ have centers $\lambda + b_{ii}$ and radii bounded by

$$\sum_{j=1}^{m} |b_{ij}| - |b_{ii}| + (M-m)a \, \|B\| .$$

Hence, these disks lie entirely in the circle

$$C = \{z \in \mathbb{C} : |z - \lambda| \le p + (M-m)a \, \|B\|\}$$

where $p = \max_{i=1,\ldots,m} \sum_{j=1}^{m} |b_{ij}|$. The other $M - m$ Gerschgorin disks have centers $\lambda' + b_{ii}$ and radii bounded by

$$ma^{-1} \, \|B\| + (M - m - 1) \, \|B\| .$$

Therefore, if

$$p + (M-m)a \, \|B\| + ma^{-1} \, \|B\| + (M-m) \, \|B\| < \delta, \tag{47}$$

the circle C will be disjoint from the last $M - m$ Gerschgorin disks. Let B be so small that

$$p + (M-m) \, \|B\| < \frac{\delta}{2}.$$

In this case, (47) is satisfied if

$$(M-m)a \, \|B\| + ma^{-1} \, \|B\| < \frac{\delta}{2}, \tag{48}$$

and (48) is satisfied if

$$Ma \, \|B\| + Ma^{-1} \, \|B\| < \frac{\delta}{2}, \tag{49}$$

If B is small enough so that $4M\|B\| < \delta$, (49) holds for

$$a = \frac{4M\|B\|}{\delta}.$$

Then, the circle C:

(1) has the radius bounded by

$$p + 4M(M-m)\|B\|^2/\delta = p + O(\|B\|^2)$$

(2) contains the first m Gerschgorin disks of $\tilde{\Lambda}_a$.
(3) is disjoint from the last $M-m$ Gerschgorin disks of $\tilde{\Lambda}_a$.

From Gerschgorin's theorem, C contains exactly m eingenvalues of $\tilde{\Lambda}_a$ and, therefore, exactly m eigenvalues of $\tilde{A}$.

Proof of Theorem 2: Consider the matrices

$$A := S_{\mathbf{X}} + \frac{\sigma^2 N}{N-1} I,$$

$$B := S_{\mathbf{N}} - \frac{\sigma^2 N}{N-1} I + S_{\mathbf{XN}} + S_{\mathbf{NX}}.$$

The matrix $S_{\mathbf{Y}}$ can be represented in the following form: $S_{\mathbf{Y}} = S_{\mathbf{X}} + S_{\mathbf{N}} + S_{\mathbf{XN}} + S_{\mathbf{NX}} = A + B$. Since

$$A\mathbf{v}_{\mathbf{X},i} = S_{\mathbf{X}}\mathbf{v}_{\mathbf{X},i} + \frac{\sigma^2 N}{N-1}\mathbf{v}_{\mathbf{X},i} = \left(\lambda_{\mathbf{X},i} + \frac{\sigma^2 N}{N-1}\right)\mathbf{v}_{\mathbf{X},i},$$

each eigenvalue of A equals the sum of an eigenvalue of $S_{\mathbf{X}}$ and $\frac{\sigma^2 N}{N-1}$, and the eigenvectors of $S_{\mathbf{X}}$ and A are the same. Under assumption 1, the last m eigenvalues of $S_{\mathbf{X}}$ are zeros, therefore the last m eigenvalues of A equal $\frac{\sigma^2 N}{N-1}$. Let $J = \{M-m+1, \dots M\}$ be the set of indices of zero eigenvalues of $S_{\mathbf{X}}$. Using Lemma 1, for $j \in J$

$$\left|\lambda_{\mathbf{Y},k} - \frac{\sigma^2 N}{N-1}\right| \le \max_{i\in J} \sum_{j\in J} \left|\mathbf{v}_{\mathbf{X},i}^T B \mathbf{v}_{\mathbf{X},j}\right| + O(\|B\|^2). \tag{50}$$

Consider the first part on the right side of (50). Denoting the sample covariance by q, for $i, j \in J$ we have

$$\left|\mathbf{v}_{\mathbf{X},i}^T S_{\mathbf{XN}} \mathbf{v}_{\mathbf{X},j}\right| = \left|q(\mathbf{v}_{\mathbf{X},i}^T \mathbf{X}, \mathbf{v}_{\mathbf{X},j}^T \mathbf{N})\right| \le \sqrt{s^2(\mathbf{v}_{\mathbf{X},i}^T \mathbf{X}) s^2(\mathbf{v}_{\mathbf{X},j}^T \mathbf{N})} = \sqrt{\lambda_{\mathbf{X},i} s^2(\mathbf{v}_{\mathbf{X},j}^T \mathbf{N})} = 0$$

and, similarly,

$$\mathbf{v}_{\mathbf{X},i}^T S_{\mathbf{NX}} \mathbf{v}_{\mathbf{X},j} = 0.$$

Hence, for $i \in J$

$$\begin{aligned} \textstyle\sum_{j\in J}\left|\mathbf{v}_{\mathbf{X},i}^T B \mathbf{v}_{\mathbf{X},j}\right| &= \textstyle\sum_{j\in J}\left|\mathbf{v}_{\mathbf{X},i}^T (S_{\mathbf{N}} - \frac{\sigma^2 N}{N-1} I)\mathbf{v}_{\mathbf{X},j}\right| \le \\ &\le \textstyle\sum_{j\in J} \|\mathbf{v}_{\mathbf{X},i}\| \left\|S_{\mathbf{N}} - \frac{\sigma^2 N}{N-1} I\right\| \|\mathbf{v}_{\mathbf{X},j}\| = \\ &= m\left\|S_{\mathbf{N}} - \tfrac{\sigma^2 N}{N-1} I\right\| \end{aligned}$$

and

$$\max_{i\in J}\sum_{j\in J}\left|\mathbf{v}_{\mathbf{X},i}^T B\mathbf{v}_{\mathbf{X},j}\right| \le m\left\|S_{\mathbf{N}}-\frac{\sigma^2 N}{N-1}I\right\| \le m\sqrt{\sum_{i,j=1}^{M}\left(n_{ij}-\frac{\delta_{ij}\sigma^2 N}{N-1}\right)^2}, \tag{51}$$

where n_{ij} are the entries of $S_{\mathbf{N}}$ and δ_{ij} is the Kronecker delta. Since $\mathbf{N}\sim\mathcal{N}_M(0,\sigma^2 I)$, the matrix $(N-1)S_{\mathbf{N}}$ has the Wishart distribution $W_M(\sigma^2 I, M)$. Therefore,

$$\mathbb{E}\left(n_{ii}\right)=\frac{N\sigma^2}{N-1},\ \mathrm{Var}\left(n_{ii}\right)=\frac{2\sigma^4 N}{(N-1)^2},$$

$$\mathbb{E}\left(n_{ij}\right)=0,\ \mathrm{Var}\left(n_{ij}\right)=\frac{\sigma^4 N}{(N-1)^2},\ i\neq j \tag{52}$$

and

$$\left[\mathbb{E}\left(n_{ij}-\frac{\delta_{ij}\sigma^2 N}{N-1}\right)\right]^2=\mathrm{Var}\left(n_{ij}\right)\le\frac{2\sigma^4 N}{(N-1)^2}. \tag{53}$$

Since $\mathbb{E}\left(\sqrt{X}\right)\le\sqrt{\mathbb{E}\left(X\right)}$, (51) and (53) can be combined to give

(54)

$$\mathbb{E}\left(\max_{i\in J}\sum_{j\in J}\left|\mathbf{v}_{\mathbf{X},i}^T B\mathbf{v}_{\mathbf{X},j}\right|\right)\le m\sqrt{\sum_{i,j=1}^{M}\mathbb{E}\left(\left(n_{ij}-\frac{\delta_{ij}\sigma^2 N}{N-1}\right)^2\right)}\le\frac{mM\sigma^2\sqrt{2N}}{N-1}=O(\sigma^2/\sqrt{N}).$$

For the second part on the right side of (50), let us construct an upper bound for $\mathbb{E}\left(\|B\|^2\right)$ now. Let b_{ij} be the entries of B and c_{ij} the entries of $S_{\mathbf{XN}}+S_{\mathbf{NX}}$. Since

$$\mathbb{E}\left(S_{\mathbf{XN}}+S_{\mathbf{NX}}\right)=\mathrm{cov}(\mathbf{X},\mathbf{N})+\mathrm{cov}(\mathbf{N},\mathbf{X})=0,$$

$\mathbb{E}\left(c_{ij}\right)=0$. Additionally, $\mathrm{Var}\left(c_{ij}\right)=\sigma^2\frac{\mathrm{Var}(X)}{N-1}$. Combining it with (52), it gives

$$\mathbb{E}\left(b_{ij}\right)=0$$

and

$$\begin{aligned}\mathrm{Var}\left(b_{ij}\right) &= \mathrm{Var}\left(n_{ij}\right)+\mathrm{Var}\left(c_{ij}\right)+2\mathrm{cov}(n_{ij},c_{ij})\le\\ &\le \left(\sqrt{\mathrm{Var}\left(n_{ij}\right)}+\sqrt{\mathrm{Var}\left(c_{ij}\right)}\right)^2\le\\ &\le \left(\frac{\sigma^2\sqrt{2N}}{N-1}+\frac{\sigma\sqrt{\mathrm{Var}(\mathbf{X})}}{\sqrt{N-1}}\right)^2=\\ &= O\left(\frac{\left(\sigma^2+\sigma\sqrt{\mathrm{Var}(\mathbf{X})}\right)^2}{N}\right).\end{aligned}$$

Therefore,

$$\mathbb{E}\left(\|B\|^2\right)\le\mathbb{E}\left(\sum_{i,j=1}^{M}b_{i,j}^2\right)=\sum_{i,j=1}^{M}\mathrm{Var}\left(b_{ij}\right)=O\left(\frac{\left(\sigma^2+\sigma\sqrt{\mathrm{Var}\left(\mathbf{X}\right)}\right)^2}{N}\right). \tag{55}$$

Combining (50), (54), and (55) gives the final result. Since $N \to \infty$, $\frac{1}{N}$ is infinitesimal compared to $\frac{1}{\sqrt{N}}$,

$$\mathbb{E}\left(\left|\lambda_{\mathbf{Y},i} - \frac{\sigma^2 N}{N-1}\right|\right) = O\left(\frac{\sigma^2}{N}\right) + O\left(\frac{\left(\sigma^2 + \sigma\sqrt{\operatorname{Var}(\mathbf{X})}\right)^2}{N}\right) = O(\sigma^2/\sqrt{N}) \quad \square$$

