
ON NORMALIZATION-EQUIVARIANCE PROPERTIES OF SUPERVISED AND UNSUPERVISED DENOISING METHODS: A SURVEY

Sébastien Herbreteau, Charles Kervrann
Centre Inria de l'Université de Rennes, France
{sebastien.herbreteau, charles.kervrann}@inria.fr

ABSTRACT

Image denoising is probably the oldest and still one of the most active research topic in image processing. Many methodological concepts have been introduced in the past decades and have improved performances significantly in recent years, especially with the emergence of convolutional neural networks and supervised deep learning. In this paper, we propose a survey of guided tour of supervised and unsupervised learning methods for image denoising, classifying the main principles elaborated during this evolution, with a particular concern given to recent developments in supervised learning. It is conceived as a tutorial organizing in a comprehensive framework current approaches. We give insights on the rationales and limitations of the most performant methods in the literature, and we highlight the common features between many of them. Finally, we focus on the normalization equivariance properties that is surprisingly not guaranteed with most of supervised methods. It is of paramount importance that intensity shifting or scaling applied to the input image results in a corresponding change in the denoiser output.

1 Introduction

In the realm of digital image acquisition, two significant independent types of noise can degrade the quality of captured images [50, 108, 11, 143, 42]. On the first hand, shot noise stems from the inherent random nature of light. Classically, the number of photons detected by an image sensor is described by a Poisson distribution whose parameter is proportional to both the true signal value and the time exposure. On the other hand, read noise is typically independent of the light intensity hitting the sensor and is introduced during the process of converting the analog signal from the camera sensor into a digital representation. Read noise is caused by various factors, including the electronic components of the sensor, circuitry, and analog-to-digital conversion process. Traditionally, this type of noise is mathematically modeled by an additive white Gaussian noise, justified by the application of the central limit theorem.

Therefore, image noise is commonly described by a mixed Poisson-Gaussian model. Formally, representing a grayscale image with n pixels by a vector of \mathbb{R}^n where each entry encodes the pixel intensity, the noise model is:

$$y \sim a\mathcal{P}(x/a) + \mathcal{N}(\mathbf{0}_n, bI_n), \quad (1)$$

where $y \in \mathbb{R}^n$ is the observed noisy image, $x \in \mathbb{R}^n$ is the noise-free image (true signal), and $a, b \in \mathbb{R}_*^+$ are the parameters relative to shot and read noise, respectively, depending in particular on the acquisition system and on the exposure time. A widespread simpler alternative to the mixed Poisson-Gaussian model (1) is the additive white Gaussian noise (AWGN) model:

$$y \sim \mathcal{N}(x, \sigma^2 I_n), \quad (2)$$

where σ^2 is the signal-independent variance of the noise. The formulation (2) can be seen as an approximation of (1) where the signal-dependent shot noise is neglected. Although it may seem to be a limitation of this model, formulation (1) actually transposes to (2) when using a variance-stabilizing transformation (VST) such as the Anscombe transform [128] and its generalizations [127, 9, 94] that amount to applying per-pixel nonlinearities that effectively reduce the signal dependence. Ultimately, due to its mathematical convenience, the AWGN model is the most widely-used one.

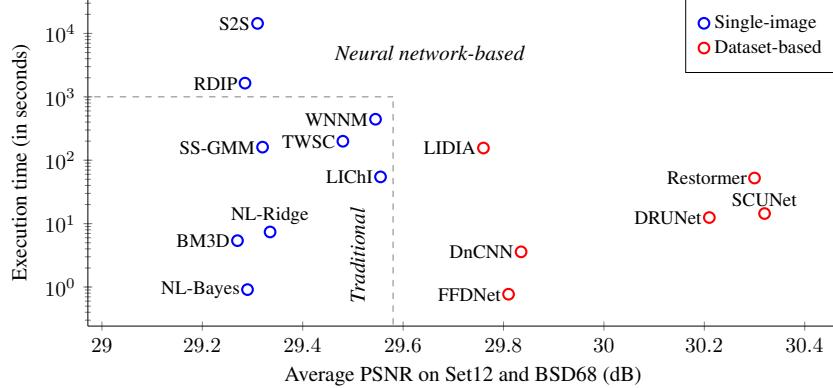


Figure 1: Execution time on CPU for images of size 512×512 v.s the average PSNR results on the union of Set12 and BSD68 datasets for Gaussian noise with $\sigma = 25$ for popular methods.

From the noisy observation y , which follows either (1) or (2) but also any other, possibly unknown, noise distribution, the aim of image denoising is to design a method for estimating the original unknown signal x as faithfully as possible [39]. This amounts to identifying a function $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ such that a noisy observation y can be mapped to a satisfactory estimate of x , i.e. $f(y) \approx x$. Over the years, a rich variety of strategies, tools and theories have emerged to address the issue of image denoising at the intersection of statistics, signal processing, optimization and functional analysis. The performance and the limitations of resulting single-shot methods are generally well understood. But this field has been recently immensely influenced by the development of machine learning techniques and artificial intelligence. Viewing denoising as a simple regression problem, this task ultimately amounts to learn to match the corrupted image to its source. The very best methods in image denoising leverage deep neural networks which are trained on large external datasets consisting of clean/noisy image pairs (see Fig. 1). However, though fast and efficient, these supervised networks suffer from their lack of interpretability and usually have fewer good mathematical properties than their conventional counterparts. Therefore, it is of paramount importance to examine several mathematical properties which are desirable in image denoising, especially the so-called normalization-equivariance, which ensures that any change of the input noisy image, whether by shifting or scaling, results in a corresponding change in the denoising response. While this property is partially fulfilled by single-shot methods, current deep neural networks surprisingly do not guarantee such a property, which can be detrimental in many situations (source of misinterpretation in critical applications).

The remainder of the paper is organized as follows. In Section II, we take the reader on a guided tour of supervised learning methods for image denoising. In Section III, we review the unsupervised denoising methods and focus on the most performant methods. In Section IV, we study the normalization equivariance (NE) properties of the reviewed methods and provide cues so that NE holds by design.

2 Review of supervised learning methods

Starting from a general framework based on empirical risk minimization, we present the three main classes of parameterized functions, also known as neural network architectures in artificial intelligence. For each architecture, we study a popular state-of-the-art representative for image denoising. Next, we address the issue of finding the best function for denoising among a given family of parametric functions, more commonly known as parameter training. Finally, we study the special case of weakly supervised learning, which does not require noise-free images for training.

2.1 Principle of supervised learning

The holy grail in image denoising is to find a universal function f that, given a noisy observation $y \in \mathcal{Y}$, maps the corresponding noise-free image $x \in \mathcal{X}$. Unfortunately, such a function is purely hypothetical as image denoising is an ill-posed inverse problem in the sense that the mere experimental observation of a noisy image is not enough to perfectly determine the unknown true image. In order to narrow down the space of possibilities and arrive at a unique solution, a risk minimization point of view has been widely adopted in past years. More precisely, let us define the risk of function f as:

$$\mathcal{R}(f) = \mathbb{E}_{x,y} \|f(y) - x\|, \quad (3)$$

where $(x, y) \in \mathcal{X} \times \mathcal{Y}$ model all possible pairs of clean/noisy *natural* images, with the associated joint probability distribution $p(x, y)$. One wants ideally to find:

$$f^* \in \arg \min_f \mathcal{R}(f). \quad (4)$$

Usually the squared ℓ_2 norm or the ℓ_1 norm are used to measure closeness in (3) and are examples of so-called loss functions. In the case of the squared ℓ_2 norm, $f^*(y)$ is nothing else than the minimum mean square error (MMSE) estimator. Restricting f to be a member of a sufficiently general class of parameterized functions (f_θ), the problem (4) transposes to the following parameter optimization problem:

$$\theta^* \in \arg \min_\theta \mathcal{R}(f_\theta). \quad (5)$$

In general, as the joint distribution $p(x, y)$ is unknown, an empirical sample consisting of a finite number S of pairs of clean/noisy images, called *training set*, is used as a surrogate. The empirical risk is then defined as:

$$\mathcal{R}_{\text{emp}}(f_\theta) = \frac{1}{S} \sum_{s=1}^S \|f_\theta(y_s) - x_s\|. \quad (6)$$

Note that, depending on the standard chosen to measure proximity, minimizing the empirical risk (6) with respect to θ actually amounts to minimizing the mean square error (MSE) or the mean absolute error (MAE), in most cases, over a finite set of image pairs. This approach is said to be supervised in the sense that it relies on an external dataset of clean/noisy pairs of images on which the model is optimized. However, minimizing the risk on a finite subset of $\mathcal{X} \times \mathcal{Y}$, designating all possible pairs of clean/noisy images, cannot guarantee that the model will provide also good performance on unseen samples. Indeed, a function f_θ that presents a low empirical risk (6) may sometimes be far from optimality with regard to the true risk defined in (3). This well-known phenomenon is called overfitting and may basically occur either when the *training set* is not enough representative of the true distribution $p(x, y)$ of data in $\mathcal{X} \times \mathcal{Y}$, or when (f_θ) is over-parameterized such that it may match too closely or even exactly the *training set* (in this latter case, we say that the function interpolates the data points). In that respect, optimization needs to be differentiated from machine learning which is precisely concerned with minimizing the loss on samples outside the *training set*.

Machine learning theory states that a necessary condition for good generalization beyond the *training set*, is that this latter must provide sufficiently diverse, abundant and representative examples of $\mathcal{X} \times \mathcal{Y}$. Collecting high-quality *training sets* may be very challenging in some situations, but the success of supervised learning depends on it, and image denoising is no exception [11, 143, 17, 144, 1]. In order to assess the generalization capabilities of the learned model, a *test set* is used, consisting of a finite subset drawn randomly from $\mathcal{X} \times \mathcal{Y}$ and strictly disjoint from the *training set* on which optimization is done. The performance of the model on the *test set* is an imperfect measure of its generalization as there exists no finite subset of $\mathcal{X} \times \mathcal{Y}$ that represents perfectly the true distribution $p(x, y)$ but it is the only metric at our disposal.

From this very general paradigm, several issues need be addressed. First of all, the choice of the class of parameterized functions (f_θ) is an important part of the success of supervised machine learning. The chosen class must indeed be sufficiently large for a chance to contain high-performance functions for the denoising task; but at the same time, oversized classes may lead to an overfitted model. Then, once the parameterized class of functions has been chosen, solving the inherent optimization problem defined in (5) can be particularly cumbersome and one would like to be able to rely on efficient and general heuristics to deal with it. Finally, the quality of the *training set* is crucial but, in numerous contexts, sufficiently many diverse and abundant noise-free images are unfortunately not available. A recent line of research proposes to relax the need for clean images by adopting a so-called *weakly* supervised learning approach. Below, we show how all these issues are commonly addressed in the case of image denoising.

2.2 Classes of parameterized functions

In this section, we review the most three major classes of parameterized functions (f_θ) that were successfully experimented in image denoising. All of them are in fact subcategories of the general class of parameterized functions that are called (*improperly?*) “artificial neural networks”.

2.2.1 Multi-layer perceptron (MLP)

Historically, the first class of parameterized functions used in supervised machine learning is the multi-layer perceptron (MLP) proposed by F. Rosenblatt [116]. The seminal work from H. C. Burger *et al.* [14] constitutes the first successful attempt of learning the mapping from a noisy image to its corresponding noise-free one with such an artificial neural network. For the first time in the field of image denoising, learning approaches have been compared favorably with unsupervised (*a.k.a* non-learning) methods, without making any assumptions about natural images or about noise type.

Mathematical description Formally, a multi-layer perceptron with $L \geq 1$ hidden layers is a nonlinear function $f_\theta : y \in \mathbb{R}^{n_0} \mapsto \mathbb{R}^{n_{L+1}}$ of the following form:

$$f_\theta(y) = [\varphi_{\Theta_{L+1}, b_{L+1}} \circ \xi_L \circ \varphi_{\Theta_L, b_L} \circ \dots \circ \xi_1 \circ \varphi_{\Theta_1, b_1}](y), \quad (7)$$

composed of:

- $L + 1$ parameterized affine functions $\varphi_{\Theta_l, b_l} : z \in \mathbb{R}^{n_{l-1}} \mapsto \Theta_l z + b_l$, where $\Theta_l \in \mathbb{R}^{n_l \times n_{l-1}}$ and $b_l \in \mathbb{R}^{n_l}$ are the weight matrices and bias vectors, respectively, that parameterize the MLP: $\theta = \bigcup_{l=1}^{L+1} \{\Theta_l, b_l\}$,
- L nonlinear functions ξ_l that operate component-wise.

Interestingly, any function f_θ belonging to the MLP class in (7) can be viewed as a neural network. Indeed, by definition, the L intermediate vectors

$$h^{(l)} = [\xi_l \circ \varphi_{\Theta_l, b_l} \circ \dots \circ \xi_1 \circ \varphi_{\Theta_1, b_1}](y) \in \mathbb{R}^{n_l} \quad (8)$$

are called hidden layers and their components are referred to as hidden neurons. In the same way, vectors $h^{(0)} = y$ and $h^{(L+1)} = f_\theta(y)$ are called input layer and output layer, respectively, and their components are named neurons as well, for the sake of consistency. Moreover, the components of matrices Θ_l can be viewed as neural connections since the i^{th} row of Θ_l basically maps all the neurons from the layer $h^{(l)}$ to the i^{th} neuron of the following layer $h^{(l+1)}$. Finally, the nonlinear functions ξ_l are called *activation functions* because they aim to mimic the frequency of action potentials, or “firing”, of real biological neurons.

Historically, the first activation functions that were investigated are the sigmoid functions, characterized by their “S”-shaped curves. In particular the hyperbolic tangent:

$$\tanh : t \mapsto \frac{e^t - e^{-t}}{e^t + e^{-t}} = \frac{1 - e^{-2t}}{1 + e^{-2t}}, \quad (9)$$

ranging from -1 to 1 , and its variants such as the standard logistic function were favored because they are mathematically convenient (easily computable and differentiable as $\tanh'(t) = 1 - \tanh^2(t)$) and are close to linear near origin while saturating rather quickly when getting away from it. In recent developments of deep learning the rectified linear unit (ReLU) is more frequently used as a cost-efficient alternative:

$$\text{ReLU} : t \mapsto \max(0, t). \quad (10)$$

A particularly important result [57, 45, 29] states that any continuous function $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ can be approximated to any given accuracy by a MLP on any compact subspace of \mathbb{R}^n , provided that sufficiently many neurons are available. This result and its derivatives were subsequently named “universal approximation theorems”. They all imply that neural networks can represent a wide variety of interesting functions when given appropriate weights. On the other hand, they typically do not provide a construction for the weights, but merely state that such a construction is possible.

MLP applied on patches for image denoising Given the strong mathematical guarantees provided by the “universal approximation theorems”, the parameterized functions (f_θ) belonging to the MLP class are particularly suitable for approximating the ideal function f minimizing the risk defined in (3). H. C. Burger *et al.* [14] were among the first to investigate the potential of such functions in the field of image denoising. They proposed to use a MLP to denoise the overlapping patches of noisy images, assuming that noise removal is a local issue in the images. This choice is supported by two technical observations. First of all, if MLPs were used on the entire image instead, they would be dependent on the image size which is unintended. Second, and not least, MLPs applied on the complete image would require an intractable number of parameters. Indeed, since a transition from one layer to the next requires a matrix Θ_l of parameters, the total number of parameters of a MLP is of the order of the square of the input size, in the case of constant width MLP. Transposed to images, this represents as many parameters as the square of the number of pixels! This large number of parameters makes its use prohibitive in most cases.

The retained architecture is made up of 4 hidden layers of size 2047 each and is intended to be applied to patches of size $17 \times 17 = 289$. The resulting parameterized function is:

$$f_\theta^{\text{MLP}} : y \in \mathbb{R}^{289} \mapsto [\varphi_{\Theta_{L+1}, b_{L+1}} \circ \xi \circ \dots \circ \xi \circ \varphi_{\Theta_1, b_1}](y), \quad (11)$$

where $L = 4$, the nonlinear activation function ξ is the hyperbolic tangent (9), and the dimensions of the weights and biases are $\Theta_1 \in \mathbb{R}^{2047 \times 289}$, $\Theta_l \in \mathbb{R}^{2047 \times 2047}$ for $2 \leq l \leq 4$, $\Theta_5 \in \mathbb{R}^{289 \times 2047}$, $b_l \in \mathbb{R}^{2047}$ for $l \leq 4$ and $b_5 \in \mathbb{R}^{289}$. The total number of trainable parameters for this MLP is then: $\dim(\theta) = 2 \times 2047 \times 289 + 3 \times 2047 \times 2047 + 4 \times 2047 + 289 = 13,762,270$. For training, H. C. Burger *et al.* [14] used a large *training set* of pairs of

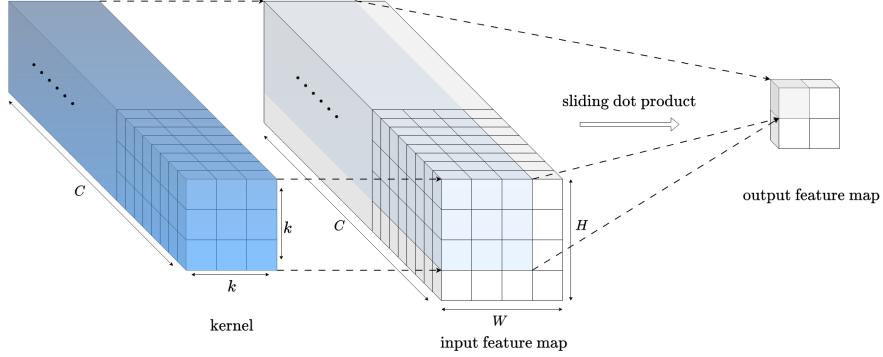


Figure 2: A 3×3 2D convolution (without padding) producing 4 output neurons.

clean/noisy flattened patches (362 million training samples in their experiments of size 17×17 taken from the union of the LabelMe dataset [119], containing approximately 150,000 images, and the Berkeley Segmentation Dataset [99] composed of 400 images) on which the empirical quadratic risk (6) is minimized. At inference, a given noisy image is decomposed into its overlapping flattened patches and each patch is denoised separately with the learned MLP. The final denoised image is obtained by averaging the numerous estimates available for each pixel.

H. C. Burger *et al.* [14] achieved state-of-the-art results on homoscedastic Gaussian noise that compared favorably with BM3D [30], the most cited unsupervised denoiser, at the cost of a full month of training on a GPU at the time. While promising, the resulting denoiser was not yet competitive in terms of inference time and flexibility, as the model handled a single noise level and did not generalize well to other noise levels compared to other denoising methods (although solutions were proposed [136]). Moreover, the multiple artifacts induced by the method as well as its lack of interpretability made it less usable in practice than its conventional counterparts.

2.2.2 Convolutional neural networks (CNN)

Convolutional neural networks is a class of parameterized functions (f_θ) that can be described essentially as a sparse version of multi-layer perceptrons dedicated to two-dimensional inputs. This architecture is widely used in all areas of image processing for its lightness compared to MLPs and its increased performance, image denoising being no exception [25, 97, 145, 146, 148, 147, 87, 5].

Mathematical description The 2D convolution, or 2D cross-correlation, of an image $y \in \mathbb{R}^{H \times W \times C}$, or feature map, of size $H \times W$ composed of C channels (color components for instance but also any abstract embedding of the input pixels) with a weight kernel $\Theta \in \mathbb{R}^{k_1 \times k_2 \times C}$ (restricted to be smaller than the dimensions of the feature map: $k_1 \leq H$ and $k_2 \leq W$), denoted $y \otimes \Theta$, is defined as a sliding dot product between Θ and the local features of y . This operation produces a single-channel output feature map of size $(H - k_1 + 1) \times (W - k_2 + 1)$. More precisely, a 2D convolution $y \otimes \Theta$ consists in splitting the input feature map y into its overlapping 3D blocks of the same size as the kernel Θ – there are $(H - k_1 + 1) \times (W - k_2 + 1)$ overlapping blocks – and computing the dot product with kernel Θ for all of them: each dot product creates a pre-activated neuron. Figure 2 illustrates the process of a 2D convolution. In practice, numerous 2D convolutions are performed successively, involving a different weight kernel Θ_i each time, and their results are concatenated along channels to produce a multi-channel output, or layer. Note that the channel size C' of the output layer is strictly equal to the number of 2D convolutions that were performed. For the sake of notation simplicity, the C' convolutional kernels relative to a same layer are gathered together into a unique 4D kernel, denoted by the same symbol $\Theta \in \mathbb{R}^{k_1 \times k_2 \times C \times C'}$. Finally, a trainable vector (“bias”) $b \in \mathbb{R}^{C'}$ is generally added channel-wisely, leading to the general form of function for 2D convolutions:

$$\psi_{\Theta, b}(y) = y \otimes \Theta + b, \quad (12)$$

where addition applies along channels.

In some cases, it is desirable that the size $H \times W$ of the input image y stays unchanged after a convolutional operation (which is generally not the case, unless $k_1 = k_2 = 1$). A common trick to ensure size preservation is to artificially extend the size of the input image both horizontally and vertically beforehand. This operation is called padding, and the most commonly used padding strategy is simply to add zero-intensity pixels around the edges of the image: zero-padding.

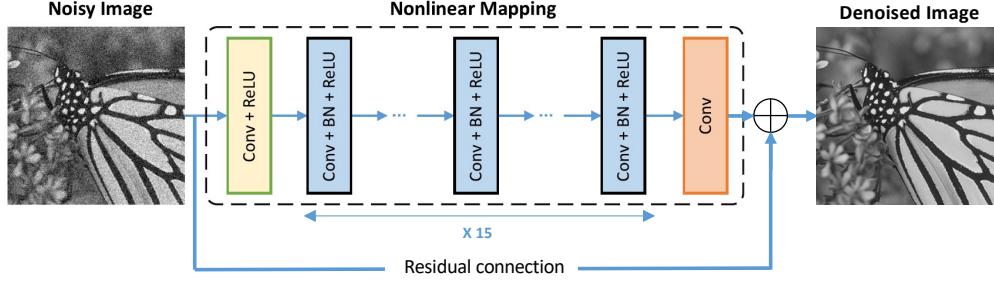


Figure 3: The architecture of DnCNN denoising network. Source: [146].

Formally, a (feed-forward) convolutional neural network with $L \geq 1$ hidden layers is a nonlinear function $f_\theta : y \in \mathbb{R}^{H_0 \times W_0 \times C_0} \mapsto \mathbb{R}^{H_{L+1} \times W_{L+1} \times C_{L+1}}$ that chains 2D convolutions interspersed with nonlinear element-wise operations:

$$f_\theta(y) = [\psi_{\Theta_{L+1}, b_{L+1}} \circ \xi_L \circ \psi_{\Theta_L, b_L} \circ \dots \circ \xi_1 \circ \psi_{\Theta_1, b_1}](y), \quad (13)$$

composed of:

- $L+1$ parameterized convolutional functions $\psi_{\Theta_l, b_l} : z \mapsto z \otimes \Theta_l + b_l$, where Θ_l and b_l are the weight kernels and bias, respectively, that parameterize the CNN: $\theta = \bigcup_{l=1}^{L+1} \{\Theta_l, b_l\}$,
- L nonlinear functions ξ_l that operate component-wise.

Just like MLPs, the L intermediate vectors:

$$h^{(l)} = [\xi_l \circ \psi_{\Theta_l, b_l} \circ \dots \circ \xi_1 \circ \psi_{\Theta_1, b_1}](y) \in \mathbb{R}^{H_l \times W_l \times C_l} \quad (14)$$

are called hidden layers and their components are referred to as hidden neurons. Essentially, a CNN is a MLP where affine functions are replaced with convolutional ones. A direct advantage of CNNs over MLPs is that the number of parameters is generally much smaller, as neural connections are local and identical, whatever the pixel position in the image.

Note that the basic parameterized form (13) of CNNs can be made more complex by adding, amongst others, strided or dilated convolutions [139], skip or residual connections [51], downscaling operations via pooling layers (*e.g.* max pooling, average pooling...) and upscaling operations via bilinear or bicubic interpolation. A general architecture possibly incorporating all of these features is the famous U-Net architecture [115], widely used in computer vision.

Receptive field In a convolutional layer as shown in Fig. 2, each neuron receives input from only a restricted area of the previous layer called the neuron's receptive field. The receptive field has typically a 3D rectangle shape. When the network processes the input data through multiple convolutional layers, the receptive field of a neuron in deeper layers becomes larger as it incorporates information from a broader area of the input. For instance, the receptive field of a network chaining two successive 3×3 convolutional layers is the same as the receptive field of a 5×5 convolution. The receptive field of a CNN is determined by its architectural characteristics, such as the size of the convolutional filters or the downscaling pooling operations. As moving deeper into the network, each neuron's receptive field expands due to the cascading effect of the multiple layers. Consequently, neurons in the deeper layers capture more global and complex features that encompass larger regions of the input image. Understanding the receptive field is crucial in CNNs, as it determines the spatial context that a network can capture, which is particularly essential in image denoising. Indeed, the spatial context may potentially be very useful to detect repeated patterns and denoise them properly. This is why deep CNNs with small convolutional kernels (3×3) are widely used in computer vision; the receptive field is directly proportional to the width of the network, while the number of parameters is contained with small kernels.

Focus on DnCNN architecture DnCNN [146] (Denoising Convolutional Neural Network) is the most cited artificial neural network for image denoising so far. Its widespread popularity is due to both its simplicity and its effectiveness. Although it was developed in the early years of deep learning for image denoising, it is still considered a reference today. DnCNN is basically a feed-forward denoising convolutional neural network that chains “conv+ReLU” blocks, and where residual learning [51] and batch normalization [60] are utilized to speed up the training process as well as boost the denoising performance. Its architecture is illustrated in Figure 3. Formally, DnCNN encodes the following

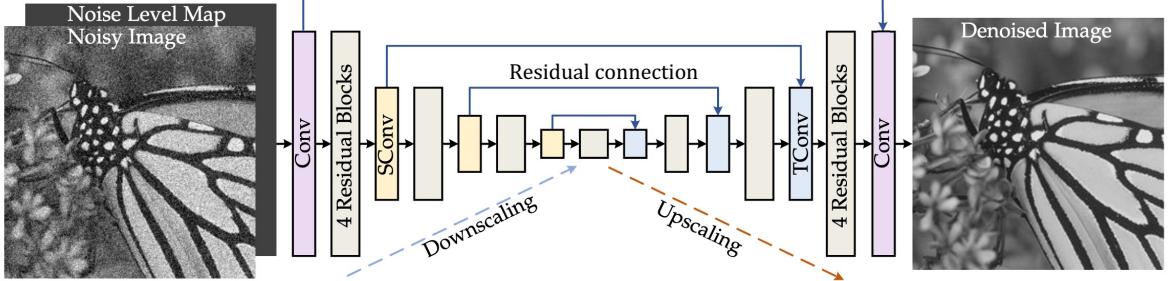


Figure 4: The architecture of DRUNet denoising network. It takes an additional noise level map as input and combines both U-Net [115] and ResNet [51]. “SConv” and “TConv” represent 2×2 strided convolution and transposed convolution, respectively. Source: [145].

parameterized function for grayscale images:

$$f_{\theta}^{\text{DnCNN}} : y \in \mathbb{R}^{H \times W \times 1} \mapsto \quad (15)$$

$$[\psi_{\Theta_{L+1}, b_{L+1}} \circ \xi \circ \psi_{\Theta_L, b_L} \circ \dots \circ \xi \circ \psi_{\Theta_1, b_1}] (y) + y,$$

where $L = 16$, the nonlinear activation function ξ is the ReLU function (10), and the dimensions of the kernels and biases are $\Theta_1 \in \mathbb{R}^{3 \times 3 \times 1 \times 64}$, $\Theta_l \in \mathbb{R}^{3 \times 3 \times 64 \times 64}$ for $2 \leq l \leq 16$, $\Theta_{17} \in \mathbb{R}^{3 \times 3 \times 64 \times 1}$, $b_l \in \mathbb{R}^{64}$ for $l \leq 16$ and $b_{17} \in \mathbb{R}$. Note that the width of the hidden layers (number of channels) is arbitrarily set to 64 for each and neither spatial upscaling, nor downscaling is used (zero-padding is leveraged all along the layers to preserve the spatial input size $H \times W$). The total number of trainable parameters for DnCNN is then: $\dim(\theta) = 3 \times 3 \times 64 \times 64 \times 15 + 3 \times 3 \times 64 \times 2 + 64 \times 16 + 1 = 555,137$, making it much lighter than the MLP proposed by H. C. Burger *et al.* [14]. For training, the authors [146] used the 400 clean images from the Berkeley Segmentation Dataset [99] (BSD400) that they corrupted artificially with additive white Gaussian noise (AWGN) to create pairs of clean/noisy images on which the MSE is minimized. Unlike existing denoising models, which typically trained a specific set of parameters for AWGN for each noise level, DnCNN is also able, at the cost of a relatively small drop in terms of performance, to handle Gaussian denoising with an unknown noise level using a single set of parameters. This characteristic is generally referred to as “blind” Gaussian denoising, since the network has no knowledge of the input noise level. Moreover, the authors showed that this architecture is actually much more versatile, and can be efficiently used beyond Gaussian denoising to tackle several other inverse problems close to Gaussian image denoising. In particular, they trained a single model for three general tasks at once, namely blind Gaussian denoising, single image super-resolution (SISR) and JPEG image deblocking. For SISR, a high-resolution image is generated by first applying the bicubic upscaling on the low resolution image and then treating the inherent remaining “error noise” with DnCNN. Likewise, the unavoidable JPEG artifacts produced by a JPEG encoder during lossy compression are viewed as a particular type of additive noise and treated as such with the general model. Note that treating JPEG deblocking with a denoiser dedicated to Gaussian noise was already studied in [41].

Focus on DRUNet architecture More recently, DRUNet [145] (Denoising Residual U-Net) is an architecture that was proposed as an even more competitive alternative to DnCNN [146], at the price of an increased number of parameters and a longer training on a larger dataset. It achieves state-of-the-art performances for Gaussian noise removal. Contrary to DnCNN, DRUNet adopts a U-Net architecture [115], and as such has an encoder-decoder type pathway, with residual connections [51] all along the network. Spatial downscaling is performed using 2×2 convolutions with stride 2 (“SConv”), while spatial upscaling leverages 2×2 transposed convolutions with stride 2 (“TConv”) (which is equivalent to a 1×1 sub-pixel convolution [124]). The number of channels in each layer from the first scale to the fourth scale are 64, 128, 256 and 512, respectively and each scale is composed of 4 successive residual blocks “ 3×3 conv + ReLU + 3×3 conv”. In total, the retained architecture presents 32,638,656 parameters, which is approximately 60 times more than the number of parameters of DnCNN [146], but thanks to the spatial downscaling operations, the computational complexity is contained. DRUNet architecture is illustrated in Figure 4. Contrary to DnCNN, DRUNet is a “non-blind” denoiser and thus achieve increased performance over ‘blind’ models [146, 147], by passing an additional noisemap as input. In the case of additive white Gaussian noise of variance σ^2 , the noisemap is constant equal to σ . Note that this feature was first proposed by FFDNet [148], which is more or less the flexible “non-blind” variant of DnCNN [146].

Training plays a major role in the success of DRUNet. Indeed, it is widely acknowledged that convolutional neural networks generally benefit from the availability of large training data. Therefore, the training dataset BSD400 [99] has been considerably enriched with the addition of many high-definition images, namely 4,744 images from the Waterloo Exploration Database [92], 900 images from the DIV2K dataset [3], and 2,750 images from the Flickr2K dataset [82]. Moreover, the authors recommend to train it by minimizing the ℓ_1 loss instead of the mean squared error (MSE), supposedly due to its outlier robustness properties. DRUNet was trained to deal with images corrupted with noise levels up to $\sigma = 50$.

2.2.3 Transformers

Originally stemming from the field of natural language processing (NLP), where their introduction have led to significant improvements over convolutional neural networks, transformer-based models [133] have recently been investigated in image denoising [107, 84, 149, 81, 142, 144, 22]. This type of artificial neural network is based on the mechanism of self-attention, which allows a model to decide how important each part of an input sequence is, making it possible to find complex correlations in the data.