CHAPTER 13

Evaluation of the adapted methods

In Chapters 10, 11, and 12, the Ponomarenko et al., Percentile and PCA signal-dependent noise estimation methods were described in detail. In this chapter we evaluate them using three different tests:

- *A test with simulated white Gaussian noise using the noise-free images in Figure 1 (Section 1). The aim of this test is to verify that the methods give accurate results for the noise is homoscedastic (fixed variance, not depending on the intensity). As explained in Chapter 9, it is needed that the noise estimator is accurate before adapting it to signal-dependent noise, since the number of samples will be shared by total number of bins. We consider seven bins to classify the blocks according to their means (see Section 1.1 of Chapter 9).*
- *A test with raw images (Section 2), which compares the noise curve obtained by the methods to the ground-truth noise curve obtained by the camera that took the photograph, under the same ISO speed (see Chapter 2).*
- *A test on multiscale coherence (Section 3). The standard deviation of a Gaussian white noise is divided by two when the image is down-scaled. By* down-scaling *the image we mean a sub-sampling of the image where each block of four pixels is substituted by their mean. This test checks if the measured noise is divided by two at each image down-scaling.*

1. Evaluation with simulated white Gaussian noise

In this test, white Gaussian noise was simulated and then added to a set of ten noise-free images. Since the noise is perfectly known *a priori*, it can be used as a ground truth. One can therefore compute the RMSE of the standard deviation estimations. Fig. 1 shows a set of 704×469 pixels noise-free images that were used in this test. In order to get the noise-free images, we have applied the following procedure. The pictures were taken with a Canon EOS 30D reflex camera of scenes under good lighting conditions and with a low ISO level. To reduce further the noise level, the average of each block of 5×5 pixels was computed, reducing therefore the noise by a factor of 5. Since the images are RGB, the mean of the three channels was computed, reducing the noise by a further $\sqrt{3}$ factor. Therefore the noise was reduced by a $5\sqrt{3} \simeq 8.66$ factor. Finally, the images, which already had a good SNR before they were processed, can be considered noise-free.

To measure the error made when estimating the standard deviation σ of the simulated noise in the bin b in the image i, the RMSE along all the bins was used. This RMSE is denoted by $E_{i,\sigma}^{(1)}$

FIGURE 1. Set of noise-free images used to test the noise estimation algorithm with white Gaussian noise. From left to right and from top to bottom: bag, building1, computer, dice, flowers2, hose, lawn, leaves, stairs and traffic. Each image is 704×469 pixels.

and it is defined by

$$E_{i,\sigma}^{(1)} := \sqrt{\frac{1}{|B|}\sum_{b=1}^{|B|} |\hat{\sigma}_{i,b} - \sigma|^2}, \tag{56}$$

where $|I|$ is the number of images, i is the image index ($1 \leq i \leq |I|$), $|B|$ is the number of bins, b is the index of the bin ($1 \leq b \leq |B|$), σ is the standard deviation of the simulated noise and $\hat{\sigma}_{i,b}$ is the estimated noise for the image i at the bin b. Tables 1, 2, and 3 show the obtained $E_{i,\sigma}^{(1)}$ for each image i and each σ of the simulated noise, for the Ponomarenko, Percentile, and PCA methods, namely. For the Ponomarenko et. al and the Percentile methods seven bins are used. For the PCA method three bins are used, since it requires mores samples/bin. A new image is added to the set of noise-free images, the *flat image*, which is a constant image where all pixels have the value 127, useful to measure the performance of the noise estimators with an image of pure homoscedastic Gaussian noise.

Note that the PCA method is able to give an accurate estimation of the noise even when $\sigma = 1$ when the image has large flat zones (building1, computer, dice, flowers2, traffic), but fails to give a good estimation for very textured images (bag, hose, leaves, lawn). For very high noise levels ($\sigma = 50$, $\sigma = 80$) the estimation is inaccurate. Surprisingly, the PCA method behaves better for small levels of noise.

For the Percentile method, it is apparent that the highly textured images create a significant error, particularly when little noise was added. Estimates of noise below $\sigma = 2$ are therefore obviously clearly unreliable. All in all, the estimate is nevertheless quite reliable for values $\sigma > 5$.

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.86	0.48	0.28	0.59	0.45	1.10	1.05
building1	0.19	0.16	0.06	0.29	0.52	1.16	1.66
computer	0.20	0.10	0.19	0.21	0.48	2.26	1.82
dice	0.12	0.05	0.11	0.18	0.43	0.94	2.17
flowers2	0.19	0.08	0.13	0.30	0.50	1.60	0.89
hose	0.86	0.60	0.39	0.42	0.58	1.68	1.18
lawn	1.46	1.25	0.68	0.47	0.51	1.40	1.92
leaves	1.47	1.09	0.65	0.56	0.44	1.43	2.17
stairs	0.59	0.32	0.34	0.28	0.49	0.80	1.30
traffic	0.13	0.09	0.21	0.22	0.64	1.44	2.21
Flat image	0.02	0.03	0.05	0.17	0.37	1.34	1.23
$E^{(2)}_{\sigma}$	**0.56**	**0.39**	**0.28**	**0.34**	**0.49**	**1.38**	**1.60**

TABLE 1. This table shows, for the Ponomarenko method, the $E^{(1)}_{i,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{i,b}$ of all the images. The percentile $p = 0.005$ and seven bins are used.

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.74	0.62	0.47	0.37	0.81	0.73	2.26
building1	0.34	0.24	0.55	0.62	0.82	1.20	1.58
computer	0.35	0.36	0.55	0.64	0.86	1.25	3.10
dice	0.12	0.12	0.17	0.24	0.50	1.20	1.68
flowers2	0.15	0.13	0.15	0.27	0.81	1.54	2.82
hose	0.87	0.62	0.49	0.41	0.49	1.37	1.55
lawn	0.99	1.20	0.86	0.64	0.62	1.64	1.90
leaves	1.43	1.11	0.95	0.74	0.70	0.96	1.67
stairs	0.94	0.89	0.66	0.56	0.64	0.89	1.34
traffic	0.45	0.42	0.56	0.58	0.91	1.36	2.23
Flat image	0.02	0.04	0.15	0.10	0.20	1.25	2.63
$E^{(2)}_{\sigma}$	**0.58**	**0.52**	**0.51**	**0.47**	**0.67**	**1.22**	**2.07**

TABLE 2. This table shows, for the Percentile method, the $E^{(1)}_{i,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{i,b}$ of all the images. The parameters used in the tests are percentile $p = 0.005$, block of size 15×15, DCT filter with support 7×7 and seven bins.

The last row is the RMSE obtained for a given σ and all the images. It is denoted by $E^{(2)}_{\sigma}$ and defined as

$$E^{(2)}_{\sigma} := \sqrt{\frac{1}{|B||I|}\sum_{i=1}^{|I|}\sum_{b=1}^{|B|}|\hat{\sigma}_{i,b} - \sigma|^2} = \sqrt{\frac{1}{|I|}\sum_{i=1}^{|I|}\left(E^{(1)}_{i,\sigma}\right)^2}. \tag{57}$$

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.65	0.49	0.29	0.27	0.40	6.04	22.98
building1	0.18	0.10	0.21	0.46	0.65	2.17	2.38
computer	0.14	0.11	0.17	0.37	0.67	1.10	3.43
dice	0.13	0.06	0.09	0.30	0.59	1.81	3.77
flowers2	0.17	0.10	0.07	0.22	0.81	2.78	9.28
hose	0.76	0.66	0.35	0.28	0.25	9.61	26.78
lawn	0.87	0.62	0.46	0.57	0.32	13.71	29.49
leaves	1.16	0.71	0.35	0.31	0.40	3.67	17.83
stairs	0.55	0.55	0.50	0.53	1.01	17.15	33.67
traffic	0.14	0.23	0.34	0.59	0.59	0.58	3.36
Flat image	0.58	1.18	2.90	5.85	11.59	28.92	46.80
$E^{(2)}_{\sigma}$	**0.49**	**0.44**	**0.52**	**0.89**	**1.57**	**7.96**	**18.16**

TABLE 3. This table shows, for the PCA method, the $E^{(1)}_{i,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{i,b}$ of all the images. Note that the PCA method is able to give an accurate estimation of the noise even when $\sigma = 1$ when the image has large flat zones (building1, computer, dice, flowers2, traffic), but fails to given a good estimation for very textured images (bag, hose, leaves, lawn). For very high levels of noise ($\sigma = 50$, $\sigma = 80$) the estimation is inaccurate. Surprisingly, the method behaves better for small levels of noise. Three bins are used.

For completeness, the results corresponding to the estimation using just a single bin are shown in Tables 4 (Ponomarenko), 5 (Percentile), and 6 (PCA). Assuming homoscedastic noise is not a realistic model, but it is needed that the algorithms give accurate results with homoscedastic noise before attempting their adaptation to signal-dependent noise (see Chapter 2).

Tables 7 (Ponomarenko), 8 (Percentile), and 9 (PCA) show the obtained $E^{(2)}_{\sigma}$ RMSE depending on the number of the iterations of the noise curve filter (see Section 1.2). Using five filtering iterations seems to be safe for any image with independence of the standard deviation of the noise, the kind of textures or the number and type of the edges the image may contain. More than five iterations is useless.

2. Evaluation comparing the noise curve of the raw image with the ground truth

In this test, the noise curve obtained by the algorithm for the raw images in Figure 2 (12 bits/channel, ISO 1600, t=1/30s) was compared to the “ground truth" noise curve of the camera for that ISO and exposure times. The ground truth was obtained by computing for each pixel the standard deviation of a large burst [**24**] of fixed snapshots of the same calibration pattern (Figure 3).

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	1.78	1.57	0.73	0.32	0.26	0.13	0.16
building1	0.16	0.09	0.02	0.10	0.07	0.04	0.30
computer	0.15	0.05	0.08	0.05	0.05	0.15	0.23
dice	0.11	0.06	0.01	0.03	0.07	0.51	0.29
flowers2	0.09	0.03	0.02	0.05	0.11	0.17	0.07
hose	0.75	0.36	0.24	0.17	0.06	0.44	0.22
lawn	1.86	0.42	0.75	0.27	0.37	0.09	0.00
leaves	3.04	1.30	0.55	0.67	0.44	0.41	0.05
stairs	1.06	0.85	0.45	0.23	0.14	0.41	0.27
traffic	0.10	0.07	0.08	0.13	0.11	0.47	0.27
Flat image	0.00	0.01	0.01	0.02	0.05	0.23	0.22
$E^{(2)}_{\sigma}$	**0.83**	**0.44**	**0.27**	**0.18**	**0.16**	**0.28**	**0.19**

TABLE 4. This table shows, for the Ponomarenko method, the $E^{(1)}_{1,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{1,b}$ of all the images. It corresponds to the original signal-independent method, using a single bin and the iterations show in Algorithm 26 to fix percentile K.

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.71	0.60	0.43	0.32	0.65	0.29	1.22
building1	0.21	0.22	0.29	0.21	0.51	0.55	1.24
computer	0.29	0.25	0.38	0.32	0.08	0.99	0.86
dice	0.11	0.11	0.05	0.08	0.29	1.06	1.24
flowers2	0.13	0.10	0.12	0.03	0.00	0.54	2.03
hose	0.60	0.44	0.28	0.19	0.11	0.29	0.48
lawn	0.68	0.81	0.65	0.50	0.28	0.74	0.39
leaves	1.34	1.10	0.91	0.64	0.51	0.76	0.05
stairs	0.82	0.80	0.59	0.40	0.56	0.38	0.52
traffic	0.32	0.33	0.42	0.21	0.58	0.50	0.34
Flat image	0.02	0.03	0.05	0.05	0.16	0.61	1.93
$E^{(2)}_{\sigma}$	**0.47**	**0.44**	**0.38**	**0.27**	**0.34**	**0.61**	**0.94**

TABLE 5. This table shows, for the Percentile method, the $E^{(1)}_{1,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{1,b}$ of all the images. The parameters used in the tests are percentile $p = 0.005$, block of size 15×15, DCT filter with support 7×7 and a single bin.

Image / $E^{(1)}_{i,\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
bag	0.55	0.40	0.21	0.08	0.35	0.15	1.27
building1	0.17	0.08	0.08	0.12	0.18	1.48	1.83
computer	0.11	0.05	0.12	0.28	0.25	0.69	3.03
dice	0.11	0.07	0.07	0.26	0.52	1.71	3.76
flowers2	0.14	0.09	0.02	0.08	0.54	1.87	2.76
hose	0.32	0.25	0.08	0.20	0.18	0.91	2.83
lawn	0.33	0.12	0.15	0.32	0.38	1.97	1.65
leaves	1.02	0.62	0.06	0.18	0.18	0.29	2.34
stairs	0.38	0.24	0.24	0.44	0.29	1.25	2.61
traffic	0.25	0.28	0.17	0.33	0.29	0.03	2.99
Flat image	0.04	0.08	0.20	0.40	0.85	2.47	3.22
$E^{(2)}_{\sigma}$	**0.31**	**0.21**	**0.13**	**0.25**	**0.37**	**1.17**	**2.57**

TABLE 6. This table shows, for the PCA method, the $E^{(1)}_{1,\sigma}$ RMSE after adding simulated noise to the set of noise-free images (Figure 1) with several values of standard deviation σ. The last row is the $E^{(2)}_{\sigma}$ RMSE using the estimated $\hat{\sigma}_{1,b}$ of all the images. A single bin is used.

Image / $E^{(2)}_{\sigma}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
No filtering	0.56	0.39	0.28	0.34	0.49	1.38	1.60
1 iteration	0.55	0.38	0.26	0.27	0.41	1.23	1.35
2 iterations	0.54	0.37	0.24	0.24	0.38	1.16	1.22
3 iterations	0.53	0.37	0.23	0.23	0.36	1.12	1.13
4 iterations	0.53	0.36	0.23	0.22	0.36	1.10	1.11
5 iterations	0.52	0.35	0.22	0.21	0.35	1.09	1.10
6 iterations	0.51	0.35	0.21	0.21	0.35	1.08	1.10
7 iterations	0.51	0.34	0.21	0.21	0.35	1.08	1.09

TABLE 7. This table shows, for the Ponomarenko method, the obtained $E^{(2)}_{\sigma}$ RMSE values depending on the number of iterations of the noise curve filtering and the standard deviation of the noise (see Section 1.2). The percentile $p = 0.005$ and seven bins are used. Five iterations is the recommend value, since using more iterations does not improve the result significantly and it could soften too much the noise curves for certain images.

To get the ground truth of the camera, we fixed the ISO sensitivity (in this case at ISO 1600) and used four exposure times, $t \in \{1/30s, 1/250s, 1/400s, 1/640s\}$. For each exposure time about two hundred pictures of the pattern were taken (see Figure 3). After cropping the area of the image that does not contain the calibration pattern, the final size of the raw image was 1352×1952. Since

Image / $E_\sigma^{(2)}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
No filtering	0.58	0.52	0.51	0.47	0.67	1.22	2.07
1 iteration	0.57	0.52	0.49	0.44	0.60	1.12	1.84
2 iterations	0.56	0.52	0.49	0.42	0.57	1.07	1.75
3 iterations	0.56	0.51	0.48	0.41	0.55	1.04	1.70
4 iterations	0.55	0.50	0.47	0.39	0.53	1.02	1.68
5 iterations	0.55	0.50	0.46	0.38	0.52	1.00	1.67
6 iterations	0.54	0.49	0.46	0.38	0.51	0.99	1.66
7 iterations	0.54	0.48	0.45	0.37	0.50	0.98	1.65

TABLE 8. This table shows, for the Percentile method, the obtained $E_\sigma^{(2)}$ RMSE values depending on the number of iterations of the noise curve filtering and the standard deviation of the noise (see Section 1.2). The parameters used are percentile $p = 0.005$, block of size 15×15, the DCT filter with support 7×7 and seven bins. Five iterations is the recommend value, since using more iterations does not improve the result significantly and it could soften too much the noise curves for certain images.

Image / $E_\sigma^{(2)}$	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$	$\sigma = 50$	$\sigma = 80$
No filtering	0.49	0.44	0.52	0.89	1.57	7.96	18.16
1 iteration	0.49	0.42	0.48	0.82	1.46	7.22	16.90
2 iterations	0.49	0.41	0.46	0.78	1.39	6.66	15.93
3 iterations	0.49	0.41	0.45	0.76	1.35	6.23	15.17
4 iterations	0.48	0.40	0.45	0.75	1.34	6.25	15.19
5 iterations	0.48	0.40	0.45	0.75	1.34	6.26	15.20
6 iterations	0.48	0.40	0.45	0.74	1.33	6.27	15.21
7 iterations	0.47	0.40	0.45	0.74	1.33	6.28	15.23

TABLE 9. This table shows, for the PCA method, the obtained $E_\sigma^{(2)}$ RMSE values depending on the number of iterations of the noise curve filtering and the standard deviation of the noise (see Section 1.2). Three bins are used. Five iterations is the recommended value, since using more iterations does not improve the result significantly and it could smooth too much the noise curves for certain images.

each 2×2 block of the CFA[1] contains one sample of the red channel, two samples of the green channel and one sample of the blue channel, one of the green channels can be discarded to get a single color pixel of each 2×2 block of the CFA, given an effective size of the color raw image of 676×976 pixels. The position of the camera was fixed when taking the snapshots of the image. Assuming constant lighting, the variance along several samples of image coming from different

[1]Color Filter Array.

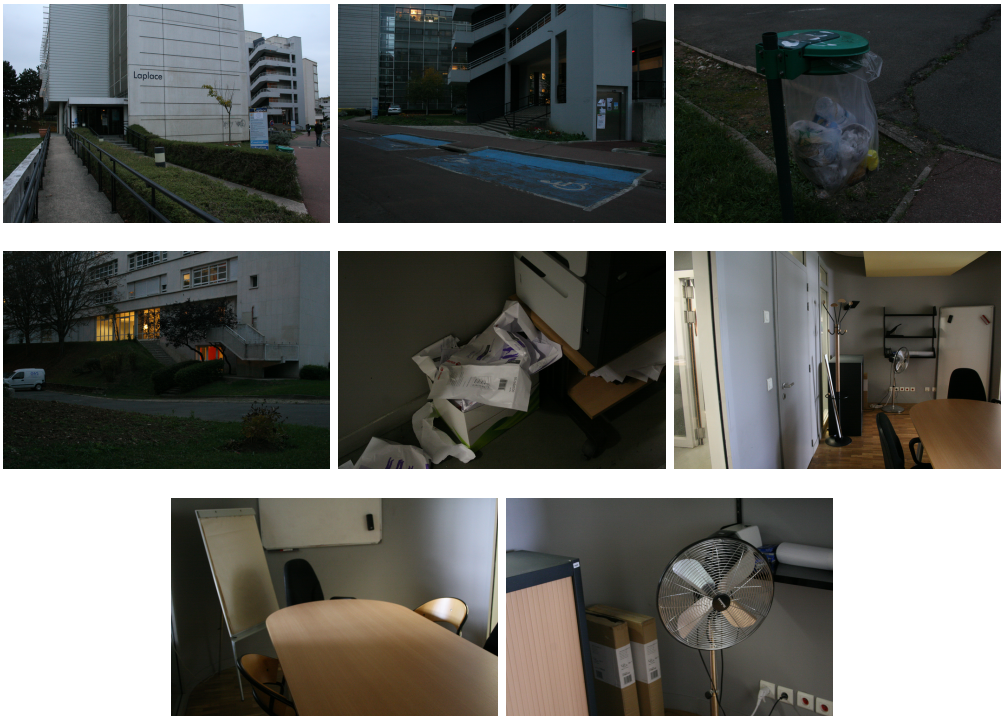


FIGURE 2. Set of raw images used to test the noise estimation algorithm using 25 bins and without any noise curve filtering. The images are raw 12 bits/channel, taken with a Canon EOS 30D camera, ISO 1600 and exposure time t=1/30s. From left to right and up to bottom: images 1, 2, 3, 4, 5, 6, 7, and 8.

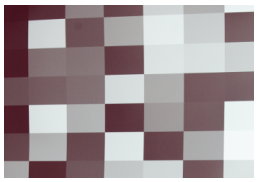


FIGURE 3. One of the pictures of the calibration pattern mire used to build the ground truth noise curve of the camera.

images for the same pixel position can only be explained by the noise. Therefore, it was possible to measure the mean of a block and the temporal standard deviation along all the snapshots to create an association mean→standard deviation, that is, a ground truth for camera noise curve, given the ISO and exposure times. Moreover, since the exposure time only affects the photon count and not the noise model, it was possible to overlap the noise curves for the four exposure times tested in a single ground truth noise curve depending only on the ISO parameter (see Figure 4).