Mathematical description From a multi-channel input $Y \in \mathbb{R}^{n \times m}$, where n denotes the number of pixels, in the case of image denoising, and m denotes the channel-size (color components for instance but also any abstract embedding of input pixels), a self-attention module produces at first three different embeddings of Y : queries $Q \in \mathbb{R}^{n \times l}$, keys $K \in \mathbb{R}^{n \times l}$, and values $V \in \mathbb{R}^{n \times k}$. Traditionally, matrices Q , K and V are learned via three projection matrices $\Theta_Q \in \mathbb{R}^{m \times l}$, $\Theta_K \in \mathbb{R}^{m \times l}$ and $\Theta_V \in \mathbb{R}^{m \times k}$, such that $Q = Y\Theta_Q$, $K = Y\Theta_K$ and $V = Y\Theta_V$; but any transformation that produces the desired output shapes from Y is actually suitable. Then, the self-attention is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^\top)V \quad (16)$$

where $\text{softmax} : \mathbb{R}^n \mapsto \mathbb{R}^n$ is such that $\text{softmax}(z)_i = e^{z_i} / \sum_{j=1}^n e^{z_j}$ and is applied over the horizontal axis in (16). Note that $\text{softmax}(QK^\top)$ is nothing else than a right stochastic matrix of size n , which aims at encoding the attention weights. In others words, a self-attention module processes each entry, or “token”, by a convex combination of all the values $V_{i,:}$, weighted by the degree of attention or similarity. Moreover, it is worth noticing that the fact that Q and K are *a priori* different matrices allows attention matrix $\text{softmax}(QK^\top)$ to be non-symmetric: token i may be strongly related to token j and, at the same time, token j may be weakly related to token i on the contrary.

The self-attention operation can actually be viewed as a general learned version of the popular NL-means [13] denoiser, when rewritten as follows:

$$\begin{aligned} \text{Attention}(Q, K, V)_{i,:} &= W_i^{-1} \sum_{j=1}^n e^{-d(Q_{i,:}, K_{j,:})} V_{j,:} \\ W_i &= \sum_{j=1}^n e^{-d(Q_{i,:}, K_{j,:})}, \end{aligned}$$

where the *pseudo* distance metric d between $Q_{i,:}$ and $K_{j,:}$ is defined as $d(Q_{i,:}, K_{j,:}) = -\langle Q_{i,:}, K_{j,:} \rangle$. Indeed, as observed by [84], the NL-Means denoiser [13] is basically a transformer from the matrix of noisy patches $Y \in \mathbb{R}^{n \times m}$ ($m = p \times p$ where p denotes the patch size), with identity embeddings $Q = K = Y$ and values $V = Ye_{[m/2]} = y \in \mathbb{R}^{n \times 1}$ equal to the input noisy image, and with d replaced by the squared Euclidean distance: $d(Q_{i,:}, K_{j,:}) = \|Q_{i,:} - K_{j,:}\|_2^2/h^2$, with the hyperparameter h , often chosen to be proportional to the noise level σ [12, 96, 35]. As a matter of fact, even if the distance metric d was originally chosen as the opposite of the dot product between two embedded vectors for the sake of computational efficiency, the squared Euclidean distance yields comparable performance in image denoising [84].

In practice, self-attention operations cannot be applied on the entire image for the reason that the attention matrix $\text{softmax}(QK^\top)$ in (16) has as many entries as the squared of the input size n , which is in general intractable. That is why, just like MLPs (7), self-attention modules are deployed on subparts of the image. In general, it is used to process non-overlapping groups of neighboring embedded patches of the image [81, 142, 144]. Finally, self-attention modules are usually combined with convolutional layers (12) to get the best of both [107, 84, 149, 81, 142, 144].

Focus on SCUNet architecture Relying heavily on the DRUNet architecture (see Fig. 4), the Swin-Conv-UNet (SCUNet) denoising network [144] has recently been proposed as a successful attempt to incorporate self-attention modules into a convolutional neural network in order to achieve state-of-the-art performances in supervised image denoising. SCUNet basically adopts the same U-Net backbone of DRUNet and replaces the residual convolutional blocks “ 3×3 conv + ReLU + 3×3 conv” by Swin-Conv (SC) hybrid blocks. Figure 5 summarizes the overall architecture. As illustrated, a Swin-Conv (SC) block divides in half along the channels the feature map of a 1×1 convolution to feed two independent branches, namely the “RConv” branch and the “SwinT” branch. The “RConv” branch is simply a residual convolutional block “ 3×3 conv + ReLU + 3×3 conv”, already used in DRUNet [145],

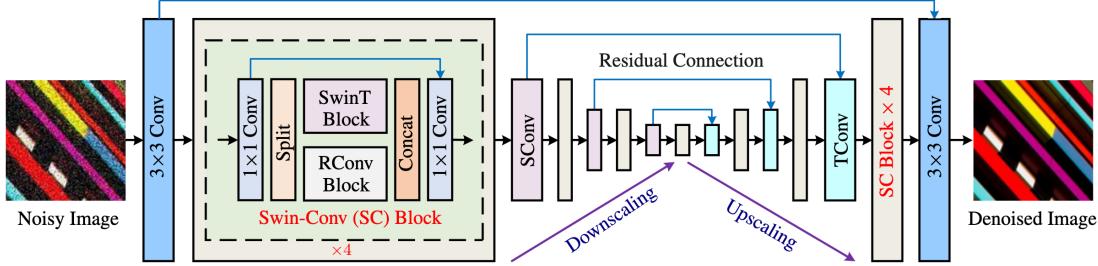


Figure 5: The architecture of SCUNet denoising network. “SConv”, “TConv”, “RConv” and “SwinT” represent 2×2 strided convolution, 2×2 strided transposed convolution, residual “ 3×3 conv + ReLU + 3×3 conv” block and swin transformer block, respectively. Source: [144].

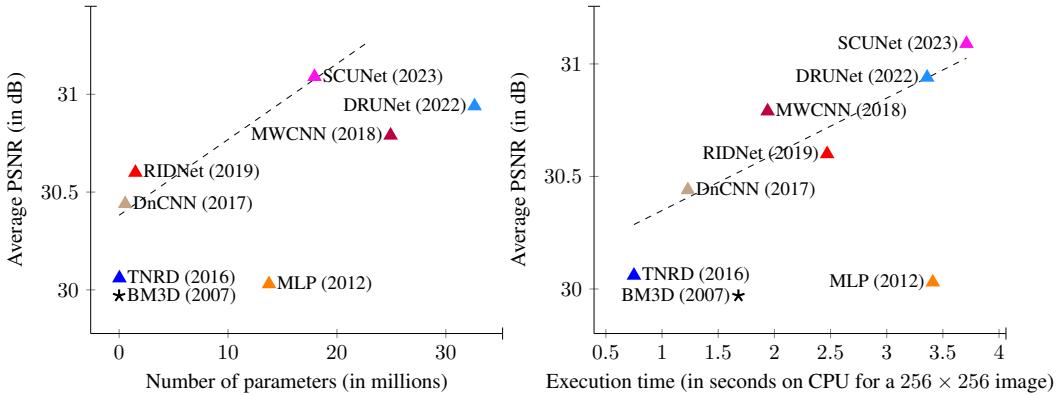


Figure 6: Performance evolution of supervised models [144, 146, 5, 14, 145, 87, 25] with the number of parameters (left) and execution time at inference (right), respectively, for grayscale Gaussian denoising on the Set12 dataset at $\sigma = 25$ (CPU: 2,3 GHz Intel Core i7). A general trend can be observed: increased performance is achieved at the cost of an increase in the number of parameters and execution time (the linear trend, in dashed line, is estimated with Theil-Sen method).

with twice less parameters as in the original network, since the channel size has been halved due to the split of the feature map. As for the “SwinT” branch, it implements the swin transformer block described in [81], in turn based on the standard multi-head self-attention of the original Transformer layer [133]. Essentially, it consists in partitioning the input feature map of size $H \times W \times C$ into multiple non-overlapping groups, or windows, of equal size $(h \times w) \times c$, with $h < H$, $w < W$ and $c < C$, and processing them independently by leveraging self-attention (see formula (16)), with shared projection matrices across different windows. In the retained architecture, all windows are of equal size $(8 \times 8) \times 32$, involving self-attention matrices of size 64×64 . Finally, in order to enable cross-window connections, regular and shifted window (swin) partitioning are used alternately [88], where shifted window partitioning means shifting the feature map by $(\lfloor \frac{h}{2} \rfloor, \lfloor \frac{w}{2} \rfloor)$ pixels before partitioning. In the end, the outputs of the two branches “RConv” and “SwinT” are concatenated channel-wisely and then passed through a 1×1 convolution to produce the final residual of the input.

Although the number of parameters of SCUNet is approximately reduced by half compared to DRUNet [145], since the number of parameters of “SwinT” blocks is negligible in relation to “RConv” blocks, the complexity is slightly increased, though contained. Training basically follows the instructions of DRUNet [145]. Unlike DRUNet, SCUNet was not trained as a ”non-blind” denoiser (*i.e.* with an additional noise level map as input), and requires instead a specific set of parameters for each noise level in the case of AWGN.

In summary, within a decade of research in supervised image denoising, the quality has been considerably enhanced, but at the price of an increased number of parameters and increased execution time (see Fig. 6). Nonetheless, the very best methods [144, 145] are now capable of recovering details barely perceptible to the human eye.

2.3 Parameter optimization

Once the class of parameterized functions (f_θ) – that is the architecture of the neural network – has been chosen, it still remains to select the best member of this class for the task of image denoising. As explained in the previous section, a proven heuristic consists in finding the optimal parameters θ^* that best minimize the empirical risk (6), for want of knowing the true risk (3). In this section, we present the technique commonly adopted to solve this optimization problem, which is essentially based on the gradient descent algorithm.

2.3.1 Back-propagation

In most of cases, the minimization of the empirical risk (6) cannot be performed analytically for the reason that the chosen class of parameterized functions is in general very complex. Indeed, the resulting optimization problem (5) is usually highly non-convex and the only fast and efficient algorithms that remain at our disposal to solve it are first-order gradient-based optimization algorithms. Calculating the gradient $\nabla_\theta \mathcal{R}_{\text{emp}}(f_\theta)$ then becomes essential.

The practical computation of the gradient of any weakly differentiable function at a given point θ has recently been considerably facilitated by the advent of modern machine learning libraries such as Pytorch [106]. Indeed, these novel frameworks are equipped with an automatic differentiation engine that powers the computation of partial derivatives. Automatic differentiation exploits the fact that the computation of a scalar value (e.g., the empirical risk 6) executes a sequence of elementary arithmetic operations (addition, multiplication, etc) and elementary functions (exp, square, etc). By keeping a record of data and all executed operations, partial derivatives can be computed automatically, accurately to working precision, by applying the *chain rule* repeatedly to these operations.

2.3.2 Stochastic gradient descent

Provided with the gradient of the empirical risk with respect to the parameters $\nabla_\theta \mathcal{R}_{\text{emp}}(f_\theta)$, the most basic first-order gradient-based optimization algorithm to solve (5) is the gradient descent algorithm. However, it is in practice computationally very expensive, especially for large training sets. An alternative method for more frequent updating is the stochastic gradient descent (SGD) [113]. Its principle is simple: an approximation of the gradient is computed using a different random subset (mini-batch) of the entire training set at each step. With the same notations as (6), $\mathcal{R}_{\text{emp}}(f_\theta)$ can be approximated by:

$$\mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_\theta) = \frac{1}{|\mathcal{B}|} \sum_{s \in \mathcal{B}} \|f_\theta(y_s) - x_s\|, \quad (17)$$

where \mathcal{B} denotes a random subset of $\{1, \dots, S\}$, so that $\nabla_\theta \mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_\theta) \approx \nabla_\theta \mathcal{R}_{\text{emp}}(f_\theta)$. Then, $\nabla_\theta \mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_\theta)$ can be viewed as a noisy version of the true gradient $\nabla_\theta \mathcal{R}_{\text{emp}}(f_\theta)$. Note that, computing the gradient over a single pair of clean/noisy images (x_s, y_s) , can still be computationally expensive when dealing with high resolution images. This is why, $\mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_\theta)$ is usually further approximated by replacing the image pairs (x_s, y_s) in (17) by pairs of small image patches, typically of size 128×128 , randomly cropped from the same images. The procedure is summarized in Algorithm 1.

Algorithm 1 Stochastic Gradient Descent (SGD) algorithm

Require: Initial parameters θ_0 , learning rate α , batch size b , number of iterations T .

Ensure: Updated parameters θ_T

for $t = 1, \dots, T$ **do**

Select a random subset $\mathcal{B} \subset \{1, \dots, S\}$ of size b .

Compute gradient at point θ_{t-1} : $g_t \leftarrow \nabla_\theta \mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_{\theta_{t-1}})$.

Update parameters: $\theta_t \leftarrow \theta_{t-1} - \alpha g_t$.

end for

2.3.3 Adam optimization algorithm

Adam [69] (Adaptive Moment Estimation) is a popular extension of the stochastic gradient descent algorithm [113], widely used in the field of image denoising [146, 145, 144, 148, 107, 84] for its computational efficiency and little memory requirements. Adam combines the concepts of adaptive learning rates and momentum to provide faster convergence compared to traditional gradient descent methods, while making it less sensitive to the choice of initial learning rate. To do so, the algorithm keeps track of statistics of the first and second moment vectors, that is the gradient and its per-element square, via an exponentially decaying average. The first order moment incorporates the momentum and helps in maintaining the direction of the gradients, while the second order moment captures the

magnitudes of the gradients for better adjusting the learning rates. The algorithm 2 provides an update rule similar to SGD [113].

Algorithm 2 Adam algorithm

Require: Initial parameters θ_0 , learning rate α , batch size b , number of iterations T , running average parameters $(\beta_1, \beta_2) = (0.9, 0.999)$, additional term for numerical stability $\varepsilon = 10^{-8}$.

Ensure: Updated parameters θ_T

Initialize first and second moment vectors: $m_0 \leftarrow \mathbf{0}$ and $v_0 \leftarrow \mathbf{0}$.

for $t = 1, \dots, T$ **do**

Select a random subset $\mathcal{B} \subset \{1, \dots, S\}$ of size b .

Compute gradient at point θ_{t-1} : $g_t \leftarrow \nabla_{\theta} \mathcal{R}_{\text{emp}}^{\mathcal{B}}(f_{\theta_{t-1}})$.

Update running averages: $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$ and $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^{\odot 2}$.

Compute bias-corrected moments: $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ and $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$.

Update parameters: $\theta_t \leftarrow \theta_{t-1} - \alpha \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$

end for

Unfortunately, the best neural network architecture for image denoising, combined with the best optimization procedure, is powerless if high-quality clean/noisy image pairs are lacking for learning in some respects. A recent line of research tries to relax the need for clean images by adopting a so-called *weakly* supervised learning approach.

2.4 Weakly supervised learning

In numerous contexts, the availability of sufficiently many noise-free images is not guaranteed and supervised learning cannot be applied effectively. To circumvent this problem, attempts have been made recently to adapt empirical risk minimization (6) with neural networks without ground truth. Note that, in the following, we make the arbitrary distinction between a supervised approach – for which the *training set* consists in a subset of $\mathcal{X} \times \mathcal{Y}$, designating all possible pairs of clean/noisy images, whether it is physically acquired or synthetically generated (approximated) – and a weakly supervised approach, for which the *training set* present solely representative images from \mathcal{Y} .

2.4.1 Learning from noisy image pairs

A pioneer work in this spirit is Noise2Noise [77] that assumes that, for the same underlying ground truth image x_s , two independent noisy observations y_s and \bar{y}_s are available. It was observed that replacing the clean/noisy pairs (x_s, y_s) by the noisy/noisy ones (\bar{y}_s, y_s) in the empirical quadratic risk (6) enables comparable performance to be achieved without the need for ground truths, provided that the noise is zero-mean.

A typical use case is for example fluorescence microscopy where biological cells can be fixed using a fixative agent which causes cell death, while maintaining cellular structure. By taking two successive shots of the same scene, assuming that the noise realizations are independent between them and zero-mean, it is possible to constitute a dataset composed of noisy/noisy pairs (\bar{y}_s, y_s) to train a neural network f_θ . Once optimized for specifically denoising fluorescence microscopy images, the network can be deployed in a complete image processing pipeline, where noisy image pairs are no longer required (in particular, cells no longer need to be fixed).