Given an estimated noise curve A of a test image, its control points are the pairs $(\hat{\mu}_{A,i,b}, \hat{\sigma}_{A,i,b}) \in A$ where $\hat{\mu}_{A,i,b}$ is the mean intensity for bin b and image i in A and $\hat{\sigma}_{A,i,b}$ is the corresponding standard deviation values for bin b and image i in A. In the same way, given a ground truth noise curve G, its control points are the pairs $(\mu_{G,v}, \sigma_{G,v}) \in G$. Unfortunately the means of the noise

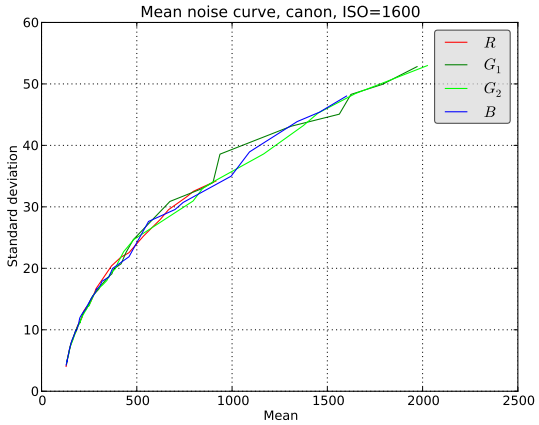


FIGURE 4. Ground truth of the Canon EOS 30D camera with ISO=1600.

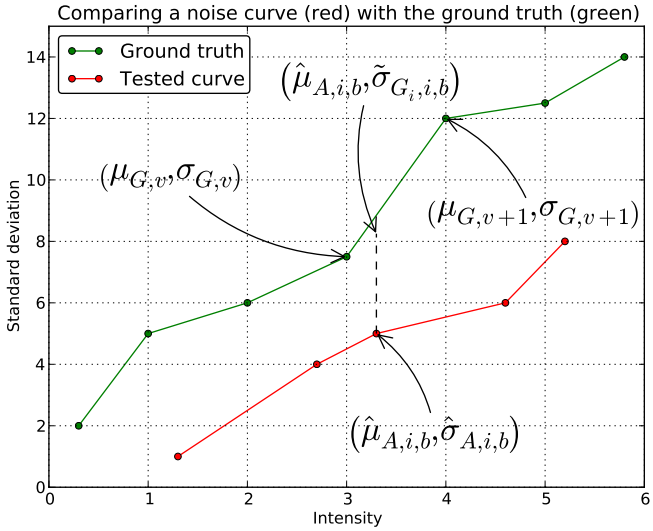


FIGURE 5. Checking a noise curve A (red) against the ground truth G (green), where i is the index of the image, b is the index of the bin, $(\hat{\mu}_{A,i,b}, \hat{\sigma}_{A,i,b})$ are the control points of the noise curve A, $(\mu_{G,v}, \sigma_{G,v})$ are the control points of G, and $\tilde{\sigma}_{G_i,i,b}$ is the standard deviation value projected from A into G.

curve A and those in G do not necessarily coincide; that is, $\hat{\mu}_{A,i,b} \neq \mu_{G,v}$ for most (b, v) pairs. To solve this problem, instead of using G, a new ground truth curve $\tilde{G}_i$ is used. This $\tilde{G}_i$ curve has the same means $\hat{\mu}_{A,i,b}$ as A (and therefore the same number of bins), and its standard deviation values are obtained by a simple proportionality rule. Therefore, the control points in the new curve $\tilde{G}_i$ are $(\hat{\mu}_{A,i,b}, \tilde{\sigma}_{G_i,i,b}) = \left(\hat{\mu}_{A,i,b}, \frac{\sigma_{G,v+1}-\sigma_{G,v}}{\mu_{G,v+1}-\mu_{G,v}}(\hat{\mu}_{A,i,b} - \mu_{G,v+1}) + \sigma_{G,v}\right)$ where v is the index of the bin in the curve G such that $\mu_{G,v} \leq \hat{\mu}_{A,i,b} < \mu_{G,v+1}$ (see Figure 5).

The error between the ground truth noise curve G and the test noise curve A for the image i and bin b is defined as

$$E^{(3)}_{G,A,i,b} := |\tilde{\sigma}_{G_i,i,b} - \hat{\sigma}_{A,i,b}| = \left| \frac{(\sigma_{G,v+1} - \sigma_{G,v})(\hat{\mu}_{A,i,b} - \mu_{G,v+1})}{\mu_{G,v+1} - \mu_{G,v}} + \sigma_{G,v} - \hat{\sigma}_{A,i,b} \right|.$$

The values of $E^{(3)}_{G,A,i,b}$ for each test image in Figure 2 and each method are shown in Table 10. The Ponomarenko method gives systematically the smallest error. The PCA method is significantly sensitive to textures: the error is worse for those images that are highly textured (for example, Img. 3) and better when they contain large flat zones (for example, Img. 6 and Img. 7).

Method	Img. 1	Img. 2	Img. 3	Img. 4	Img. 5	Img. 6	Img. 7	Img. 8
Ponomarenko	**0.802**	**0.381**	**0.372**	**0.307**	**0.437**	**0.694**	**0.436**	**0.580**
Percentile	0.910	0.680	0.836	0.850	0.871	0.867	0.779	0.669
PCA	0.984	1.107	3.577	1.640	2.103	1.010	0.788	1.381

TABLE 10. Values of $E^{(3)}_{G,A,i,b}$ measuring the error between the noise curve A obtained for each test image i (Figure 2) and the ground truth curve G for the Canon EOS 30D with ISO 1600. The Ponomarenko method gives systematically the smallest error. The PCA method is significatly sensitive to textures: the error is worse for those images that are highly textured (for example, Img. 3) and better when they contain large flat zones (for example, Img. 6 and Img. 7).

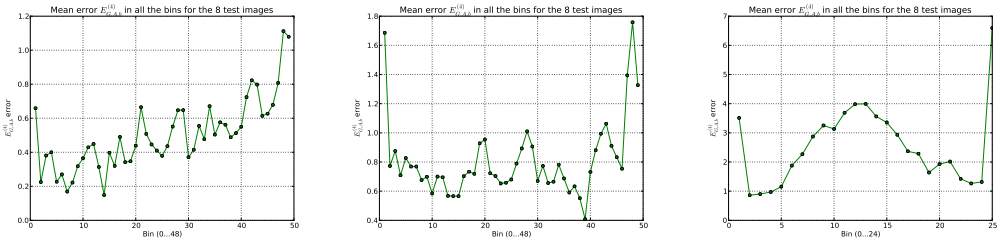
To test the average behavior of the algorithm in all the bins of any test image, we define a mean error function $E^{(4)}_{G,A,b}$ as the mean of the $E^{(3)}_{G,A,i,b}$ values over the $|I|$ images for each of the $|B|$ bins, that is,

$$E^{(4)}_{G,A,b} := \frac{1}{|I|} \sum_{i=0}^{|I|-1} E^{(3)}_{G,A,i,b}. \tag{58}$$

Figure 6 shows the $E^{(4)}_{G,A,b}$ error for all the 49 bins (25 bins in the PCA method) of the test images in Figure 2 for the first green channel, in the three methods.

3. Evaluation of the multiscale coherence of the result

Consider the down-scaling operator $\mathbf{S}$ that tessellates the image into 2×2 pixels blocks, and replaces each block by a pixel having the mean of the four previous pixels as new value. If $\tilde{\mathbf{U}}$ is a discrete pure Gaussian noise image with standard deviation σ, then $\mathbf{S}(\tilde{\mathbf{U}})$ has standard deviation $\frac{\sigma}{2}$. Indeed, if a block $\mathbf{W}$ contains the pixels $\{u_1, u_2, u_3, u_4\}$ each one with variance σ^2, the variance of the mean of $\mathbf{W}$ is $\mathrm{Var}\bar{\mathbf{W}} = \mathrm{Var}\frac{u_1+u_2+u_3+u_4}{4} = \frac{1}{16}[\mathrm{Var}u_1 + \mathrm{Var}u_2 + \mathrm{Var}u_3 + \mathrm{Var}u_4] = \frac{1}{16}[4\sigma^2] = \frac{\sigma^2}{4}$. Therefore, the standard deviation is $\mathrm{Std}(\bar{\mathbf{W}}) = \frac{\sigma}{2}$; the noise has been divided by two. The objective of this test is to check if the noise estimation algorithm indeed divides the noise by two when the



(a) (b) (c)

FIGURE 6. Mean error $E^{(4)}_{G,A,b}$ for the 49 bins (25 in the case of the PCA method) of all the eight tests images in Figure 2, for the Ponomarenko (a), Percentile (b) and PCA (c) methods.

image is down-scaled several times.

(59)
$$\text{Set:}\quad E^{(5)}_{A_0,A_k,i,b} = \left|\frac{\tilde{\sigma}_{A_0,i,b}}{\hat{\sigma}_{A_k,i,b}} - 2^k\right| = \left|\frac{(\hat{\sigma}_{A_0,i,v+1} - \hat{\sigma}_{A_0,i,v})(\hat{\mu}_{A_k,i,b} - \hat{\mu}_{A_0,i,v+1})}{\hat{\sigma}_{A_k,i,b}(\hat{\mu}_{A_0,i,v+1} - \hat{\mu}_{A_0,i,v})} + \frac{\hat{\sigma}_{A_0,i,v}}{\hat{\sigma}_{A_k,i,b}} - 2^k\right|,$$

where

- A_k is the noise curve corresponding to the input image i after applying the down-scaling operator k times. For example, if $k = 2$ then A corresponds to the curve of the noise estimation of $\mathbf{SS}(\tilde{\mathbf{U}})$.
- i is the image index, for the raw images in Fig. 2, $1 \leq i \leq |I|$, where $|I|$ is the number of images. $|I| = 8$ images were used.
- b is the bin index, $1 \leq b \leq |B_k|$ where $|B_k|$ is the number of bins of the noise curve at scale k. For the test images $|B_0| = 49$, $|B_1| = 12$ and $|B_2| = 3$ bins are used.
- v is the index of the bin in the curve A_0 such that $\hat{\mu}_{A_0,i,v} \leq \hat{\mu}_{A_k,i,b} < \hat{\mu}_{A_0,i,v+1}$ (see Figure 7).

Remark: since the noise should be divided by two when operator $\mathbf{S}$ is applied and an ideal noise estimator is used, the relation $\frac{\tilde{\sigma}_{A_0,i,b}}{\tilde{\sigma}_{A_k,i,b}}$ between the standard deviation estimations at scale 0 and scale k (applying k times $\mathbf{D}$) should be equal to 2^k. The error $E^{(5)}_{A_0,A_k,i,b}$ measures, for the tested noise estimator, the absolute deviation from the ideal value 2^k at each bin. To get a mean estimation of the error $E^{(5)}_{A_0,A_k,i,b}$ along all the test images and bins in Figure 2, we define another error function as

(60)
$$E^{(6)}_{A_0,A_k,b} := \frac{1}{|I||B_k|}\sum_{i=1}^{|I|}\sum_{b=1}^{|B_k|} E^{(5)}_{A_0,A_k,i,b}$$

Table 11 shows the obtained mean down-scale error $E^{(6)}_{A_0,A_k,b}$ for the raw images in Figure 2 depending on the scale k, for the three methods. The measurements are done for one of the green channels of the raw images. The results of the PCA method are significantly worse than those given by the Ponomarenko et al. [**15**] and the Percentile [**17**] methods. The reason is that the

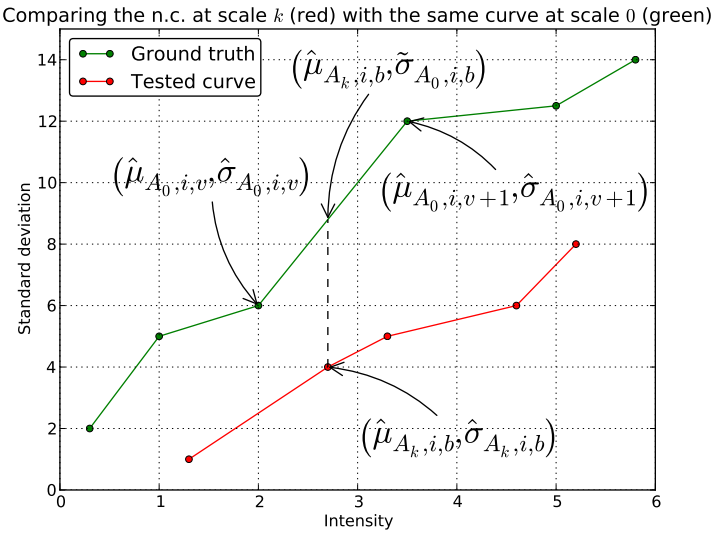


FIGURE 7. Checking a noise curve A_k at scale k (red) against the noise curve A_0 of the same image at scale 0 (green), where i is the index of the image, b is the index of the bin, $(\hat{\mu}_{A_k,i,b}, \hat{\sigma}_{A_k,i,b})$ are the control points of the noise curve of the sub-scaled image, $(\hat{\mu}_{A_0,i,v}, \hat{\sigma}_{A_0,i,v})$ are the control points of the noise curve of the image at scale 0, and $\tilde{\sigma}_{A_0,i,b}$ is the standard deviation value projected from A_k into A_0.

Method	$k=1$	$k=2$	$k=3$
Ponomarenko	0.195	**0.738**	**1.684**
Percentile	**0.158**	0.922	3.416
PCA	0.627	2.407	4.997

TABLE 11. Evaluation $E^{(6)}_{A_0,A_k,b}$ for the raw images in Figure 2 depending on the scale k. The measurements are done for one of the green channels of the raw images.

PCA method needs 112000 samples/bin while Ponomarenko et al. and the Percentile need 42000 samples/bin (see Section 1.1 of Chapter 9). When the image is down-scaled it becomes too small for the PCA method to get an accurate estimation, since the minimum number of samples/bin is not attained and only one bin is used for the third and fourth scales.

4. Online demo

An online demo is available for all three methods in IPOL. The users can upload any image to measure its noise. The demo also offers several types of pre-uploaded images to test the algorithm:

- Raw images obtained by splitting the raw channels R, G_1, G_2, B and leaving out the G_2 channel. Then, an RGB image is formed by using the R, G_1, B channels. Since in the

raw image no gamma correction has been done yet, the values of the image are multiplied by 32 to increase their dynamics and screen visibility. The colors of these images are not quite adapted to human visualization, because no white balance has been applied to them.
- The JPEG versions of the same raw images, as they are encoded by the camera.
- Various JPEG images.
- High SNR raw images, down-scaled by eight with their color channels averaged, so that they are nearly noiseless. In the demo they are referred to as “no noise" images.

Once an image has been chosen the following parameters can be configured:

- **Treatment of groups (2×2) of equal pixels**. It allows to choose between ignoring the blocks that contain a group of four equal pixels in any channel (default), or using all the blocks (see Section 1.3 of Chapter 9) unconditionally.
- **Curve filter iterations**. It indicates the number of filtering iterations that are applied to filter the noise curve (see Section 1.2). Default: five iterations.
- **Number of bins**. It is the number of bins in the noise curve (see Section 1.1 of Chapter 9). If it is set to *automatic selection*, each bin will contain approximately 112000 samples/bin.
- **A and B noise parameters**. Add a simulated noise with variance $A + B\tilde{\mathbf{U}}$ is added to the input image. If $A = B = 0$ no noise will be added. If $B = 0$ white Gaussian noise with variance A will be added. Default: $A = B = 0$.

For the Ponomarenko method, the following parameters can be configured:

Percentile.: The possible values are 0.01%, 0.1%, 0.5% (default), 5%, 10%, 50%. It can be also configured to use the iterations of the original method in order to find the percentile K (see Algorithm 26).

Block size.: The possible choices are 3×3, 5×5, 7×7, 8×8 (default), 11×11, 15×15 and 21×21.

Mean of blocks computation.: permits to choose how the intensity associated to each bin is calculated. The possible choices are the average of the mean value of the pixels of the blocks that belong to the bin, or the median of the pixels of the blocks that belong to the bin (default).

Curve filter iterations.: It indicates the number of filtering iterations that are applied to filter the noise curve (see Section 1.2). Default: five iterations.

Treatment of groups (2×2) of equal pixels.: It allows to choose between ignoring the blocks that contain a group of four equal pixels in any channel (default), or using all the blocks unconditionally (see Section 1.3 of Chapter 9).

Number of bins.: It is the number of bins in the noise curve (see Section 1.1 of Chapter 9). The number of bins that are used depends on the size of the image when “automatic selection" is chosen. First, a nearest compatible size of the image is considered. For images whose

size is S_0, S_1 or S_2, the number of bins is given by dividing the total number of pixels of the image by 42000. Default: automatic selection.

A and B noise parameters.: Add a simulated noise with variance $A+B\tilde{\mathbf{U}}$ is added to the input image. If $A = B = 0$ no noise will be added. If $B = 0$ white Gaussian noise with variance A will be added. Default: $A = B = 0$.

For the Percentile method, the following parameters can be configured:

- Percentile. The possible values are 0.01%, 0.1%, 0.5% (default), 5%, 10% and 50%.
- Pre-filter operator. Operator $\mathbf{R}$ whose stencil $\mathbf{F}$ is used to convolve the image with. The possible operators are: Identity (no filter), Directional derivative, Laplace, Laplace (2 iterations), Laplace (3 iterations), Laplace (4 iterations), DCT with support 7×7 (default), DCT with support 5×5, DCT with support 3×3 and the filter of the article Fast Noise Variance Estimation [**4**].
- Block size. Possible choices: 3×3, 7×7, 8×8, 15×21 and 21×21 (default).
- Curve filter iterations.
- Treatment of groups (2×2) of equal pixels.
- Number of bins.
- A and B noise parameters.

For the PCA method, the following parameters can be configured:

- Treatment of groups (2×2) of equal pixels.
- Curve filter iterations.
- Number of bins.
- A and B noise parameters.

4.1. Subtraction of the quantization noise. In the online demo, all the images are encoded using 8 bits/pixel/channel. This adds a quantization error over the noise being estimated that must be subtracted. Indeed, the variance of a uniform random variable is $\sigma_q^2 = \int_{-\frac{1}{2}}^{\frac{1}{2}} (x-\bar{x})^2 \, \mathrm{d}x = \int_{-\frac{1}{2}}^{\frac{1}{2}} x^2 \, \mathrm{d}x = \left[\frac{x^3}{3}\right]_{-\frac{1}{2}}^{\frac{1}{2}} = \frac{1}{12}$. This is the variance of the quantization error that must be subtracted at each scale. The standard deviation of the noise is computed at each bin as the square root of the noise variance computed directly by the algorithm minus the variance of the variance of the (independent) quantization error. At each scale k the variance is divided by 4^k and thus the corrected standard deviation of the noise given by the demo is

$$\tilde{\sigma}_k = \sqrt{\hat{\sigma}_k^2 - \frac{\sigma_q^2}{4^k}} = \sqrt{\hat{\sigma}_k^2 - \frac{1}{4^k 12}}.$$

5. Complexity analysis of the algorithms

5.1. Common subalgorithms. This chapter discusses the algorithmic complexity of the sub-algorithms which are common to any method adapted to signal-dependent noise with the techniques explained in Chapter 9.

Algorithm 21 first executes the *argsort* (implemented with the Quicksort algorithm) operation with complexity $O(N \log N)$. The loop that iterates idx $= 0 \dots N$ just copies data in linear time $O(N)$. Therefore, Algorithm 21 is executed with complexity $O(N \log N)$. Algorithm 22, simply checks the condition at the *if* statement and computes simple arithmetic operations. Therefore, Algorithm 22 is executed in constant time $O(1)$. Algorithm 24 loops over the number of bins of the noise curve and inside the loop Algorithm 22 is called. Since Algorithm 22 is executed in constant time $O(1)$, Algorithm 24 has a linear complexity $O(B)$, being B the number of bins. Algorithm 25 loops over all possible pixels in the image (with the exception of the boundary of the image). The loop iterates through all the channels of the image and looks for groups of four connected pixels. Therefore, the inner loop is executed in linear time with the number of channels, $O(\text{num\_channels})$. It can actually be considered executed in constant time $O(1)$ once the number of channels have been fixed. The complexity of Algorithm 25 is given by its main loop, that is executed in linear time with complexity $O(N)$.

Algorithm 25 loops over all possible pixels in the image (with the exception of the boundary of the image). The loop iterates through all the channels of the image and looks for groups of four connected pixels. Therefore, the inner loop is executed in linear time with the number of channels, $O(\text{num\_channels})$. Since the number of channels is fixed and smaller than M, the complexity of Algorithm 25 is given by its main loop, that is executed in linear time with complexity $O(M)$.