Formally, let f_θ be a parameterized function, x following the distribution of *natural* images, and y and \bar{y} two independent random vectors following the same noise distribution from x (for instance $y \sim \mathcal{N}(x, \sigma^2 I_n)$ or $y \sim \mathcal{P}(x)$). Assuming that $\mathbb{E}_{y|x}(y) = \mathbb{E}_{\bar{y}|x}(\bar{y}) = x$, we have, by developing the squared ℓ_2 norm:

$$\begin{aligned} \|f_\theta(y) - \bar{y}\|_2^2 &= \|(f_\theta(y) - x) - (\bar{y} - x)\|_2^2 \\ &= \|f_\theta(y) - x\|_2^2 + \|\bar{y} - x\|_2^2 - 2\langle f_\theta(y) - x, \bar{y} - x \rangle. \end{aligned}$$

Therefore, by taking the expected value over x, y and \bar{y} :

$$\begin{aligned} \text{N2N}(f_\theta) &:= \mathbb{E}_{y, \bar{y}} \|f_\theta(y) - \bar{y}\|_2^2 \\ &= \mathbb{E}_{x, y} \|f_\theta(y) - x\|_2^2 + \mathbb{E}_{x, \bar{y}} \|\bar{y} - x\|_2^2 \\ &\quad - 2\mathbb{E}_x (\mathbb{E}_{y, \bar{y}|x} \langle f_\theta(y) - x, \bar{y} - x \rangle) \\ &= \mathcal{R}(f_\theta) + \text{const}, \end{aligned} \tag{18}$$

where $\mathcal{R}(f_\theta) := \mathbb{E}_{x, y} \|f_\theta(y) - x\|_2^2$ is the quadratic risk already defined in (3). Note that the expected value of the dot product cancels out since the components of y and \bar{y} are independent, and the noise is assumed to be zero-mean.

In the end, minimizing the risk $\mathcal{R}(f_\theta)$ amounts to minimizing the surrogate N2N(f_θ) insofar as they differ by a constant value. The advantage of using N2N(f_θ) is that this expression depends only on the observations (y, \bar{y}) and does not involve the clean images x anymore. Consequently, minimizing the Noise2Noise loss is formally equivalent to minimizing the usual supervised quadratic risk. For a given neural network f_θ , assuming ideal optimization, the Noise2Noise approach leads to the exact same weights θ^* as the supervised approach and so yields exact same performances even if it is trained without ground truth.

However, the above reasoning assumes that an infinite amount of noisy training data is provided. In practice, for want of knowing the true risk (3), the empirical risk (6) is minimized instead, and the equality (18) does not hold for finite samples. Indeed, the average of dot product in (18) is as close to zero as the number of noisy data increases. Consequently, the performance of Noise2Noise drops when the amount of training data is reduced, limiting its capability in practical scenarios. In order to get the best out of Noise2Noise potential with limited noisy data, A. F. Calvarons [18] recently proposed to exploit the duplicity of information in the noisy pairs to generate some sort of data augmentation.

2.4.2 Learning from single noisy images

In certain denoising tasks, however, the acquisition of two or more noisy copies per image can be very expensive or impractical, in particular in medical imaging where patients are moving during the acquisition, or in videos with moving objects, etc. An even more remarkable line of research focuses on the possibility to train neural networks on datasets composed only of single noisy observations y_s .

SURE Assuming an additive white Gaussian noise model of variance σ^2 , a classical result from estimation theory – Stein’s unbiased risk estimate (SURE) [129] – was investigated for training neural networks on datasets composed only of single noisy observations (y_s) [125]. Formally, let x follow the distribution of *natural* images and $y \sim \mathcal{N}(x, \sigma^2 I_n)$. According to [129], we have:

$$\begin{aligned} \text{SURE}(f_\theta) &:= \mathbb{E}_y \|f_\theta(y) - y\|_2^2 + 2\sigma^2 \text{div}(f_\theta)(y) - n\sigma^2 \\ &= \mathbb{E}_{x,y} \|f_\theta(y) - x\|_2^2 = \mathcal{R}(f_\theta), \end{aligned} \quad (19)$$

where n is the dimension of images y (*i.e.* number of pixels). The advantage of using SURE is that the risk is expressed in such a way that it depends only on the observations y . Nevertheless, the SURE loss requires the computation of the divergence of f_θ at points y which is cumbersome. To overcome this difficulty, the use of a fast Monte-Carlo approximation to compute the divergence term defined in [112] is leveraged in [125]:

$$\text{div}(f_\theta)(y) \approx \varepsilon^\top \frac{f_\theta(y + h\varepsilon) - f_\theta(y)}{h}, \quad (20)$$

where ε is one single realization of the standard normal distribution $\mathcal{N}(0, I_n)$ and h is a fixed small positive value.

As in the case of the N2N loss (18), minimizing the SURE loss is strictly equivalent to minimizing the usual supervised quadratic risk only if an infinite amount of training data is provided, which in practice does not happen. Indeed, the equality (19) does not hold for finite samples for similar reasons. For a sufficiently large number of data samples however, it is possible to obtain performances close to those of networks trained with ground truths.

Blind-spot networks A radical way to get rid of the divergence term is to force f_θ to be divergence-free, *i.e.* $\text{div}(f_\theta)(y) = 0$ for all y . To that end, Noise2Self [6] introduces the concept of \mathcal{J} -invariance. Namely, a function f_θ is said to be \mathcal{J} -invariant if for each subset of pixels $J \in \mathcal{J}$, the pixel values of $f_\theta(y)$ at J are computed such that they do not depend on the values of y at J . Note that such functions are in particular divergence-free since $\frac{\partial f_\theta^i}{\partial y_i}(y) = 0$ for all y , where f_θ^i denotes the i^{th} component of f_θ . In the literature, divergence-free networks are more often referred to as blind-spot networks [70], as they are constrained to estimate the pixel value based on the neighboring pixels only.

Contrary to SURE loss which is limited to additive white Gaussian noise, blind-spot networks can be leveraged in a more general context. Indeed, provided that the noise is independent between pixels and is zero-mean, the minimizer the so-called self-supervised loss $\text{N2S}(f_\theta) := \mathbb{E}_y \|f_\theta(y) - y\|_2^2$ is exactly the minimizer of the quadratic risk (3) [6]. Formally, let x follow the distribution of *natural* images and let y follow a noise distribution from x which is independent between pixels (for example $y \sim \mathcal{N}(x, \sigma^2 I_n)$ or $y \sim \mathcal{P}(x)$). Assuming that $\mathbb{E}_{y|x}(y) = x$, we have, by developing the squared ℓ_2 norm:

$$\|f_\theta(y) - y\|_2^2 = \|f_\theta(y) - x\|_2^2 + \|y - x\|_2^2 - 2\langle f_\theta(y) - x, y - x \rangle.$$

Therefore, by taking the expected value over x and y :

$$\begin{aligned} \text{N2S}(f_\theta) &:= \mathbb{E}_y \|f_\theta(y) - y\|_2^2 \\ &= \mathbb{E}_{x,y} \|f_\theta(y) - x\|_2^2 + \mathbb{E}_{x,y} \|y - x\|_2^2 \\ &\quad - 2\mathbb{E}_x (\mathbb{E}_{y|x} \langle f_\theta(y) - x, y - x \rangle) \\ &= \mathcal{R}(f_\theta) + \text{const} , \end{aligned} \tag{21}$$

where $\mathcal{R}(f_\theta) := \mathbb{E}_{x,y} \|f_\theta(y) - x\|_2^2$ is the quadratic risk already defined in (3). Note that the expected value of the dot product cancels out since f_θ is blind-spot, the components of y are independent between pixels, and the noise is assumed to be zero-mean. Therefore, minimizing the risk $\mathcal{R}(f_\theta)$ amounts to minimizing the surrogate N2S(f_θ) insofar as they differ by a constant value. An ingenious example of a divergence-free network is proposed by Noise2Kernel [76] that exploits donut kernels for the first layer and dilated convolutional kernels for the next layers. Finally, note that Noise2Void [70] proposed before Noise2Self [6] the idea of using the self-supervised loss with a blind-spot network, although the theoretical justification provided was not as strong as that of [6].

Nevertheless, the performance of divergence-free functions is considerably limited by the constraint of not voluntarily using the information of key pixels. Indeed, except from the parts of the signal that are easily predictable (for example uniform regions), counting exclusively on the information provided by the neighborhood to denoise the pixels is an inefficient strategy. Think for example of the extreme case of a uniform black image with a single white pixel on its center. With a blind-spot network, the central white pixel will be lost and wrongly replaced by a black one.

Probabilistic blind-spot networks To improve the performance of blind-spot networks, several authors [71, 72, 110] propose to refine the predictions during inference when the noise model is known. For this purpose, they adopt a Bayesian point of view, different from the risk minimization point of view (3), used until now. Following this paradigm, a network f_θ is trained so that, given exclusively the noisy surroundings Ω_y of a noisy pixel y (the central noisy pixel y is excluded), it outputs a (parameterized) probability distribution $p_\theta(x|\Omega_y)$ of the central clean pixel. In other words, f_θ is such that $f_\theta(\Omega_y)$ predicts a learned prior probability distribution of the expected central clean value, instead of just predicting a value without taking uncertainty into account, as in the risk minimization paradigm. Equipped with such a function f_θ , Bayes' rule can be applied to update the prior with new information of the noisy central pixel y , provided that the noise model is known, to obtain the posterior distribution:

$$\underbrace{p(x|y, \Omega_y)}_{\text{posterior}} \propto \underbrace{p(y|x, \Omega_y)}_{\text{likelihood}} \underbrace{p(x|\Omega_y)}_{\text{prior}} \approx \underbrace{p(y|x)}_{\text{noise model}} \underbrace{p_\theta(x|\Omega_y)}_{\text{learned prior}} . \tag{22}$$

From the posterior, the Minimum Mean Squared Error (MMSE) estimate (*i.e.* the conditional expectation) or the Maximum A Posteriori (MAP) is produced, which can be considered as an improved version of the prediction given by Noise2Self [6], since it is refined with the information of the central pixel. Note that the adopted Bayesian point of view enables to efficiently combine the knowledge learned on an external dataset composed of noisy images and the information of the input noisy image, which would not have been possible with a risk minimization paradigm.

The remaining questions are now how to construct f_θ and how to train it. First of all, an arbitrary parametric model for the prior $p_\theta(x|\Omega_y)$ needs to be chosen. In [71], f_θ is built in such a way that $f_\theta(\Omega_y)$ outputs a vector of the size of the number of different intensities of the image (a 256-dimensional vector when images are coded on 8 bits for example) where all entries are non-negative and sum to one, interpreted as the histogram of a discrete probability distribution. In [72], $f_\theta(\Omega_y)$ is constrained to follow a Gaussian model and so the output simply consists in a two-dimensional vector, encoding the mean $f_\theta(\Omega_y)_1$ and standard deviation $f_\theta(\Omega_y)_2$ of a Gaussian distribution. As for training, they both use the method of Maximum Likelihood Estimation (MLE). For a data sample $\{y_s\}_{s \in \{1, \dots, S\}}$ of S noisy central pixels surrounded by neighborhoods $\{\Omega_{y_s}\}_{s \in \{1, \dots, S\}}$, the log-likelihood function reads (using the formula of total probability):

$$\ln \mathcal{L}(\theta; \{y_s\}) = \sum_{s=1}^S \ln \underbrace{\int_{-\infty}^{+\infty} p(y_s|x) p_\theta(x|\Omega_{y_s}) dx}_{p_\theta(y_s|\Omega_{y_s})} . \tag{23}$$

In the case of an additive white Gaussian noise model of variance σ^2 and when $f_\theta(\Omega_y)$ is constrained to output a Gaussian model [72], we have: $p(y_s|x) = \mathcal{N}(y_s; x, \sigma^2)$ and $p_\theta(x|\Omega_{y_s}) = \mathcal{N}(x; f_\theta(\Omega_{y_s})_1, f_\theta(\Omega_{y_s})_2^2)$. It follows that

$$\begin{aligned} p_\theta(y_s|\Omega_{y_s}) &= \int_{-\infty}^{+\infty} p(y_s|x) p_\theta(x|\Omega_{y_s}) dx \\ &= \mathcal{N}(y_s; f_\theta(\Omega_{y_s})_1, \sigma^2 + f_\theta(\Omega_{y_s})_2^2) . \end{aligned} \tag{24}$$

Finally, the solution $\theta^* \in \arg \max_{\theta} \ln \mathcal{L}(\theta; \{y_s\})$ is as follows:

$$\theta^* = \arg \min_{\theta} \sum_{s=1}^S \ln(\sigma^2 + f_{\theta}(\Omega_{y_s})_2^2) + \frac{(y_s - f_{\theta}(\Omega_{y_s})_1)^2}{\sigma^2 + f_{\theta}(\Omega_{y_s})_2^2}, \quad (25)$$

which is solved using Adam algorithm [69].

Experiments on artificially noisy images [72] but also on real-world noisy images [71] tend to show that weakly supervised probabilistic approaches are almost on par with their supervised counterparts.

Noisier2Noise An ingenious way of dispensing with the probabilistic approach, while making full use of the central pixel, was proposed by Noisier2Noise [103] and Recorrupted-to-Recorrupted [105]. Their approach is based on adding more noise to single noisy images in the dataset, although this may seem counter-intuitive. The idea of Noisier2Noise [103] is to train a network f_{θ} that maps the original noisy images y from noisier versions z synthetically generated by adding extra noise. The authors argue that, with this strategy, the network is encouraged to predict $\mathbb{E}(y|z)$; and $\mathbb{E}(x|z)$ can be estimated thereafter during the inference step via a linear combination of $\mathbb{E}(y|z) \approx f_{\theta^*}(z)$ and z . For example, in the most simple case where $y = x + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_n)$ and $z = y + \varepsilon'$ with $\varepsilon' \sim \mathcal{N}(0, \sigma^2 I_n)$ with ε' independent from ε , we have, by linearity of expectation and by noticing that $\mathbb{E}(\varepsilon|z) = \mathbb{E}(\varepsilon'|z)$:

$$\begin{aligned} 2\mathbb{E}(y|z) &= \mathbb{E}(x|z) + (\mathbb{E}(x|z) + \mathbb{E}(\varepsilon|z) + \mathbb{E}(\varepsilon'|z)) \\ &= \mathbb{E}(x|z) + \mathbb{E}(z|z) = \mathbb{E}(x|z) + z, \end{aligned}$$

hence $\mathbb{E}(x|z) = 2\mathbb{E}(y|z) - z$. Therefore, at inference, for a noisy observation y , the denoised image is finally estimated by $2f_{\theta^*}(y + \varepsilon') - (y + \varepsilon')$ where ε' is a realization of $\mathcal{N}(0, \sigma^2 I_n)$.

More recently, still in the setting of additive white Gaussian noise (AWGN) of variance σ^2 , i.e. $y \sim \mathcal{N}(x, \sigma^2 I_n)$, Recorrupted-to-recorrupted [105] showed that it is possible, from a noisy image y , to construct an artificial pair of independent noisier images (z, \bar{z}) , centered in x , that can be exploited to train a neural network, just like in [77] (see equation (18)). In the end, a Noise2Noise-like equality holds:

$$\text{R2R}(f_{\theta}) := \mathbb{E}_{z, \bar{z}} \|f_{\theta}(z) - \bar{z}\|_2^2 = \mathbb{E}_{x, z} \|f_{\theta}(z) - x\|_2^2 + \text{const}, \quad (26)$$

where $\mathbb{E}_{x, z} \|f_{\theta}(z) - x\|_2^2$ is a “noisier” risk close to the target risk $\mathcal{R}(f_{\theta})$ defined in (3). Minimizing the R2R loss is then equivalent to minimizing the “noisier” risk. To denoise an input noisy image y at inference, it is first renoised according to the recorruption model z to get the final estimate $f_{\theta^*}(z)$. Provided that the artificial z is not much noisier than y , this strategy achieves performances close to those of networks trained with ground truths.

Interestingly, among the different possible recorruption models, there is the straightforward setting $z = y + \alpha\varepsilon$ and $\bar{z} = y - \varepsilon/\alpha$, with $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_n)$ and $\alpha \in \mathbb{R}^*$. According to the property of affine transformation of Gaussian vectors, we have:

$$\begin{pmatrix} z \\ \bar{z} \end{pmatrix} \sim \mathcal{N} \left(\begin{pmatrix} x \\ x \end{pmatrix}, \begin{pmatrix} (1 + \alpha^2)\sigma^2 I_n & \mathbf{0}_{n \times n} \\ \mathbf{0}_{n \times n} & (1 + 1/\alpha^2)\sigma^2 I_n \end{pmatrix} \right), \quad (27)$$

meaning that z and \bar{z} are independent from each other. In practice, $\alpha = 0.5$ is recommended for training to balance the noise of z and \bar{z} [105].