5.2. Analysis specific to Ponomarenko et al. This complexity analysis is specific to the Ponomarenko et al. method.

The noise estimation procedure (Algorithm 26) first computes the DCT of all the $w \times w$ blocks in the image. Since the DCT is computed using the FFT algorithm, that has a complexity $O(M \log M)$. The loop that iterates seven times to get the optimal K executes the *argmin* operation that involves the Quick-sort algorithm and therefore it can be done with $O(n \log n)$ with $n = |\mathbf{V}_m^L|$. The computation of $\mathbf{V}_m^L = \frac{1}{\theta} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} [\mathbf{D}_m(i,j)]^2 \delta(i,j)$ and $\mathbf{V}^H(i,j) = \frac{1}{K} \sum_{k=0}^{K-1} \left[\mathbf{D}_{(k)}(i,j)\right]^2$ have linear complexity $O(M)$ and $O(K)$, namely. Therefore, Algorithm 26 has a linear complexity $O(M)$ since $M > K$. Algorithm 27 performs the same operations with the exception of substituting the seven iterations to get K with the product $k = pM$. Therefore, Algorithm 27 has also linear complexity $O(M)$.

5.3. Analysis specific to Percentile. This complexity analysis is specific to the Percentile method.

Algorithm 28 creates a matrix of $(N_x - 2b \times N_y - 2b)$ elements and then fills matrix $\mathbf{V}$ with $N_y - b - 1 \times N_x - b - 1$ values. Therefore, the complexity is linear, $O(N)$ with $N \sim N_x \times N_y$. The noise estimation procedure (Algorithm 29) computes the convolution $\tilde{\mathbf{U}} * \mathbf{F}$ in the Fourier domain using the FFT algorithm and then crops the result using Algorithm 28, that has linear complexity. The complexity of computing the convolution is $O(N \log N)$, where $N \sim (N_x - s + 1) \times (N_y - s + 1)$. Therefore, the complexity of cropping the convolution is $O(N \log N)$. Then, Algorithm 28 iterates M times through a loop that reads the blocks $\mathbf{W}_m^f$ and computes its variance. Since s is fixed,

the complexity of the loop is linear with the number of iterations, $O(M)$, where $M = (N_x - w - s + 1) \times (N_y - w - s + 1)$ is the number of overlapping blocks. After the execution of this loop, the SORTED function (implemented with the Quicksort algorithm) is called. Thus, the sorting operation has complexity $O(M \log M)$. Therefore, Algorithm 29 has complexity $O(M \log M)$.

5.4. Analysis specific to PCA. This complexity analysis is specific to the PCA method.

In Function *GetNextEstimate* (in Algorithm 30) the most computationally expensive part is the *ApplyPCA* function that is iterated several times until convergence is reached. The *ApplyPCA* function implies two steps:

- Computing the sample covariance matrix (45). If computed directly, it comprises about $|B(p)|M^2$ operations.
- Computing of the eigenvalues of the sample covariance matrix. Without any optimization, the number of operations is proportional to M^3. Again, since M is fixed, the cost of this step can be considered constant, $O(1)$.

With the optimization explained in Section 4 of Chapter 12, the sample covariance matrix can be expressed as

$$\frac{1}{|B(p) - 1|}\left(C_{B(p)} - \frac{1}{|B(p)|}\mathbf{c}_{B(p)}\mathbf{c}_{B(p)}^T\right)$$

where $C_X := \sum_{\mathbf{y}_i \in X} \mathbf{y}_i \mathbf{y}_i^T$ and $\mathbf{c}_X := \sum_{\mathbf{y}_i \in X} \mathbf{y}_i$.

Since the optimized algorithm first pre-computes the matrix $C_{B(1-j\Delta p)}$ and the vector $\mathbf{c}_{B(1-j\Delta p)}$ for $j = n-1, \ldots, 0$, if the worst case is considered ($j = 0$), then $B(1-j\Delta p) = B(1)$, that is, all the N blocks in the images would be used. The matrices and vectors are computed at most $\frac{1}{\Delta p} = \frac{1}{0.05} = 20$ times. To get the subsets of $BB(1 - j\Delta p)$ according to j it suffices to call just one the *argsort* function, implemented with the Quicksort algorithm with complexity $O(N \log N)$. Assuming that the matrices $C_{B(p)}$ and vectors $\mathbf{c}_{B(p)}$ and computed 20 times with the worst case $p = 1$, we have that $C_{B(p)} = C_{B(1)} = \sum_{\mathbf{y}_i \in B(1)} \mathbf{y}_i \mathbf{y}_i^T$ is executed with complexity $O(|B(1)|) = O(N)$. The same for $\mathbf{c}_{B(1)} := \sum_{\mathbf{y}_i \in B(1)} \mathbf{y}_i$, also with complexity $O(N)$. In summary, the computation of the sample covariance matrix consists on the execution of the *argsort* operation with complexity $N \log N$ and then looping at most 20 times through a loop that executes operations with complexity $O(N)$. Therefore, the complexity of Function *GetNextEstimate* (in Algorithm 30) is $N \log N$, being N the number of overlapping $M \times M$ blocks in the image.

6. Conclusion

The Ponomarenko et al. method gave the best results according to its RMSE, when adapted to the signal-dependent noise. The Percentile method achieves a similar accuracy, but is slightly worse.

Although the PCA method gave the best results when the noise is *a priori* known to be homoscedastic, when it is signal-dependent the method looses accuracy. It is easily explained if we take into account that to get a reliable estimation of the noise variance it needs at least 112000

samples/bin, whereas the Ponomarenko et al. and the Percentile methods only require 42000 samples/bin.

The strategy follow by the Ponomarenko et al. method (measuring the variance of the noise at high-frequency coefficients in a low quantile of the set of blocks whose variance measured at the low-frequencies is minimal) gives the best results and can be considered the state of the art in signal-dependent noise estimation.

Final conclusion

In the first part of this thesis, Noise Estimation, we have discussed several strategies to estimate the noise, from the simplest techniques, that simply consist on simulating white Gaussian noise and adding it to a noise-free image, signal-dependent noise estimators, to the most complete noise estimators, that permit to estimate a complex model on which the noise depends on both the intensity and the frequency.

For the homoscedastic white Gaussian noise estimators, Chapter 1 introduces the importance of the noise estimation for denoising and reviews the most representative classic methods. However, the simply homoscedastic noise model is only useful for raw images, where the noise is known to depend on the intensity, as explained in Chapter 2. This chapter also proposes a way to obtain a ground-truth noise curve for a particular camera configured with some ISO speed.

Chapter 3 showed the noise curves of the noise throughout the complete camera processing pipeline. This chapter made it clear that the signal-dependent noise model is not enough to measure the noise in most of the pictures, because after demosaicing the noise becomes highly correlated and signal-dependent noise estimators give underestimations in these conditions.

Therefore, a new noise estimation algorithm was developed and presented in Chapter 8. This new multiscale method is able to accurately measure the variance of highly correlated noise, even in the case of JPEG-compressed images, which have undergone demosaicing, white balance, gamma correction, and lossy JPEG-encoding. We introduced a new method to compute the difference between to noise blocks, the *sparse distance*, which is robust to noise and compares only the geometry of the blocks, without the interference of the noise. The comparison between the spatial and the temporal noise curves demonstrates that indeed the method is valid. Other tests also proved the method valid: a comparison of the noise estimates with the values expected after filtering the image with a low-pass filter and the observation of denoising results.

In the second part of this thesis, Patch Denoising, we propose in Chapter 8 a new denoising algorithm. It uses the noise estimation of the method presented in Chapter 4 to denoise each patch of the noisy image. With this fully automatic tool, users input a noisy image with unknown noise model, the noise is characterized, the noise is removed at each scale, and finally a denoised image is returned to the users without any other interaction. We called this tool the “Noise Clinic", as a metaphor of a real clinic, where the patients enter with some unknown problem, which is diagnosed, and sometimes cured.

Definitely, this thesis studied in detail the problem of how to estimate noise to afterwards denoise an image, using only the information of the noisy image (blind noise estimation and blind

denoising) and assuming only SFD noise. This minimal assumption of SFD permits to denoise most of the images obtained with digital cameras, even when they are JPEG-compressed.

The third part of the thesis, Reproducible Research Contributions, presents three of the methods that were adapted to measure signal-dependent noise, along with some proposed general techniques that permit to reliably adapt almost any block-based noise estimator to signal-dependent noise. The articles presented in this third part were published in the Image Processing On Line (IPOL) journal.

Some problems are still open and invite to further research, as the case of images with position-dependent noise, adapting the size of the scanning window to the characteristics of the image, or the search of optimal functions to compute the similarity between two noisy patches.

Bibliography

[1] M. Lebrun, M. Colom, A. Buades, and J.M. Morel, "Secrets of image denoising cuisine", *Acta Numerica*, vol. 21, pp. 475–576, 2012, doi: 10.1017/S0962492912000062.

[2] S.I. Olsen, "Estimation of noise in images: An evaluation", *CVGIP: Graphical Models and Image Processing*, vol. 55, no. 4, pp. 319–323, 1993.

[3] S. Pyatykh, Hesser J., and Zheng L., "Image noise level estimation by principal component analysis", *IEEE Transactions on Image Processing*, 2012, doi: 10.1109/TIP.2012.2221728.

[4] J. Immerkaer, "Fast noise variance estimation", *Computer Vision and Image Understanding*, vol. 64, no. 2, pp. 300–302, 1996, doi: 10.1006/cviu.1996.0060.

[5] K. Rank, M. Lendl, and R. Unbehauen, "Estimation of image noise variance", in *Vision, Image and Signal Processing*. IET, 1999, doi: 10.1049/ip-vis:19990238, vol. 146, pp. 80–84.

[6] P. Meer, J.M. Jolion, and A. Rosenfeld, "A fast parallel algorithm for blind estimation of noise variance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 216–223, 1990, doi: 10.1109/34.44408.

[7] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical poissonian-gaussian noise modeling and fitting for single-image raw-data", *IEEE Transactions on Image Processing*, vol. 17, no. 10, pp. 1737–1754, 2008, doi: 10.1109/TIP.2008.2001399.

[8] R. A. Boie and I. J. Cox, "An analysis of camera noise", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 6, pp. 671–674, 1992.

[9] M. Colom, A. Buades, and J.M. Morel, "Nonparametric noise estimation method for raw images", *Journal of the Optical Society of America A*, vol. 31, 2014. doi: 10.1364/JOSAA.31.000863.

[10] M. Colom, M. Lebrun, A. Buades, and J.M. Morel, "Multiscale estimation of intensity and frequency dependent noise", *IEEE Transactions On Image Processing*, 2014, Submitted.

[11] M. Raphan and E.P. Simoncelli, "An empirical bayesian interpretation and generalization of nl-means", Tech. Rep., Technical Report TR2010-934, Computer Science Technical Report, Courant Inst. of Mathematical Sciences, New York University, 2010.

[12] D. Van De Ville and M. Kocher, "Sure-based non-local means", *Signal Processing Letters, IEEE*, vol. 16, no. 11, pp. 973–976, 2009.

[13] C.A. Deledalle, V. Duval, and J. Salmon, "Non-local methods with shape-adaptive patches (NLM-SAP)", *Journal of Mathematical Imaging and Vision*, pp. 1–18, 2010.

[14] J. Portilla, "Blind non-white noise removal in images using Gaussian scale mixtures in the wavelet domain", *Benelux Signal Processing Symposium*, 2004.

[15] M. Colom and A. Buades, "Analysis and Extension of the Ponomarenko et al. Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, vol. 3, pp. 173–197, 2013. doi: 10.5201/ipol.2013.45.

[16] N.N. Ponomarenko, V.V. Lukin, M.S. Zriakhov, A. Kaarna, and J.T. Astola, "An automatic approach to lossy compression of AVIRIS images", in *IEEE International Geoscience and Remote Sensing Symposium*, 2007, doi: 10.1109/IGARSS.2007.4422833, pp. 472–475.

[17] M. Colom and A. Buades, "Analysis and Extension of the Percentile Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, vol. 2013, pp. 322–349, 2013. doi: 10.5201/ipol.2013.90.

[18] M. Colom and A. Buades, "Analysis and Extension of the PCA Method, Estimating a Noise Curve from a Single Image", *Image Processing On Line*, 2014, Submitted.

[19] M. Colom, G. Facciolo, M. Lebrun, N. Pierazzo, M. Rais, Y. Wang, and J.M. Morel, "A mathematical perspective of image denoising", *International Congress of Mathematicians*, 2014, To appear.

[20] M. Colom, M. Lebrun, A. Buades, and J.M. Morel, "A non-parametric approach for the estimation of intensity-frequency dependent noise", *IEEE International Conference on Image Processing*, 2014, Submitted.

[21] M. Lebrun, M. Colom, and J.M. Morel, "The Noise Clinic: a universal blind denoising algorithm", *IEEE International Conference on Image Processing*, 2014, Submitted.

[22] M. Lebrun, M. Colom, and J.M. Morel, "Multiscale image blind denoising", *IEEE Transactions On Image Processing*, 2014, Submitted.

[23] C. Chevalier, G. Roman, and J.N. Niepce, *Guide du photographe.*, C. Chevalier, 1854.

[24] A. Buades, Y. Lou, J.M. Morel, and Z. Tang, "A note on multi-image denoising", in *Proceedings of the International Workshop on Local and Non-Local Approximation in Image Processing*. IEEE, 2009, doi: 10.1109/LNLA.2009.5278408, pp. 1–15.

[25] C.E. Shannon, "A mathematical theory of communication", *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[26] S.R.J.L. HARRIS, "Image evaluation and restoration", *JOSA*, vol. 56, no. 5, pp. 569–570, 1966.

[27] W.H. Richardson, "Bayesian-based iterative method of image restoration", *JOSA*, vol. 62, no. 1, pp. 55–59, 1972.

[28] L. Yaroslavsky and M. Eden, "Fundamentals of Digital Optics", 2003.

[29] Y. Meyer, "Wavelets-algorithms and applications", *Wavelets-Algorithms and applications Society for Industrial and Applied Mathematics Translation., 142 p.*, vol. 1, 1993.

[30] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions and the bayesian restoration of images", *IEEE Pat. Anal. Mach. Intell.*, vol. 6, pp. 721–741, 1984.

[31] A. Efros and T. Leung, "Texture synthesis by non parametric sampling", in *Proc. Int. Conf. Computer Vision*, 1999, vol. 2, pp. 1033–1038.

[32] A. Buades, B. Coll, J.M. Morel, et al., "A review of image denoising algorithms, with a new one", *Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490–530, 2006.

[33] J. Portilla, V. Strela, M.J. Wainwright, and E.P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain", *Image Processing, IEEE Transactions on*, vol. 12, no. 11, pp. 1338–1351, 2003.

[34] A. Levin and B. Nadler, "Natural image denoising: Optimality and inherent bounds", in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 2833–2840.

[35] P. Chatterjee and P. Milanfar, "Is denoising dead?", *Image Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 895–911, 2010.

[36] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3D transform-domain collaborative filtering", *IEEE Transactions on image processing*, vol. 16, pp. 2007, 2007.

[37] J. Portilla, "Full blind denoising through noise covariance estimation using Gaussian scale mixtures in the wavelet domain", *Image Processing, 2004. ICIP'04. 2004 International Conference on*, vol. 2, pp. 1217–1220, 2004.

[38] B. Rajaei, "An analysis and improvement of the BLS-GSM denoising method", *Image Processing On Line*, vol. 2013, 2013, Preprint.

[39] A.B. Lee, K.S. Pedersen, and D. Mumford, "The nonlinear statistics of high-contrast patches in natural images", *International journal of Computer Vision*, vol. 54, no. 1, pp. 83–103, 2003.

[40] A. Foi, S. Alenius, and K. Katkovnik, V.and Egiazarian, "Noise measurement for raw-data of digital imaging sensors by automatic segmentation of non-uniform targets", *IEEE Sensors journal*, vol. 7, no. 10, pp. 1456–1461, 2007, doi: 10.1109/JSEN.2007.904864.

[41] Siméon-Denis Poisson, "Recherches sur la probabilité des jugements en criminelle et en matière civile, précédées des règles générales du calcul des probabilités", *Bachelier, Paris*, 1837.

[42] F. J. Anscombe, "The transformation of Poisson, binomial and negative-binomial data", *Biometrika*, vol. 35, no. 3, pp. 246–254, 1948. doi: 10.2307/2332343.

[43] M. Makitalo and A. Foi, "Optimal inversion of the Anscombe transformation in low-count Poisson image denoising", *Image Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 99–109, 2011. doi: 10.1109/TIP.2010.2056693.

[44] Robert D. Nowak and Richard G. Baraniuk, "Wavelet-domain filtering for photon imaging systems", *IEEE Transactions on Image Processing*, vol. 8, no. 5, pp. 666–678, 1997. doi: 10.1109/83.760334.

[45] E. D. Kolaczyk, "Wavelet shrinkage estimation of certain Poisson intensity signals using corrected thresholds", *Statist. Sin.*, vol. 9, pp. 119–135, 1999.

[46] Stamatios Lefkimmiatis, Petros Maragos, and George Papandreou, "Bayesian inference on multiscale models for poisson intensity estimation: Application to photo-limited image denoising", *IEEE Transactions on Image Processing*, vol. 18, no. 8, pp. 1724–1741, 2009.

[47] C.A. Deledalle, L. Denis, and F. Tupin, "Nl-insar: Nonlocal interferogram estimation", *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 4, pp. 1441–1452, 2011.

[48] C.A. Deledalle, F. Tupin, and L. Denis, "Poisson nl means: Unsupervised non local means for poisson noise", in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 801–804.

[49] C.A. Deledalle, F. Tupin, and L. Denis, "Polarimetric sar estimation based on non-local means", in *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*. IEEE, 2010, pp. 2515–2518.

[50] A. Foi, "Noise estimation and removal in mr imaging: The variance-stabilization approach", in *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*. IEEE, 2011, pp. 1809–1814.

[51] D. Zoran and Y. Weiss, "Scale invariance and noise in natural images", in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2209–2216.

[52] C. Liu, W. Freeman, R. Szeliski, and S. Kang, "Automatic estimation and removal of noise from a single image", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 299–314, February 2008.

[53] C. Liu, W.T. Freeman, R. Szeliski, and S.B. Kang, "Noise estimation from a single image", in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Ieee, 2006, vol. 1, pp. 901–908.

[54] Aram Danielyan and Alessandro Foi, "Noise variance estimation in nonlocal transform domain", in *Local and Non-Local Approximation in Image Processing, 2009. LNLA 2009. International Workshop on*. IEEE, 2009, pp. 41–45.

[55] S.I. Olsen, "Estimation of noise in images: an evaluation", *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 55, no. 4, pp. 319–323, July 1993. doi: 10.1006/cgip.1993.1022.

[56] J.S. Lee, "Refined filtering of image noise using local statistics", *Computer graphics and image processing*, vol. 15, no. 4, pp. 380–389, 1981.

[57] G.A. Mastin, "Adaptive filters for digital image noise smoothing: An evaluation", *Computer Vision, Graphics, and Image Processing*, vol. 31, no. 1, pp. 103–121, 1985, doi: 10.1016/S0734-189X(85)80078-5.

[58] R. Bracho and AC Sanderson, "Segmentation of images based on intensity gradient information", in *Proc. IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, 1985, pp. 19–23.

[59] H. Voorhees and T. Poggio, "Detecting textons and texture boundaries in natural images", in *Proceedings of the First International Conference on Computer Vision, London*. IEEE, Washington, DC, 1987, pp. 250–258.

[60] J. S Lee and K. Hoppel, "Noise modelling and estimation of remotely-sensed images", *Proceedings of the International Geoscience and Remote Sensing Symposium*, vol. 2, pp. 1005–1008, 1989. doi: 10.1109/IGARSS.1989.579061.

[61] D.L. Donoho and I.M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage", *journal of the American Statistical Association*, pp. 1200–1224, 1995, doi: 10.2307/2291512.

[62] P. Milanfar, "A tour of modern image filtering: New insights and methods, both practical and theoretical", *Signal Processing Magazine*, vol. 30, no. 1, pp. 106–128, 2013.

[63] Hossein Rabbani, Milan Sonka, and Michael D Abramoff, "Optical coherence tomography noise reduction using anisotropic local bivariate", *International journal of Biomedical Imaging*, vol. 3, no. 225, 2011.

[64] J. Schmitt, J.L. Starck, J.M. Casandjian, J. Fadili, and I. Grenier, "Multichannel poisson denoising and deconvolution on the sphere: Application to the fermi gamma ray space telescope", *Astronomy and Astrophysics*, vol. 546, pp. 10, 2012.

[65] Florian Luisier, Thierry Blu, and Michael Unser, "Image denoising in mixed poisson–gaussian noise", *Image Processing, IEEE Transactions on*, vol. 20, no. 3, pp. 696–708, 2011.

[66] F-X Dupé, Jalal M Fadili, and J-L Starck, "A proximal iteration for deconvolving poisson noisy images using sparse representations", *Image Processing, IEEE Transactions on*, vol. 18, no. 2, pp. 310–321, 2009.

[67] Hossein Rabbani, Reza Nezafat, and Saeed Gazor, "Wavelet-domain medical image denoising using bivariate laplacian mixture model", *Biomedical Engineering, IEEE Transactions on*, vol. 56, no. 12, pp. 2826–2837, 2009.