Noise2Score Finally, another original and versatile method for learning without ground truths was proposed by Noise2Score [68]. In this novel approach, the conditional mean of the posterior distribution $\mathbb{E}(x|y)$ (posterior expectation of x given noisy observation y) is calculated leveraging a classical result from Bayesian statistics, namely Tweedie’s formula [37], which involves the so-called score function. Formally, assuming that the likelihood $p(y|x)$ can be written under the form $p(y|x) = a(x)b(y) \exp(x^\top T(y))$ with $a : \mathbb{R}^n \mapsto \mathbb{R}$, $b : \mathbb{R}^n \mapsto \mathbb{R}$ and $T : \mathbb{R}^n \mapsto \mathbb{R}^n$ (subset of the exponential family which covers a large class of important distributions such as the Gaussian, binomial, multinomial, Poisson, gamma, and beta distributions, as well as many others), then the following equality holds:

$$J_T(y)^\top \mathbb{E}(x|y) = \nabla_y \ln(p(y)) - \nabla_y \ln(b(y)), \quad (28)$$

where J_T denotes the Jacobian matrix of function T . In particular, when T has the simple form $T(y) = cy$, with $c \in \mathbb{R}^*$, $J_T(y)^\top = cI_n$ and finally the conditional mean of the posterior distribution is:

$$\mathbb{E}(x|y) = (\nabla_y \ln(p(y)) - \nabla_y \ln(b(y))) / c, \quad (29)$$

where $\nabla_y \ln(p(y))$ is referred to as the score (gradient of the marginal distribution of y).

As it stands, the formula (29) is purely theoretical since the distribution of *natural* noisy images $p(y)$ is at least as difficult to know as the distribution of *natural* images $p(x)$. However, capitalizing on the recent finding that the score function can be stably estimated from the noisy images [83], Noise2Score [68] suggests to use a residual denoising autoencoder f_θ for approximating the score:

$$\begin{aligned} \nabla_y \ln(p(y)) &\approx f_{\theta^*}(y), \\ \theta^* &\in \arg \min_{\substack{y \sim p(y), \varepsilon \sim \mathcal{N}(0,1) \\ \alpha \sim \mathcal{N}(0, \delta^2)}} \mathbb{E}_{y \sim p(y), \varepsilon \sim \mathcal{N}(0,1)} \|f_\theta(y + \alpha\varepsilon) + \varepsilon/\alpha\|_2^2 \end{aligned} \quad (30)$$

with $\delta \rightarrow 0$ (note the similarity with Recorrupted-to-recorrupted [105] for recorrupted images $y + \alpha\varepsilon$). The advantage of Noise2Score [68] is that, provided that the noise model belongs to the exponential family distribution, the problem comes down to estimating the score function always approximated by the same universal training (30).

In the case of an additive white Gaussian noise model of variance σ^2 , we have:

$$p(y|x) = a(x)b(y) \exp(x^\top T(y)), \quad (31)$$

with $a(x) = (\sigma\sqrt{2\pi})^{-n} \exp(-\frac{1}{2\sigma^2}\|x\|_2^2)$, $b(y) = \exp(-\frac{1}{2\sigma^2}\|y\|_2^2)$ and $T(y) = y/\sigma^2$. As $\nabla_y \ln(b(y)) = -y/\sigma^2$, Tweedie's formula then reads:

$$\mathbb{E}(x|y) = y + \sigma^2 \nabla_y \ln(p(y)) \approx y + \sigma^2 f_{\theta^*}(y). \quad (32)$$

2.5 Discussion and conclusion

In spite of their great theoretical interest, weakly supervised approaches for image denoising, which are designed to learn without ground truths, are unfortunately of limited practical value. Indeed, if collecting a dataset of noisy image pairs is assumed to be possible as in Noise2Noise [77], why not collect several n -tuples of noisy images instead which, once averaged, would constitute ground truth images for use in a supervised framework (approach retained for the datasets of [110] for example). As for learning from datasets of single noisy images, the proposed approaches are either disappointing in terms of performance [6, 70] due to strong architectural constraints, or, require the noise model to be known [125, 71, 72, 103, 105, 68] in order to achieve performance comparable to that of supervised models. As a consequence, weakly supervised learning is far from being the preferred strategy for tackling challenging benchmarks such as the Darmstadt Noise Dataset [108] where only single real-world noisy images are available, for which the real noise can only be roughly approximated mathematically by a mixed Poisson-Gaussian model. Instead, the best-performing methods [11, 143, 17, 144] simulate a large amount of realistic noisy images from clean ones by carefully considering the noise properties of image sensors, on which any denoising neural network can be trained on. The same observation can be made in fluorescence microscopy, where the most popular denoising neural network [137] was trained in a supervised way, whether on physically acquired or synthetic training data.

3 Unsupervised denoising methods

Both supervised and weakly supervised learning strategies are extremely dependent on data quality (although they do not rely on the same type of image pairs), which is a well-established weakness. In some situations, it may be challenging to gather a large enough dataset for learning. Only unsupervised methods - in which only the noisy input image is used for training - are operationally available. Historically, these methods were studied before their supervised counterparts, partly due to the computational limitations of the time that made resource-intensive supervised learning unthinkable. In this chapter, we present a non-exhaustive list of well-known unsupervised algorithms, classified according to four different main principles. As we shall see, the best unsupervised denoisers share key elements, in particular the property of self-similarity observed in images, whatever their category.

3.1 Weighted averaging methods

The most basic unsupervised methods for image denoising are without a doubt the smoothing filters, among which we can mention the averaging filter or the Gaussian filter for the linear filters and the median filter for the nonlinear ones. Interestingly, the linear smoothing filters can actually be viewed formally as elementary convolutional neural networks $f_\Theta(y) = y \otimes \Theta$ already defined in Section II with no bias, no hidden layer and no activation function, and with unique convolutional kernel Θ . In contrast to supervised CNNs, the kernel is non-trainable. Note that symmetric padding is applied on the noisy image y beforehand to ensure size preservation.

In practice, the smoothing filters act by replacing each intensity value of noisy pixels with a convex combination of those of its neighboring noisy pixels. Denoising is made possible, at the cost of edge blur, by reducing the variation

in intensity between neighboring pixels. Although these filters are extremely rudimentary, they are sometimes used as pre-processing steps in some popular algorithms where performance is not at stake such as the Canny edge detector [19] due to their unbeatable speed. Building on the idea of convex combinations of noisy pixels, numerous extensions were proposed by better adapting to the local structure of the images [131, 13, 61, 122, 101, 121]. In what follows, we review three major unsupervised denoisers [131, 13, 61] processing images via convex combinations of noisy pixels.

Formally, we denote by y a vectorized noisy image patch of size n whose central pixel is y_c (the value of index c is $\lceil n/2 \rceil$). Each method of this subsection implements a denoising function of the form $f_\theta(y) = y^\top \theta$ aimed at estimating the noise-free central pixel x_c , and where the weights $\theta \in \mathbb{R}^n$ are patch-dependent and are such that $\mathbf{1}_n^\top \theta = 1$ and $\theta \succeq 0$.

as the bilateral filter [131] that evaluates the intensity similarity between two neighboring pixels, the seminal work from A. Buades *et al.* [13] adopts a more robust approach by exploiting the similarity of patches instead. For each pixel, an average of the neighboring noisy pixels, weighted by the degree of similarity of patches they belong to, is leveraged for edge-preserving denoising. The convex weights of the N(on)-L(ocal) Means can be defined as:

$$\theta_i = K_i^s K^r(\|p(y_i) - p(y_c)\|) / \sum_{j=1}^n K_j^s K^r(\|p(y_j) - p(y_c)\|) \quad (33)$$

where $p(y_i)$ represents the vectorized patch centered at y_i (whose size can be different from the size of the image patch y), and where $K^s \in \mathbb{R}_+^n$ is a spatial kernel used to give more weight to pixels closer to the central pixel and $K^r : u \mapsto \exp(-u^2/h^2)$, where h is the range smoothing hyperparameter. As h increases, K^r approaches the constant function and the filter has a behavior close to a Gaussian smoothing filter. On the contrary, as h decreases, K^r reinforces the weighting of pixels with high patch similarity and the resulting filter becomes nonlinear and more edge-preserving.

The resulting N(on)-L(ocal) Means [13] algorithm has had a tremendous influence on the denoising field and above for the reason that it is capable of effectively process redundant information in images with the help of patches. The central idea is that, in a natural image, a patch rarely appears alone and that almost perfect copies can be found in its surroundings [150]. NL Means has paved the way for a brand new class of denoising algorithms that exploits the self-similarity assumption [30, 67, 73, 48, 93, 33, 34, 58, 65, 31]. Nevertheless, determining the optimal weights θ of convex combinations for image denoising still remained an open question, although patch self-similarity appears to be a key element for obtaining competitive results. In [61], Jin *et al.* addressed this question starting from [13, 65, 66]. They achieved state-of-the-art performances among methods restricted to convex combinations of pixels via the establishment of an upper bound for the optimal weights θ . Adopting a risk minimization approach and constraining the weights θ to encode a convex combination of pixels, the optimal weights in the case of Gaussian noise (i.e., $y \sim \mathcal{N}(x, \sigma^2 I_n)$) and in the ℓ_2 sense, are:

$$\theta^* = \arg \min_{\theta \in \mathbb{R}^n} \mathcal{R}(f_\theta) \quad \text{s.t.} \quad \mathbf{1}_n^\top \theta = 1 \text{ and } \theta \succeq 0, \quad (34)$$

where $\mathcal{R}(f_\theta) := \mathbb{E}_y((f_\theta(y) - x_c)^2)$ is the quadratic risk. By leveraging a bias-variance decomposition, the statistical risk $\mathcal{R}(f_\theta) = (\mathbb{E}_y(f_\theta(y) - x_c))^2 + \mathbb{V}_y(f_\theta(y) - x_c)$, under convex constraints, has a closed-form expression which can be upper bounded using the triangle inequality:

$$\mathcal{R}(f_\theta) \leq f_\theta(|x - x_c|)^2 + \sigma^2 \|\theta\|_2^2 = \theta^\top Q \theta, \quad (35)$$

where the subtraction applies element-wise and $Q := |x - x_c| |x - x_c|^\top + \sigma^2 I_n$ is a symmetric positive definite matrix. Finally, OWF [61] proposes to approximate the optimal weights θ^* defined in (34) by the ones minimizing the upper bound (35) under convex constraints. This amounts to solving a quadratic program and the resulting weights have a closed-form expression [61].

3.2 Sparsity methods

Sparsity methods have emerged as powerful tools for image denoising, offering effective ways to restore images corrupted by noise while preserving important structural information. These methods exploit the inherent sparsity of natural images, which implies that most image patches can be efficiently represented by a small number of non-zero coefficients in a suitable transform domain.

3.2.1 Sparsity in a fixed basis

Sparsity of patches in a fixed basis refers to the property that most image patches can be efficiently represented using only a small number of non-zero coefficients in a predetermined basis. A basis is a set of linearly independent vectors,

or patches, that spans the entire signal space. It should be distinguished from the term dictionary, for which the vectors are not necessarily linearly independent.

Formally, we denote by $x \in \mathbb{R}^n$ a vectorized clean *natural* image patch of size n . According to the sparsity assumption, there exists a fixed basis of vectors $\{b_i\}_{i \in \{1, \dots, n\}}$, where $b_i \in \mathbb{R}^n$, such that each clean patch x of a noise-free image can be exactly represented by a linear combination involving only a few basis vectors. Adopting the matrix notation where $B \in \mathbb{R}^{n \times n}$ is the matrix formed by stacking the basis vectors $\{b_i\}_{i \in \{1, \dots, n\}}$ along columns, the sparsity assumption reads:

$$\forall x \in \mathbb{R}^n, x \text{ is a natural patch} \Leftrightarrow \|B^{-1}x\|_0 \leq t_0, \quad (36)$$

where $\|\cdot\|_0$ is the ℓ_0 *pseudo* norm counting the non-zero elements of a vector, $t_0 \leq n$ is an hyperparameter controlling the sparsity and the entries of vector $B^{-1}x$ are the unique coefficients of the linear combination which generate patch x in basis B .

A general strategy for denoising a noisy patch y under the sparsity paradigm is then to find its closest sparse representation. The resulting optimization problem is as follows:

$$\arg \min_{x \in \mathbb{R}^n} \|y - x\| \quad \text{s.t.} \quad \|B^{-1}x\|_0 \leq t_0, \quad (37)$$

which is equivalent, thanks to the change of variable $x = B\theta$, to:

$$\arg \min_{\theta \in \mathbb{R}^n} \|y - B\theta\| \quad \text{s.t.} \quad \|\theta\|_0 \leq t_0. \quad (38)$$

Note that denoising under the sparsity assumption involves several poorly defined quantities, namely the number of non-zero coefficients t_0 for being considered sparse, the norm $\|\cdot\|$ to choose for assessing the patch proximity and especially the fixed basis B . Common choices for the basis B include the discrete cosine transform (DCT) or wavelets [140, 27, 91] as discussed below.

Finally, note that solving (38) exactly in the general case where B is a dictionary can be done in a finite amount of computation but this is a NP-hard problem. The algorithms designed to find an approximate solution of (38) are called pursuit algorithms and include basis pursuit, FOCUSS, or matching pursuit methods [23, 47, 95].

TV denoising Total variation (TV) denoising [118] is finally one of the most famous image denoising algorithm, appreciated for its edge-preserving properties. In its original form [118], a TV denoiser is defined as a function $f : \mathbb{R}^n \times \mathbb{R}_*^+ \mapsto \mathbb{R}^n$ that solves the following equality-constrained problem:

$$f_{\text{TV}}(y, \sigma) = \arg \min_{x \in \mathbb{R}^n} \|x\|_{\text{TV}} \quad \text{s.t.} \quad \|y - x\|_2^2 = n\sigma^2 \quad (39)$$

where $\|x\|_{\text{TV}} := \|\nabla x\|_2$ is the total variation of $x \in \mathbb{R}^n$.

DCT and DWT denoiser The discrete cosine transform (DCT) algorithm [140] is a simple and efficient sparsity-based method for image denoising. The DCT is closely related to the discrete Fourier transform, but involves only real numbers. This basis yields several pleasant mathematical properties; in particular it is orthogonal, meaning that $B^{-1} = B^\top$, and there exists a fast algorithm [24] for computing the decomposition of any vector in this basis, just like the FFT algorithm (Fast Fourier Transform). Secondly, the DCT basis is experimentally near optimal to approximate *natural* patches in the sense that it ensures maximum energy compression of data in the first components. In order quickly approach the solution of the sparsity optimization problem (38), a simple procedure [140] consists in computing the DCT of the noisy patch, that is $B^{-1}y$, and setting to zero all small coefficients (below 3σ in absolute value for Gaussian noise). At the end, all denoised patches are repositioned at their initial locations and averaged to produce the final denoised image. The discrete wavelet transform (DWT) is another example of set of orthogonal bases B that has been successfully utilized for image denoising [27, 91, 109, 20]. Contrary to the DCT, these bases are itself sparse, in the sense that a majority of coefficients of the basis vectors are zero. This property makes decomposition calculations particularly fast. Interestingly, the DWT has the ability to decompose an image into different frequency subbands at different scales. The high-frequency subbands capture the local details and fine structures, while the low-frequency subbands represent the global structures and smooth regions. The Bayesian denoising method BLS-GSM [109] achieved state-of-the-art performance at the time by carefully adapting noise processing to each scale.

BM3D BM3D (Block Matching 3D) [30] is a powerful and widely acclaimed algorithm that has achieved remarkable success in image denoising. BM3D considerably improves the performance of pure sparsity methods such as [140, 27, 91] by adding another key element, namely the grouping technique, exploiting the redundancy present in natural images. Figure 7 illustrates this popular technique in image denoising [30, 73, 48, 93, 33, 34, 58]. It basically consists

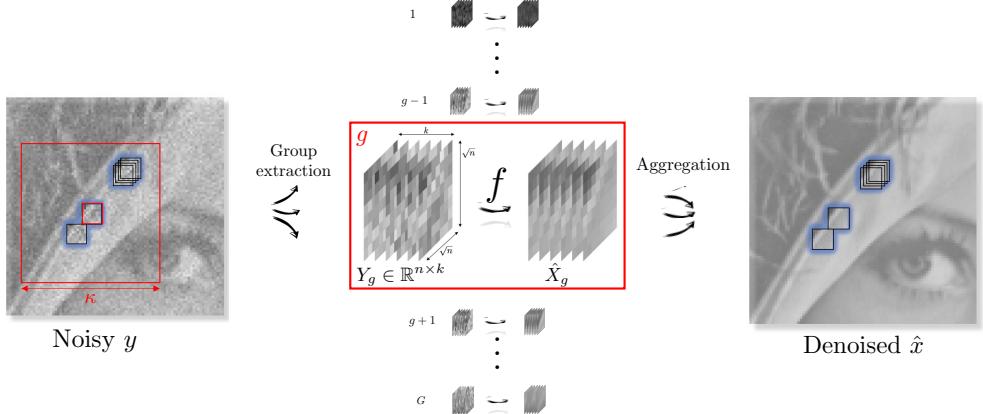


Figure 7: Illustration of the grouping technique for image denoising.

in grouping image patches based on patch resemblance into 3D blocks, also referred to as similarity matrices, in order to perform collaborative filtering. During the first stage of BM3D, the denoising of the independent 3D blocks is performed by assuming a local sparse representation in a transform domain. Essentially, BM3D solves the same optimization problem as (38) with the only difference that it involves groups of patches instead of processing each patch separately. Among the possible bases of decomposition for processing the groups, 3D-DCT is frequently used and the same fast procedure as [140] that consists in canceling all coefficients below a given threshold is adopted for fast resolution. As for the second stage, Wiener filtering is leveraged for collaborative denoising. Overall, the remarkable denoising performance of BM3D algorithm has made it a widely adopted and benchmark denoising method in applications.

3.2.2 Sparsity on a learned dictionary

The use of a fixed basis such as the DCT or the DWT for sparsity-based image denoising has the advantage of being both general and fast. However, it is not easy to know in advance which basis to choose for achieving the best denoising results on a given image, although some attempts have been made in this direction [90, 8]. A more flexible approach is to directly adapt the decomposition to the input image by unsupervised learning.

KSVD KSVD [38] is a popular unsupervised learning algorithm for creating an adaptive dictionary for sparse representations. Formally, let $Y \in \mathbb{R}^{n \times N}$ be the matrix gathering all the N overlapping vectorized patches of size n of a noisy image, $D \in \mathbb{R}^{n \times d}$ an overcomplete dictionary (a set of $d \geq n$ patches, also referred to as atoms, spanning the entire signal space), and $\Theta \in \mathbb{R}^{d \times N}$ the sparse coefficients of the linear combinations. The optimization problem at the heart of KSVD [38] is the following:

$$\arg \min_{D, \Theta} \|Y - D\Theta\|_F^2 \text{ s.t. } \|\Theta_{\cdot, j}\|_0 \leq t_0, \forall j \in \{1, \dots, N\}, \quad (40)$$

where $t_0 \leq n$ is an hyperparameter controlling the sparsity of the linear combinations. Note that this objective is very similar with (38), with the difference that the dictionary D is no longer fixed but fully integrated to the learning process. The resolution of (40) is achieved via an alternating optimization algorithm by iteratively fixing dictionary D and coefficients Θ as follows:

SPARSE CODING: For a dictionary D fixed, solving (40) amounts to solving N independent subproblems for which any pursuit algorithm [23, 47, 95] can be leveraged for resolution. Indeed, we have:

$$\|Y - D\Theta\|_F^2 = \sum_{j=1}^N \|Y_{\cdot, j} - D\Theta_{\cdot, j}\|_2^2. \quad (41)$$

DICTIONARY UPDATING: Assuming that both D and Θ are fixed, except one column in the dictionary $D_{\cdot, k}$ (atom k) and its corresponding coefficients $\Theta_{k, \cdot}$, (41) can be rewritten as:

$$\|Y - D\Theta\|_F^2 = \|Y - \sum_{j=1}^d D_{\cdot, j} \Theta_{j, \cdot}\|_F^2 = \|E_k - D_{\cdot, k} \Theta_{k, \cdot}\|_F^2, \quad (42)$$

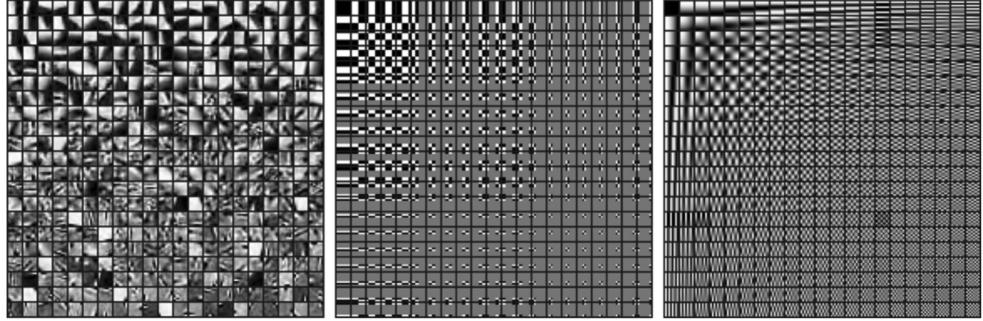


Figure 8: Example of the learned dictionary by KSVD algorithm [38] (left), the overcomplete separable Haar dictionary (middle) and the overcomplete DCT dictionary (right). Source: [4].

where $E_k := Y - \sum_{j \neq k} D_{\cdot,j} \Theta_{j,\cdot}$. In other words, it amounts to finding a matrix of rank 1 minimizing the ℓ_2 distance with E_k . The solution can be computed using the singular value decomposition (SVD) according to Eckart-Young theorem [36]. Formally, let u , v and s be the first left-singular vector, right-singular vector and singular value of E_k , respectively. Then, its closest matrix of rank 1, in the ℓ_2 sense, is simply svv^\top . However, it is very likely that the first right-singular value of E_k is not sparse; therefore, it cannot be used to update coefficients $\Theta_{j,\cdot}$. The trick of KSVD [38] consists in modifying only the nonzero entries of $\Theta_{j,\cdot}$, thus ensuring that it stays sparse. Mathematically, it comes down to computing the SVD of E_k for which the columns corresponding to a zero coefficient in $\Theta_{j,\cdot}$ have been deleted.

Following this alternating optimization procedure, KSVD [38] converges after a few iterations. At the end, all denoised patches are repositioned at their initial locations and averaged to produce the final denoised image. Interestingly, the dictionary learned in an unsupervised fashion can be displayed (see Fig. 8). In spite of its great theoretical interest, KSVD [38] is unfortunately little used in practice, due to its tedious optimization procedure, its difficult-to-set hyperparameters and its limited performance compared to BM3D [30].

Simultaneous sparse coding (SSC) from a low-rank view point While KSVD [38] tries to learn a general overcomplete dictionary, for which every patch of the input image can be reconstructed using only a few atoms, some authors argue that the dictionary should be adaptive to groups of similar patches to improve the performance of sparse representation models [93, 33, 48, 58]. Indeed, a major drawback of (40) is the assumption about the independence between sparsely-coded patches. In order to better exploit the self-similarity of patches in an image, a refinement consists in constraining the similar patches to share the same atoms in their sparse coding. To that end, the optimization problem (40) can be slightly adapted to groups of similar patches, making it even more restricted:

$$\arg \min_{D, \Theta} \|Y - D\Theta\|_F^2 \quad \text{s.t.} \quad \|\Theta\|_0 \leq t_0, \quad (43)$$

where $Y \in \mathbb{R}^{n \times k}$ is a similarity matrix, $D \in \mathbb{R}^{n \times d}$ a dictionary, $\Theta \in \mathbb{R}^{d \times k}$ the sparse coding, and where the matrix *pseudo* ℓ_0 norm counts the number of non-zero rows. Note in particular that, subject to dimensional compatibility, any admissible point Θ for (43) is also admissible for (40). Moreover, it is worth noting that the dictionary becomes strictly local under the group sparse representation contrary to (40). As a matter of fact, solving (43) amounts to solving a low-rank approximation of Y if we denote $X = D\Theta$:

$$\arg \min_X \|Y - X\|_F^2 \quad \text{s.t.} \quad \text{rank}(X) \leq t_0, \quad (44)$$

for which the solution is expressed with the help of the singular value decomposition (SVD) of Y according to Eckart-Young theorem [36]. In particular, considering the Lagrangian unconstrained formulation of (44) with hyperparameter $\gamma \geq 0$, we have (see proof in [56, 58]):

$$U\varphi_{\text{hard}, \sqrt{\gamma}}(S)V^\top = \arg \min_X \|Y - X\|_F^2 + \gamma \text{rank}(X), \quad (45)$$

where $Y = USV^\top$ is the SVD of Y and $\varphi_{\text{hard}, \gamma}$ denotes the hard shrinkage operator that applies element-wise $\varphi_{\text{hard}, \gamma}(x) = x\mathbf{1}_{\mathbb{R} \setminus [-\gamma, \gamma]}(x)$. Equation (45) is at the core of PLR algorithm [58] where the value of $\gamma = 2.25 k \sigma^2$ is recommended experimentally for denoising Y when it is corrupted by additive white Gaussian noise (AWGN) of variance σ^2 .

A relaxation of (43) is proposed by LSSC [93] through a so-called grouped-sparsity regularizer to encourage the alignment of sparse coefficients along the row direction, without imposing it as a hard constraint. Interestingly, this

relaxation has also a low-rank interpretation from a variance estimation perspective [33]. Specifically, it amounts to solving a nuclear norm minimization problem (NNM) which has a closed-form solution [16]:

$$U\varphi_{\text{soft},\gamma}(S)V^\top = \arg \min_X \frac{1}{2}\|Y - X\|_F^2 + \gamma\|X\|_* , \quad (46)$$

where $\|X\|_*$ denotes nuclear norm (sum of the singular values) and $\varphi_{\text{soft},\gamma}$ denotes the soft shrinkage operator that applies element-wise $\varphi_{\text{soft},\gamma}(x) = \text{sign}(x) \cdot \max(|x| - \gamma, 0)$. More recently, WNNM [48] refines the NNM problem (46) by assigning different weights to the singular values. Combined with iterative regularization technique [104] and multiple passes of denoising, WNNM achieves state-of-the-art performances.

3.3 Bayesian methods coupled with a Gaussian model

Bayesian methods coupled with a Gaussian model form a powerful framework for probabilistic modeling in image denoising [2, 151, 73, 85, 141, 74]. The Gaussian model (or a mixture of Gaussians) is widely used due to its simplicity and flexibility in capturing a wide range of continuous data, signal being no exception. In this section, we review two algorithms [151, 73] that are major representatives of Bayesian modeling under Gaussian prior.

EPLL Based on the observation that learning good image priors over whole images is challenging, EPLL [151] proposes to transpose the modeling of an image prior back to the prior modeling of small image patches, which is assumed to be an easier task. Specifically, in the case of additive white Gaussian noise (AWGN) of variance σ^2 , the maximum a posteriori (MAP) of the clean patch x_i given its noisy patch $y_i \sim \mathcal{N}(x, \sigma^2 I_n)$ and an arbitrary image patch prior p leads to the minimization of an energy (via Bayes' rule):

$$E_i(x_i, y_i) := \frac{1}{2\sigma^2} \|x_i - y_i\|_2^2 - \log p(x_i) . \quad (47)$$

In order to extend this energy to the whole image, EPLL [59] defines the global energy of a full image x , given y , by the average energy of energies all its N overlapping patches:

$$E(x, y) := \frac{1}{N} \sum_{i=1}^N E_i(x_i, y_i) \approx \frac{n}{2\sigma^2 N} \|x - y\|_2^2 - \text{EPLL}_p(x) , \quad (48)$$

where $\text{EPLL}_p(x) := \sum_i \log p(x_i)/N$ is the expected patch log likelihood (EPLL). Note that the approximation is legitimate because a noisy pixel belongs to exactly n overlapping patches, with the exception of pixels located close to the borders, which are neglected for the sake of simplicity. Searching for the image x with smallest global energy, the resulting optimization problem finally reads [151]:

$$\arg \min_x \frac{n}{2N\sigma^2} \|x - y\|_2^2 - \text{EPLL}_p(x) . \quad (49)$$

Thus, the expected patch log likelihood (EPLL) acts as a regularization term in (49). Direct optimization of the cost function (49) may be very hard, depending on the prior used. That is why, half quadratic splitting [46] is leveraged for efficient resolution by introducing auxiliary variables. Note that this iterative optimization method shares close links with the alternating direction method of multipliers (ADMM) [10].

Although this framework allows the use of patch priors p of any sort, the authors [151] propose to leverage a surprisingly simple Gaussian mixture model: $p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x; \mu_k, \Sigma_k)$, where π_k are the mixing weights for each of the mixture component and the μ_k and Σ_k are the corresponding mean and covariance matrix. In the original paper [151], the parameters π_k , μ_k and Σ_k for $K = 200$ components are estimated in a supervised fashion by maximum likelihood estimation (MLE) over a set of two millions clean patches collected from BSD dataset [99] using the expectation–maximization (EM) algorithm for optimization. However, a recent work [85] shows that Gaussian mixture parameters can also be estimated unsupervisedly directly from the noisy input image itself, resulting in even better image denoising performance than its supervised counterpart.

Despite their significant theoretical relevance, EPLL [151] and its variants [85, 32] have not gained much popularity in practical applications primarily due to their cumbersome optimization procedures when compared to the well-established BM3D [30] algorithm. Finally, note that the EPLL approach [151] shares some similarities with the Field-of-Experts framework [117] designed previously where the parameterized density function, namely the Gaussian mixture model, is replaced by a Product-of-Experts [55] that exploits non-linear functions of many linear filter responses and where optimization is essentially performed through gradient ascent.

NL-Bayes: The N(on)-L(ocal) Bayes [73] algorithm combines the concepts of Bayesian modeling and self-similarity [150] which appears to be key element for achieving state-of-the-art results. Formally, let $y \sim \mathcal{N}(x, \sigma^2 I_n)$ be a noisy image patch corrupted by Gaussian noise of variance σ^2 . Arbitrarily setting a multivariate Gaussian prior on the clean patch x , i.e. $p(x) = \mathcal{N}(x; \mu, \Sigma)$, where the mean μ and covariance matrix Σ are to be estimated, the maximum a posteriori (MAP), computed using Bayes' rule, is the solution of the following optimization problem:

$$\begin{aligned}\hat{x}_{MAP} &= \arg \max_x \frac{1}{2\sigma^2} \|x - y\|_2^2 + \frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu) \\ &= \mu + \Sigma (\Sigma + \sigma^2 I_n)^{-1} (y - \mu).\end{aligned}\quad (50)$$

NL-Bayes [73] proposes to construct the prior $p(x)$ from the group of image patches similar to x . Specifically, let $X \in \mathbb{R}^{n \times k}$ be the similarity matrix of x . Then, μ is estimated by the average patch and Σ is estimated by the empirical covariance matrix of the group, that is:

$$\mu_X := \frac{1}{k} X \mathbf{1}_k \text{ and } \Sigma_X := \frac{1}{k} (X - \mu_X \mathbf{1}_k^\top)(X - \mu_X \mathbf{1}_k^\top)^\top. \quad (51)$$

But of course, the true similarity matrix X is unknown and can only be deduced from its noisy version Y . To that end, unbiased estimates are leveraged as a first approximation, namely $\mu_Y = \frac{1}{k} Y \mathbf{1}_k$ and $\Sigma_Y = \frac{1}{k} (Y - \mu_Y \mathbf{1}_k^\top)^\top - \sigma^2 I_n$. Using equation (50) and the two estimates μ_Y and Σ_Y , each noisy patch of a given similarity matrix can then be denoised. Viewing this denoising step as a function f that processes similarity matrices, NL-Bayes falls into the category of non-local denoisers (see Fig. 7).

After reprojection and aggregation by average of all patch estimates, a first denoised image is built which is exploited to refine the priors $p(x)$ for each group of similar patches. Actually, using the equation (51) with approximate similarity matrices gives better results in practice than with the unbiased estimates of the first step. Repeating this second stage again and again, taking advantage of the availability of a supposedly better image estimate than in the previous step, does not bring experimentally any improvements unfortunately. That is why, the algorithm stops after two steps.

NL-Bayes [73] compares favorably with BM3D [30] and is as competitive as in terms of speed which makes it an interesting alternative for practical image denoising [74].

3.4 Deep learning-based methods

In recent years, attempts have been made to reconcile unsupervised learning and deep neural networks in image denoising [111, 78, 6]. Major representatives are Deep Image Prior (DIP) [78] and Self2self [111].