[68] Bo Zhang, Jalal M Fadili, and Jean-Luc Starck, "Wavelets, ridgelets, and curvelets for poisson noise removal", *Image Processing, IEEE Transactions on*, vol. 17, no. 7, pp. 1093–1108, 2008.

[69] Hossein Rabbani and Saeed Gazor, "Local probability distribution of natural signals in sparse domains", *International journal of Adaptive Control and Signal Processing*, 2013.

[70] Joseph Salmon Charles-Alban Deledalle and Arnak Dalalyan, "Image denoising with patch based pca: local versus global", in *Proceedings of the British Machine Vision Conference*. 2011. doi: 10.5244/C.25.25, pp. 25.1–25.10, BMVA Press.

[71] V.I.A. Katkovnik, V. Katkovnik, K. Egiazarian, and J. Astola, *Local approximation techniques in signal and image processing*, Society of Photo Optical, 2006.

[72] N.N. Ponomarenko, V.V. Lukin, S.K. Abramov, K.O. Egiazarian, and J.T. Astola, "Blind evaluation of additive noise variance in textured images by nonlinear processing of block DCT coefficients", in *Proceedings of the International Society for Optics and Photonics. Electronic Imaging, Image Processing: Algorithms and Systems II*, 2003, doi: 10.1117/12.477717, vol. 5014, pp. 178–189.

[73] D.L. Donoho and J.M. JOHNSTONE, "Ideal spatial adaptation by wavelet shrinkage", *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[74] D.L. Donoho, I.M. Johnstone, et al., "Adapting to unknown smoothness via wavelet shrinkage", *journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[75] M.L. Uss, B. Vozel, V. V. Lukin, and K. Chehdi, "Image informative maps for component-wise estimating parameters of signal-dependent noise", *journal of Electronic Imaging*, vol. 22, no. 1, pp. 013019–013019, 2013.

[76] Mikhail Uss, Benoit Vozel, V Lukin, S Abramov, Igor Baryshev, and Kacem Chehdi, "Image informative maps for estimating noise standard deviation and texture parameters", *EURASIP journal on Advances in Signal Processing*, vol. 2011, no. 2011, pp. 806516, 2011.

[77] Poorvi L Vora, Joyce E Farrell, Jerome D Tietz, and David H Brainard, "Linear models for digital cameras", in *IS & T Annual Conference*. The Society for Imaging Science and Technology, 1997, pp. 377–382.

[78] M. Lebrun, A. Buades, and J.M. Morel, "A nonlocal bayesian image denoising algorithm", *SIAM journal on Imaging Sciences*, vol. 6, no. 3, pp. 1665–1688, 2013.

[79] G. K. Wallace, "The JPEG still picture compression standard", *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[80] A. Buades, B. Coll, J. M. Morel, and C. Sbert, "Self-similarity driven demosaicking", *Image Processing On Line*, vol. 2011, 2011. doi: 10.5201/ipol.2011.bcms-ssdd.

[81] J. F. Hamilton Jr. and J. E. Adams Jr., "Adaptive color plan interpolation in single sensor color electronic camera", May 13 1997, US Patent 5,629,734.

[82] R. Neelamani, R. De Queiroz, Z. Fan, S. Dash, and R. G. Baraniuk, "JPEG compression history estimation for color images", *Image Processing, IEEE Transactions on*, vol. 15, no. 6, pp. 1365–1378, 2006.

[83] N. N. Ponomarenko, V. V. Lukin, K. O. Egiazarian, and J. T. Astola, "A method for blind estimation of spatially correlated noise characteristics", in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2010, pp. 753208–753208.

[84] M. Lebrun, A. Buades, and J.M. Morel, "Implementation of the "non-local Bayes" (NL-Bayes) image denoising algorithm", *Image Processing On Line*, vol. 2013, pp. 1–42, 2013.

[85] F. J. Anscombe, "The transformation of Poisson, binomial and negative-binomial data", *Biometrika*, vol. 35, no. 3, pp. 246–254, 1948.

[86] A. Buades, B. Coll, and J.M. Morel, "A non local algorithm for image denoising", *IEEE Computer Vision and Pattern Recognition*, vol. 2, pp. 60–65, 2005.

[87] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation", *Signal Processing, IEEE Transactions on*, vol. 54, no. 11, pp. 4311–4322, 2006.

[88] Pierre Brémaud, *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, vol. 31, springer verlag, 1999.

[89] P. Chatterjee and P. Milanfar, "Patch-based near-optimal image denoising.", *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 2011.

[90] Daniel Zoran and Yair Weiss, "From learning models of natural image patches to whole image restoration", in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 479–486.

[91] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping", *Pattern Recognition*, vol. 43, no. 4, pp. 1531–1549, 2010.

[92] R.R. Coifman and D.L. Donoho, "Translation-invariant de-noising", *Lecture Notes In Statistics*, pp. 125–125, 1995.

[93] A. Nemirovski, "Topics in non-parametric statistics", *Lectures on probability theory and statistics (Saint-Flour, 1998)*, vol. 1738, pp. 85–277, 2000.

[94] O.G. Guleryuz, "Weighted averaging for denoising with overcomplete dictionaries", *Image Processing, IEEE Transactions on*, vol. 16, no. 12, pp. 3020–3034, 2007.

[95] Guillermo Sapiro Guoshen Yu, "DCT image denoising: a simple and effective image denoising algorithm", *Image Processing On Line*, 2011. doi: 10.5201/ipol.2011.ys-dct.

[96] C. Kervrann and J. Boulanger, "Local Adaptivity to Variable Smoothness for Exemplar-Based Image Regularization and Representation", *International Journal of Computer Vision*, vol. 79, no. 1, pp. 45–69, 2008.

[97] M. Raphan and E.P. Simoncelli, "Learning to be bayesian without supervision", *Advances in neural information processing systems*, vol. 19, pp. 1145, 2007.

[98] V. Duval, J.F. Aujol, and Y. Gousseau, "A bias-variance approach for the nonlocal means", *SIAM Journal on Imaging Sciences*, vol. 4, pp. 760, 2011.

[99] 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.

[100] J.W. Tukey, "Exploratory data analysis. 1977", *Massachusetts: Addison-Wesley*, 1976.

[101] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "Using geometry and iterated refinement for inverse problems (1): Total variation based image restoration", *Department of Mathematics, UCLA, LA, CA*, vol. 90095, pp. 04–13, 2004.

[102] Alexei A Efros and Thomas K Leung, "Texture synthesis by non-parametric sampling", in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. IEEE, 1999, vol. 2, pp. 1033–1038.

[103] J.M. Morel, B. Coll, and A. Buades, "Image data processing method by reducing image noise, and camera integrating means for implementing said method", June 10 2010, US Patent App. 11/579,380.

[104] J.S. De Bonet, "Noise reduction through detection of signal redundancy", *Rethinking artificial intelligence*, 1997.

[105] T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and MJ Weinberger, "Universal discrete denoising: Known channel", *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 5–28, 2005.

[106] E. Ordentlich, G. Seroussi, S. Verdu, M. Weinberger, and T. Weissman, "A discrete universal denoiser and its application to binary images", in *International Conference on Image Processing*, 2003, vol. 1.

[107] S.P. Awate and R.T. Whitaker, "Unsupervised, information-theoretic, adaptive image filtering for image restoration", *IEEE Trans. PAMI*, vol. 28, no. 3, pp. 364–376, 2006.

[108] G. Yu and G. Sapiro, "DCT image denoising: a simple and effective image denoising algorithm", *Image Processing On Line*, vol. 2011, 2011.

[109] P. Getreuer, "Rudin-osher-fatemi total variation denoising using split bregman", *Image Processing On Line*, vol. 2012, 2012.

[110] M. Lebrun and A. Leclaire, "An implementation and detailed analysis of the k-svd image denoising algorithm", *Image Processing On Line*, vol. 2012, 2012.

[111] A. Buades, B. Coll, and J.M. Morel, "Non-local means denoising", *Image Processing On Line*, vol. 2011, 2011.

[112] M. Lebrun, "An analysis and implementation of the BM3D image denoising method", *Image Processing On Line*, vol. 2012, 2012.

[113] T. Rabie, "Robust estimation approach for blind denoising", *IEEE Transactions on Image Processing*, vol. 14, no. 11, pp. 1755–1765, November 2005.

[114] S. Gabarda and G. Cristóbal, "The generalized Rényi image entropy as a noise indicator", *Noise and Fluctuations in Photonics, Quantum Optics, and Communications*, vol. 6603, 2007. doi: 10.1117/12.725086.

[115] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration", *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[116] J. Mairal, G. Sapiro, and M. Elad, "Learning multiscale sparse representations for image and video restoration", *SIAM Multiscale Modeling and Simulation*, vol. 7, no. 1, pp. 214–241, 2008.

[117] L.P. Yaroslavsky, K.O. Egiazarian, and J.T. Astola, "Transform domain image restoration methods: review, comparison, and interpretation", in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, May 2001, vol. 4304, pp. 155–169.

[118] L.P. Yaroslavsky, "Local adaptive image restoration and enhancement with the use of DFT and DCT in a running window", in *Proceedings of SPIE*, 1996, vol. 2825, pp. 2–13.

[119] J.L. Starck, E.J. Candès, and D.L. Donoho, "The curvelet transform for image denoising", *IEEE Transactions on Image Processing*, vol. 11, pp. 670–684, 2002.

[120] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms", *Phys. D*, vol. 60, pp. 259–268, 1992.

[121] R. R. Coifman and D. L. Donoho, *Translation-invariant de-noising*, vol. 103, Springer New York, 1995. doi: 10.1007/978-1-4612-2544-7_9.

[122] A. Buades, B. Coll, J.M. Morel, and C. Sbert, "Self-similarity Driven Demosaicking", *Image Processing On Line*, vol. 1, 2011. doi: 10.5201/ipol.2011.bcms-ssdd.

[123] J.W. Cooley and J.W. Tukey, "An algorithm for the machine calculation of complex fourier series", *Mathematics of Computation*, pp. 297–301, 1965, doi: 10.2307/2003354.

[124] M. Black and G. Sapiro, "Edges as outliers: Anisotropic smoothing using local image statistics", in *Scale-Space Theories in Computer Vision*, Mads Nielsen, Peter Johansen, Ole Olsen, and Joachim Weickert, Eds., vol. 1682 of *Lecture Notes in Computer Science*, pp. 259–270. Springer Berlin / Heidelberg, 1999, doi: 10.1007/3-540-48236-9_23.

[125] S. A. Gershgorin, "Uber die abgrenzung der eigenwerte einer matrix", *Izvestija Akademii Nauk SSSR*, pp. 749–754, 1931.