Deep Image Prior Deep Image Prior [78] adopts a non-intuitive strategy which consists in training a convolutional neural network with U-net architecture f_θ to predict the input noisy image $y \in \mathbb{R}^n$ from a single realization of pure uniform noise $u \sim \mathcal{U}([0, 1]^{n \times C})$, where C denotes the number of feature maps (e.g. $C = 32$):

$$\arg \min_\theta \|f_\theta(u) - y\|_2^2. \quad (52)$$

By early stopping the optimization process based on gradient descent in order to avoid perfect reconstruction of the noisy image y , it is observed that $f_\theta(u)$ may be surprisingly very close to the true image x in practice. According to the authors [78], this intriguing phenomenon is an evidence that realistic images are naturally promoted by certain types of neural networks. Indeed, equation (52) is only composed of the data-fidelity term without any regularizer, which suggests that network architectures actually encode an implicit image prior.

Some refinements of (52) were proposed afterwards to enhance the performance of DIP [78] by adding nevertheless an explicit prior. For example, combining DIP [78] with the traditional TV regularization [118] was investigated in [86] with relative quality improvement, at the price of an extra hyperparameter balancing the data-fidelity term and the regularization term. Another interesting alternative [100] consists in explicitly regularizing DIP [78] using an existing unsupervised denoising algorithm such as BM3D [30]. The concept of regularization by denoising (RED) [114] is indeed an alternative to Plug-and-Play Prior [135] which enables to harness the implicit prior learned by a denoiser to any data-fidelity term, while avoiding the need to differentiate the chosen denoiser. Other variants of DIP [78] for improved performance include [40, 62, 26, 120].

Self2Self More recently, Self2Self [111] considers the pretext task of inpainting to tackle image denoising, namely:

$$\theta^* = \arg \min_\theta \mathbb{E}_b \|(1 - b) \odot (f_\theta(b \odot y) - y)\|_2^2, \quad (53)$$

where \odot denotes the Hadamard product and $b \in \{0, 1\}^n$ is a random vector whose components follow independent Bernoulli distributions with probability $p \in (0, 1)$. This time, dropout [126] and sampling are exploited as regularization techniques for avoiding the convergence to the constant function $f_\theta(\cdot) = y$. During the inference step, about a hundred of artificially simulated samples $f_{\theta^*}(b \odot y)$ are averaged for final estimation. Note that Self2Self [111] shares some similarities with Noise2Self [6] as f_θ is learned following the blind-spot strategy. To our knowledge, Self2Self [111] is the current state-of-the-art unsupervised deep learning-based denoiser.

Although promising, the aforementioned deep unsupervised learning methods are still limited in terms of performance and especially in terms of computational cost compared to the patch-based and non-local methods [30, 73, 52, 48, 33, 34, 64, 61]. Indeed, deep learning-based methods use the time-consuming gradient descent algorithm for optimization, whereas traditional ones have in general closed-form solutions, which speeds up learning.

4 Normalization-equivariance property of denoising methods

In many information processing systems, it may be desirable to ensure that any change of the input, whether by shifting or scaling, results in a corresponding change in the system response. While deep neural networks are gradually replacing all traditional automatic processing methods, they surprisingly do not guarantee such normalization-equivariance (scale and shift) property, which can be detrimental in many applications.

Sometimes wrongly confused with the invariance property which designates the characteristic of a function f not to be affected by a specific transformation \mathcal{T} applied beforehand, the equivariance property, on the other hand, means that f reacts in accordance with \mathcal{T} . Formally, invariance is $f \circ \mathcal{T} = f$ whereas equivariance reads $f \circ \mathcal{T} = \mathcal{T} \circ f$, where \circ denotes the function composition operator. Both invariance and equivariance play a crucial role in many areas of study, including computer vision and signal processing and have recently been studied in various settings for deep-learning-based models [28, 63, 123, 7, 98, 138, 134, 49, 130, 43, 15, 44, 75].

In this section, we focus on the equivariance of supervised and unsupervised image denoising methods f_θ to a specific transformation \mathcal{T} , namely normalization.

4.1 Normalization-equivariance of conventional denoisers

We start with formal definitions of the different types of equivariances studied in this chapter. Please note that our definition of “scale” and “shift” may differ from the definition given by some authors in the image processing literature.

Definition 1. A function $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is said to be:

- *\mathcal{T} -equivariant if $f \circ \mathcal{T} = \mathcal{T} \circ f$,*
- *scale-equivariant if $\forall x \in \mathbb{R}^n, \forall a \in \mathbb{R}_*^+, f(ax) = af(x)$,*
- *shift-equivariant if $\forall x \in \mathbb{R}^n, \forall b \in \mathbb{R}, f(x + b) = f(x) + b$,*
- *normalization-equivariant if it is both scale-equivariant and shift-equivariant:*

$$\forall x \in \mathbb{R}^n, \forall a \in \mathbb{R}_*^+, \forall b \in \mathbb{R}, f(ax + b) = af(x) + b,$$

where addition with the scalar shift b is applied element-wise.

Note that the *scale-equivariance* property is more often referred to as positive homogeneity in pure mathematics. S. Mohan et al. [102] revealed that scale-equivariant neural networks could simply be built by removing the additive constant (“bias”) terms in CNNs with ReLU activation functions without affecting performance. Moreover, they showed that a much better generalization at noise levels outside the training range was ensured by these networks as a spectacular outcome.

A (“blind”) denoiser is basically a function $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ which, given a noisy image $y \in \mathbb{R}^n$, tries to map the corresponding noise-free image $x \in \mathbb{R}^n$. Since scaling up an image by a positive factor a or adding it up a constant shift b does not change its contents, it is natural to expect scale and shift equivariance, *i.e.* normalization equivariance, from the denoising procedure emulated by f .

The most basic methods for image denoising are the smoothing filters (see Section III.A). It turns out that these “blind” denoisers process images by convex combinations of pixels and therefore all implement a *normalization-equivariant* function. More generally, one can prove that a linear filter is *normalization-equivariant* if and only if its coefficients add up to 1. In others words, *normalization-equivariant* linear filters process images by affine combinations of pixels.

Table 1: Equivariance properties of several image denoisers (left: traditional, right: deep learning-based)

	Traditional						
	TV	NLM	NL-Ridge	LIChi	DCT	BM3D	WNNM
Scale-eq	✓	✓	✓	✓	✓	✓	✓
Shift-eq	✓	✓	✓	✓	✗	✗	✗
	Deep-learning						
	DnCNN	NLRN	SCUNet	Restormer	DCT2net	DRUNet	
Scale-eq	✗	✗	✗	✗	✓	✓	
Shift-eq	✗	✗	✗	✗	✗	✗	

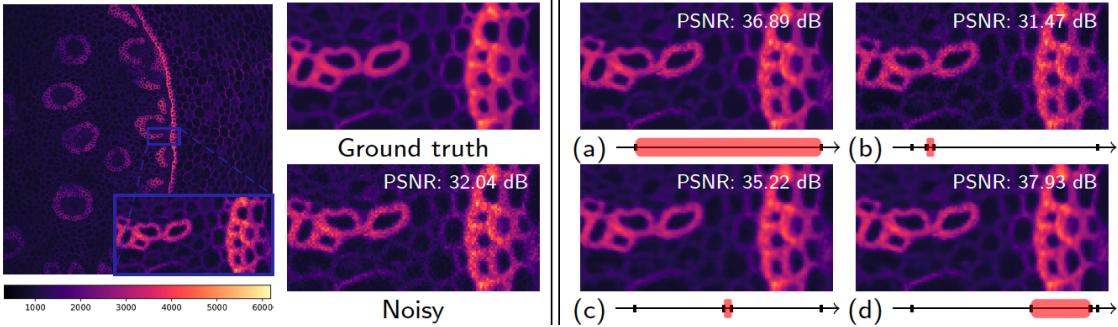


Figure 9: Impact of normalization for deep-learning-based denoising (DnCNN). All four intervals are subsets of the interval $[0, 1]$. Source: [54].

As such, $f_{\text{NLM}} = y^\top \theta$ (with the weights θ given in (33)) is a *normalization-equivariant* function. More recently, NL-Ridge [52] and LIChi [53] propose to process images by linear combinations of similar patches and achieves state-of-the-art performance in unsupervised denoising. When restricting the coefficients of the combinations to sum to 1, that is imposing affine combination constraints, the resulting algorithms encode *normalization-equivariant* functions as well.

4.2 The case of neural networks

Deep learning hides a subtlety about normalization equivariance that deserves to be highlighted. Usually, the weights of neural networks are learned on a training set containing data all normalized to the same arbitrary interval $[s_0, t_0]$. At inference, unseen data are processed within the interval $[s_0, t_0]$ via a $s-t$ linear normalization with $s_0 \leq s < t \leq t_0$ denoted $\mathcal{T}_{s,t}$ and defined by:

$$\mathcal{T}_{s,t} : y \mapsto (t-s) \frac{y - \min(y)}{\max(y) - \min(y)} + s. \quad (54)$$

Note that this transform is actually the unique linear one with positive slope that exactly bounds the output to $[s, t]$. The data is then passed to the trained network and its response is finally returned to the original range via the inverse operator $\mathcal{T}_{s,t}^{-1}$. This proven pipeline is actually relevant in light of the following proposition.

Proposition 1. $\forall s < t \in \mathbb{R}, \forall f : \mathbb{R}^n \mapsto \mathbb{R}^m, \mathcal{T}_{s,t}^{-1} \circ f \circ \mathcal{T}_{s,t}$ is a normalization-equivariant function.

Thus, if f is not normalization-equivariant, $\mathcal{T}_{s,t}^{-1} \circ f \circ \mathcal{T}_{s,t}$ is guaranteed to be. While normalization-equivariance appears to be solved, a question is still remaining: how to choose the hyperparameters s and t for a given function f ? Obviously, a natural choice for neural networks is to take the same parameters s and t as in the learning phase whatever the input image is, *i.e.* $s = s_0$ and $t = t_0$, but are they really optimal? The answer to this question is generally negative. It is particularly true in image denoising when the noise level at inference differs from the noise level on which the neural network was trained on. Indeed, as showed by [136], careful choices on parameters $s > s_0$ and $t < t_0$ can strongly mitigate the effect of the distribution gap (see Figure 9 and Figure 5 in [54]). This suggests that there are always inherent performance leaks for deep neural networks due to the two degrees of freedom induced by the normalization (*i.e.*, choice of s and choice of t). In addition, this poor conditioning can be a source of misinterpretation in critical applications.

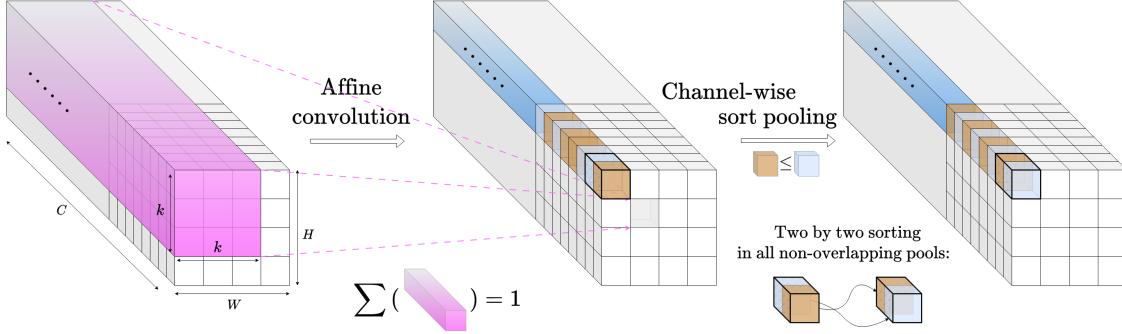


Figure 10: Illustration of the alternative for replacing the traditional scheme “convolution + element-wise activation function” in CNNs: affine convolutions supersede ordinary ones by restricting the coefficients of each kernel to sum to one and the proposed sort pooling patterns introduce nonlinearities by sorting two by two the pre-activated neurons along the channels. Source: [54].

4.3 Categorizing image denoisers

Table 1 summarizes the equivariance properties of several popular denoisers, either conventional [118, 13, 52, 53, 140, 30, 48] or deep-learning-based [146, 145, 84, 81]. Interestingly, if *scale-equivariance* is generally guaranteed for traditional denoisers, not all of them are equivariant to shifts. In particular, the widely used algorithms DCT [140] and BM3D [30] are sensitive to offsets, mainly because the hard thresholding function at their core is not *shift-equivariant*.

Regarding the deep-learning-based networks, only DRUNet [145] is insensitive to scale because it is a bias-free convolutional neural network with only ReLU activation functions [102]. In particular, all transformer models [142, 107, 84, 81, 22, 144], even bias-free, are not *scale-equivariant* due to their inherent attention-based modules. To solve this issue, an approach was proposed in [54] to adapt the architecture of existing neural networks so that normalization equivariance holds by design. The authors argue that the classical pattern ”conv+ReLU” can be favorably replaced affine convolutions that ensure that all coefficients of the convolutional kernels sum to one in one hand, and the other hand, channel-wise sort pooling nonlinearities as a substitute for all activation functions that apply element-wise, including ReLU or sigmoid functions (see Fig. 10). These two architectural modifications do preserve normalization-equivariance without loss of performance, and provide much better generalization across noise levels in practice because they can naturally adapt to the best-performing interval [54] (see Figure 9 and Figure 5 in [54]).

5 Conclusion

In this paper, we have surveyed supervised and unsupervised image denoising methods, categorized the methods based on their equivariance properties, and summarized quantitatively their performances assessed on popular benchmarks, such as Set12 and BSD68. It is clearly established that the traditional unsupervised methods [30, 73, 48, 53] outperform the deep-learning counterparts, so far. They are particularly recommended when it is not possible to collect enough high-quality dataset for training denoising models or too much time consuming. In return, it is also clear that supervised methods based on CNNs boosted with attention modules [144] surpass top-rank traditional methods [48, 53] (see Fig. 1), capable of recovering details barely perceptible by a human. The question is which architecture will prevail over the next ten years. Transformer-based methods show great promise for image denoising and are likely to play an important role in the future. Meanwhile, let us not bury too quickly the other architectures as comebacks are still possible [89, 132]. The potential of all these frameworks has not yet been fully explored and open challenges and promising research directions for image denoising in the coming years. Nevertheless, in view of the recent spectacular results, it is legitimate to wonder whether we are approaching the theoretical limit of denoising performance, which could reopen the debate on whether image denoising is close to death [21, 79, 80].

References

- [1] R. Achddou, Y. Gousseau, and S. Ladjal. Fully synthetic training for image restoration tasks. *Computer Vision and Image Understanding*, 233:103723, 2023.
- [2] C. Aguerrebere, A. Almansa, Y. Gousseau, J. Delon, and P. Musé. A hyperprior Bayesian approach for solving image inverse problems. *HAL preprint*, 2014.

- [3] E. Agustsson and R. Timofte. NTIRE 2017 Challenge on single image super-resolution: Dataset and study. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, 2017.
- [4] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.
- [5] S. Anwar and N. Barnes. Real image denoising with feature attention. In *International Conference on Computer Vision (ICCV)*, October 2019.
- [6] J. Batson and L. Royer. Noise2Self: Blind denoising by self-supervision. In *International Conference on Machine Learning (ICML)*, volume 97, pages 524–533, 2019.
- [7] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13(1):2453, 2022.
- [8] T. Blu and F. Luisier. The SURE-LET approach to image denoising. *IEEE Transactions on Image Processing*, 16(11):2778–2786, 2007.
- [9] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J.-B. Sibarita, and J. Salamero. Patch-based nonlocal functional for denoising fluorescence microscopy image sequences. *IEEE Transactions on Medical Imaging*, 29(2):442–454, 2009.
- [10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Now Foundations and Trends, 2011.
- [11] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron. Unprocessing images for learned raw denoising. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11028–11037, 2019.
- [12] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 60–65, 2005.
- [13] A. Buades, B. Coll, and J.-M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005.
- [14] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2392–2399, 2012.
- [15] G. Böckman, F. Kahla, and A. Flinth. ZZ-Net: A universal rotation equivariant architecture for 2D point clouds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10966–10975, 2022.
- [16] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
- [17] Y. Cai, X. Hu, H. Wang, Y. Zhang, H. Pfister, and D. Wei. Learning to generate realistic noisy images via pixel-level noise-aware adversarial training. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 3259–3270, 2021.
- [18] A. F. Calvarons. Improved Noise2Noise denoising with limited data. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 796–805, 2021.
- [19] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.
- [20] V. Chappelier and C. Guillemot. Oriented wavelet transform for image compression and denoising. *IEEE Transactions on Image Processing*, 15(10):2892–2903, 2006.
- [21] P. Chatterjee and P. Milanfar. Is denoising dead? *IEEE Transactions on Image Processing*, 19(4):895–911, 2010.
- [22] L. Chen, X. Chu, X. Zhang, and J. Sun. Simple baselines for image restoration. In *European Conference on Computer Vision (ECCV)*, pages 17–33, 2022.
- [23] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.
- [24] W.-H. Chen, C. Smith, and S. Fralick. A fast computational algorithm for the Discrete Cosine Transform. *IEEE Transactions on Communications*, 25(9):1004–1009, 1977.
- [25] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1256–1272, 2017.
- [26] Z. Cheng, M. Gadelha, S. Maji, and D. Sheldon. A Bayesian perspective on the Deep Image Prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

- [27] H. A. Chipman, E. D. Kolaczyk, and R. E. McCulloch. Adaptive Bayesian wavelet shrinkage. *Journal of the American Statistical Association*, 92(440):1413–1421, 1997.
- [28] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, volume 48, pages 2990–2999, 2016.
- [29] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.
- [30] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.
- [31] C.-A. Deledalle, L. Denis, and F. Tupin. Iterative weighted maximum likelihood denoising with probabilistic patch-based weights. *IEEE Transactions on Image Processing*, 18(12):2661–2672, 2009.
- [32] C.-A. Deledalle, S. Parameswaran, and T. Q. Nguyen. Image denoising with generalized Gaussian mixture model patch priors. *SIAM Journal on Imaging Sciences*, 11(4):2568–2609, 2018.
- [33] W. Dong, G. Shi, and X. Li. Nonlocal image restoration with bilateral variance estimation: A low-rank approach. *IEEE Transactions on Image Processing*, 22(2):700–711, 2013.
- [34] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, 2013.
- [35] V. Duval, J.-F. Aujol, and Y. Gousseau. A bias-variance approach for the nonlocal means. *SIAM Journal on Imaging Sciences*, 4(2):760–788, 2011.
- [36] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.
- [37] B. Efron. Tweedie’s formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.
- [38] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.
- [39] M. Elad, B. Kawar, and G. Vaksman. Image denoising: The deep learning revolution and beyond—a survey paper. *SIAM Journal on Imaging Sciences*, 16(3):1594–1654, 2023.
- [40] R. Fermanian, M. Le Pendu, and C. Guillemot. Regularizing the Deep Image Prior with a learned denoiser for linear inverse problems. In *International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6, 2021.
- [41] A. Foi, V. Katkovnik, and K. Egiazarian. Pointwise shape-adaptive dct for high-quality deblocking of compressed color images. In *European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2006.
- [42] 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*, 17(10):1737–1754, 2008.
- [43] F. Fuchs, D. Worrall, V. Fischer, and M. Welling. SE(3)-Transformers: 3D roto-translation equivariant attention networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1970–1981, 2020.
- [44] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.
- [45] K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989.
- [46] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.
- [47] I. Gorodnitsky and B. Rao. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, 45(3):600–616, 1997.
- [48] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2862–2869, 2014.
- [49] D. K. Gupta, D. Arya, and E. Gavves. Rotation equivariant siamese networks for tracking. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12357–12366, 2021.
- [50] S. W. Hasinoff. Photon, Poisson noise. *Computer Vision, A Reference Guide*, 4(16):1, 2014.
- [51] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

- [52] S. Herbreteau and C. Kervrann. Towards a unified view of unsupervised non-local methods for image denoising: the NL-Ridge approach. In *IEEE International Conference on Image Processing (ICIP)*, pages 3376–3380, 2022.
- [53] S. Herbreteau and C. Kervrann. Unsupervised linear and iterative combinations of patches for image denoising. *arXiv preprint arXiv:2212.00422*, 2022.
- [54] S. Herbreteau, E. Moebel, and C. Kervrann. Normalization-equivariant neural networks with application to image denoising. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [55] G. Hinton. Products of experts. In *International Conference on Artificial Neural Networks (ICANN)*, volume 1, pages 1–6, 1999.
- [56] J.-B. Hiriart-Urruty and H. Y. Le. From eckart and young approximation to moreau envelopes and vice versa. *RAIRO-Operations Research-Recherche Opérationnelle*, 47(3):299–310, 2013.
- [57] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.
- [58] H. Hu, J. Froment, and Q. Liu. A note on patch-based low-rank minimization for fast image denoising. *Journal of Visual Communication and Image Representation*, 50:100–110, 2018.
- [59] S. Hurault, T. Ehret, and P. Arias. EPLL: an image denoising method using a Gaussian mixture model learned on a large set of patches. *Image Processing On Line*, 8:465–489, 2018.
- [60] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, volume 37, pages 448–456, 2015.
- [61] Q. Jin, I. Grama, C. Kervrann, and Q. Liu. Nonlocal means and optimal weights for noise removal. *SIAM Journal on Imaging Sciences*, 10(4):1878–1920, 2017.
- [62] Y. Jo, S. Y. Chun, and J. Choi. Rethinking Deep Image Prior for denoising. In *International Conference on Computer Vision (ICCV)*, pages 5087–5096, 2021.
- [63] N. Keriven and G. Peyré. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
- [64] C. Kervrann. PEWA: Patch-based exponentially weighted aggregation for image denoising. In *Advances in Neural Information Processing Systems (NIPS)*, volume 27, 2014.
- [65] C. Kervrann and J. Boulanger. Optimal spatial adaptation for patch-based image denoising. *IEEE Transactions on Image Processing*, 15(10):2866–2878, 2006.
- [66] C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image regularization and representation. *International Journal of Computer Vision*, 79(1):45–69, 2008.
- [67] C. Kervrann, J. Boulanger, and P. Coupé. Bayesian non-local means filter, image redundancy and adaptive dictionaries for noise removal. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 520–532, 2007.
- [68] K. Kim and J. C. Ye. Noise2Score: Tweedie’s approach to self-supervised image denoising without clean images. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 864–874, 2021.
- [69] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.
- [70] A. Krull, T.-O. Buchholz, and F. Jug. Noise2Void - Learning denoising from single noisy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2124–2132, 2019.
- [71] A. Krull, T. Vicar, M. Prakash, M. Lalit, and F. Jug. Probabilistic Noise2Void: Unsupervised content-aware denoising. *Frontiers in Computer Science*, 2:5, 2020.
- [72] S. Laine, T. Karras, J. Lehtinen, and T. Aila. High-quality self-supervised deep image denoising. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
- [73] M. Lebrun, A. Buades, and J. M. Morel. A nonlocal Bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, 6(3):1665–1688, 2013.
- [74] M. Lebrun, M. Colom, and J.-M. Morel. The Noise Clinic: a blind image denoising algorithm. *Image Processing On Line*, 5:1–54, 2015.
- [75] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

- [76] K. Lee and W.-K. Jeong. Noise2Kernel: Adaptive self-supervised blind denoising using a dilated convolutional kernel architecture. *Sensors*, 22(11):4255, 2022.
- [77] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila. Noise2Noise: Learning image restoration without clean data. In *International Conference on Machine Learning (ICML)*, volume 80, pages 2965–2974, 2018.
- [78] V. Lempitsky, A. Vedaldi, and D. Ulyanov. Deep image prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9446–9454, 2018.
- [79] A. Levin and B. Nadler. Natural image denoising: Optimality and inherent bounds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2833–2840, 2011.
- [80] A. Levin, B. Nadler, F. Durand, and W. T. Freeman. Patch complexity, finite pixel correlations and optimal denoising. In *European Conference on Computer Vision (ECCV)*, pages 73–86, 2012.
- [81] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte. SwinIR: Image restoration using Swin Transformer. In *International Conference on Computer Vision Workshops (ICCVW)*, pages 1833–1844, 2021.
- [82] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee. Enhanced deep residual networks for single image super-resolution. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1132–1140, 2017.
- [83] J. H. Lim, A. Courville, C. Pal, and C.-W. Huang. AR-DAE: Towards unbiased neural entropy gradient estimation. In *International Conference on Machine Learning (ICML)*, volume 119, pages 6061–6071, 2020.
- [84] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang. Non-local recurrent network for image restoration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.
- [85] H. Liu, X. Liu, J. Lu, and S. Tan. Self-supervised image prior learning with GMM from a single noisy image. In *International Conference on Computer Vision (ICCV)*, pages 2825–2834, 2021.
- [86] J. Liu, Y. Sun, X. Xu, and U. S. Kamilov. Image restoration using total variation regularized deep image prior. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7715–7719, 2019.
- [87] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo. Multi-level wavelet-CNN for image restoration. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 773–782, 2018.
- [88] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin Transformer: Hierarchical vision Transformer using shifted windows. In *International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021.
- [89] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A ConvNet for the 2020s. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022.
- [90] F. Luisier, T. Blu, B. Forster, and M. Unser. Which wavelet bases are the best for image denoising? In *Wavelets XI*, volume 5914, page 59140E, 2005.
- [91] F. Luisier, T. Blu, and M. Unser. A new SURE approach to image denoising: interscale orthonormal wavelet thresholding. *IEEE Transactions on Image Processing*, 16(3):593–606, 2007.
- [92] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang. Waterloo exploration database: New challenges for image quality assessment models. *IEEE Transactions on Image Processing*, 26(2):1004–1016, 2017.
- [93] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *International Conference on Computer Vision (ICCV)*, pages 2272–2279, 2009.
- [94] M. Makitalo and A. Foi. A closed-form approximation of the exact unbiased inverse of the anscombe variance-stabilizing transformation. *IEEE Transactions on Image Processing*, 20(9):2697–2698, 2011.
- [95] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.
- [96] J. V. Manjón, J. Carbonell-Caballero, J. J. Lull, G. García-Martí, L. Martí-Bonmatí, and M. Robles. MRI denoising using non-local means. *Medical image analysis*, 12(4):514–523, 2008.
- [97] X. Mao, C. Shen, and Y.-B. Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems (NIPS)*, volume 29, 2016.
- [98] D. Marcos, M. Volpi, N. Komodakis, and D. Tuia. Rotation equivariant vector field networks. In *International Conference on Computer Vision (ICCV)*, pages 5058–5067, 2017.

- [99] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 416–423 vol.2, 2001.
- [100] G. Mataev, P. Milanfar, and M. Elad. DeepRED: Deep image prior powered by RED. In *International Conference on Computer Vision Workshops (ICCVW)*, 2019.
- [101] P. Milanfar. Symmetrizing smoothing filters. *SIAM Journal on Imaging Sciences*, 6(1):263–284, 2013.
- [102] S. Mohan, Z. Kadkhodaie, E. P. Simoncelli, and C. Fernandez-Granda. Robust and interpretable blind image denoising via bias-free convolutional neural networks. In *International Conference on Learning Representations (ICLR)*, 2020.
- [103] N. Moran, D. Schmidt, Y. Zhong, and P. Coady. Noisier2Noise: Learning to denoise from unpaired noisy data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12061–12069, 2020.
- [104] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, 4(2):460–489, 2005.
- [105] T. Pang, H. Zheng, Y. Quan, and H. Ji. Recorrupted-to-Reccorrupted: Unsupervised deep learning for image denoising. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2043–2052, 2021.
- [106] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
- [107] T. Plötz and S. Roth. Neural nearest neighbors networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.
- [108] T. Plötz and S. Roth. Benchmarking denoising algorithms with real photographs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2750–2759, 2017.
- [109] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003.
- [110] M. Prakash, M. Lalit, P. Tomancak, A. Krul, and F. Jug. Fully unsupervised probabilistic Noise2Void. In *International Symposium on Biomedical Imaging (ISBI)*, pages 154–158, 2020.
- [111] Y. Quan, M. Chen, T. Pang, and H. Ji. Self2Self with dropout: Learning self-supervised denoising from single image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1887–1895, 2020.
- [112] S. Ramani, T. Blu, and M. Unser. Monte-Carlo SURE: A black-box optimization of regularization parameters for general denoising algorithms. *IEEE Transactions on Image Processing*, 17(9):1540–1554, 2008.
- [113] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.
- [114] Y. Romano, M. Elad, and P. Milanfar. The little engine that could: Regularization by Denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.
- [115] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 234–241, 2015.
- [116] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.
- [117] S. Roth and M. Black. Fields of Experts: a framework for learning image priors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 860–867, 2005.
- [118] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268, 1992.
- [119] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77:157–173, 2008.
- [120] A. Sagel, A. Roumy, and C. Guillemot. Sub-dip: Optimization on a subspace with deep image prior regularization and application to superresolution. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2513–2517, 2020.
- [121] J. Salmon. *Agrégation d'estimateurs et méthodes à patch pour le débruitage d'images numériques*. PhD thesis, Université Paris Diderot, 2010.

- [122] J. Salmon, R. Willett, and E. Arias-Castro. A two-stage denoising filter: The preprocessed Yaroslavsky filter. In *IEEE Statistical Signal Processing Workshop (SSP)*, pages 464–467, 2012.
- [123] V. G. Satorras, E. Hoogeboom, and M. Welling. $E(n)$ equivariant graph neural networks. In *International Conference on Machine Learning (ICML)*, volume 139, pages 9323–9332, 2021.
- [124] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016.
- [125] S. Soltanayev and S. Y. Chun. Training deep learning based denoisers without ground truth data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.
- [126] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.
- [127] J.-L. Starck and F. Murtagh. Automatic noise estimation from the multiresolution support. *Publications of the Astronomical Society of the Pacific*, 110(744):193, 1998.
- [128] J.-L. Starck, F. D. Murtagh, and A. Bijaoui. *Image Processing and Data Analysis: the multiscale approach*. Cambridge University Press, 1998.
- [129] C. M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6):1135–1151, 1981.
- [130] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds. *arXiv preprint arXiv:1802.08219*, 2018.
- [131] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *International Conference on Computer Vision (ICCV)*, pages 839–846, 1998.
- [132] A. Trockman and J. Z. Kolter. Patches are all you need? *arXiv preprint arXiv:2201.09792*, 2022.
- [133] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, volume 30, 2017.
- [134] B. S. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling. Rotation equivariant CNNs for digital pathology. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 210–218, 2018.
- [135] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-Play priors for model based reconstruction. In *IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013.
- [136] Y.-Q. Wang and J.-M. Morel. Can a single image denoising neural network handle all levels of Gaussian noise? *IEEE Signal Processing Letters*, 21(9):1150–1153, 2014.
- [137] M. Weigert, U. Schmidt, T. Boothe, A. Müller, A. Dibrov, A. Jain, B. Wilhelm, D. Schmidt, C. Broaddus, S. Culley, M. Rocha-Martins, F. Segovia-Miranda, C. Norden, R. Henriques, M. Zerial, M. Solimena, J. Rink, P. Tomancak, L. Royer, and E. Myers. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nature Methods*, 15(12):1090–1097, 2018.
- [138] M. Weiler, F. A. Hamprecht, and M. Storath. Learning steerable filters for rotation equivariant CNNs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 849–858, 2018.
- [139] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [140] G. Yu and G. Sapiro. DCT image denoising: a simple and effective image denoising algorithm. *Image Processing On Line*, 1:292–296, 2011.
- [141] G. Yu, G. Sapiro, and S. Mallat. Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity. *IEEE Transactions on Image Processing*, 21(5):2481–2499, 2012.
- [142] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M. Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5718–5729, 2022.
- [143] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, and L. Shao. CycleISP: Real image restoration via improved data synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2693–2702, 2020.
- [144] K. Zhang, Y. Li, J. Liang, J. Cao, Y. Zhang, H. Tang, D.-P. Fan, R. Timofte, and L. V. Gool. Practical blind image denoising via Swin-Conv-UNet and data synthesis. *Machine Intelligence Research*, 2023.

- [145] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte. Plug-and-Play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376, 2022.
- [146] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.
- [147] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep CNN denoiser prior for image restoration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2808–2817, 2017.
- [148] K. Zhang, W. Zuo, and L. Zhang. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.
- [149] Y. Zhang, K. Li, K. Li, B. Zhong, and Y. Fu. Residual non-local attention networks for image restoration. In *International Conference on Learning Representations (ICLR)*, 2019.
- [150] M. Zontak and M. Irani. Internal statistics of a single natural image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 977–984, 2011.
- [151] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *International Conference on Computer Vision (ICCV)*, pages 479–486, 2011.