

A Tour of Modern Image Filtering^{*}

Peyman Milanfar^{*}

Abstract

Recent developments in computational imaging and restoration have heralded the arrival and convergence of several powerful methods for adaptive processing of multidimensional data. Examples include Moving Least Square (from Graphics), the Bilateral Filter and Anisotropic Diffusion (from Machine Vision), Boosting, Kernel, and Spectral Methods (from Machine Learning), Non-local Means and its variants (from Signal Processing), Bregman Iterations (from Applied Math), Kernel Regression and Iterative Scaling (from Statistics). While these approaches found their inspirations in diverse fields of nascence, they are deeply connected. In this paper:

- I present a practical and accessible framework to understand some of the basic underpinnings of these methods, with the intention of leading the reader to a broad understanding of how they interrelate. I also illustrate connections between these techniques and more classical (and empirical) Bayesian approaches.
- The proposed framework is used to arrive at new insights and methods, both practical and theoretical. In particular, several novel optimality properties of algorithms in wide use such as BM3D, and methods for their iterative improvement (or non-existence thereof) are discussed.
- A general approach is laid out to enable the performance analysis and subsequent improvement of many existing filtering algorithms.

While much of the material discussed is applicable to the wider class of linear degradation models beyond noise (e.g. blur,) in order to keep matters focused, we consider the problem of denoising here.

^{*}Electrical Engineering Department, University of California, Santa Cruz CA, 95064 USA.

e-mail: milanfar@ee.ucsc.edu, Phone:(650) 322-2311, Fax: (410) 322-2312

^{*}.... with Some New Insights. An earlier version of this manuscript was titled “A Tour of Modern Image Processing”. To learn more, view my talk based on this paper, available at <http://tinyurl.com/moderntour>

I. INTRODUCTION

Multi-dimensional filtering is *the* fundamental operation in image and video processing, and low-level machine vision. In particular, the most widely used canonical filtering operation is one that removes or attenuates the effect of noise. As such, the basic design and analysis of image filtering operations form a very large part of the image processing literature; the resulting techniques often quickly spreading to the wider range of restoration and reconstruction problems in imaging. Over the years, many approaches have been tried, but only recently in the last decade or so, a great leap forward in performance has been realized. While largely unacknowledged in our community, this phenomenal progress has been mostly thanks to the adoption and development of non-parametric point estimation procedures adapted to the local structure of the given multi-dimensional data. Viewed through the lens of the denoising application, here we develop a general framework for understanding the basic science and engineering behind these techniques and their generalizations. Surely this is not the first paper to attempt such an ambitious overview, and it will likely not be the last; but the aim here is to provide a self-contained presentation which distills, generalizes, and puts into proper context many other excellent earlier works such as [1], [2], [3], [4], [5], and more recently [6]. It is fair to say that the present paper is, by necessity, not completely tutorial. Indeed it does contain several novel results; yet these are largely novel interpretations, formalizations, or generalizations of ideas already known or empirically familiar to the community. Hence, I hope that the enterprising reader will find this paper not only a good overview, but as should be the case with any useful presentation, a source of new insights and food for thought.

So to begin, let us consider the measurement model for the denoising problem:

$$y_i = z_i + e_i, \quad \text{for } i = 1, \dots, n, \quad (1)$$

where $z_i = z(x_i)$ is the underlying latent signal of interest at a position $x_i = [x_{i,1}, x_{i,2}]^T$, y_i is the noisy measured signal (pixel value), and e_i is zero-mean, white noise with variance σ^2 . We make no other distributional assumptions for the noise. The problem of interest then is to recover the complete set of samples of $z(x)$, which we denote vectorially as $\mathbf{z} = [z(x_1), z(x_2), \dots, z(x_n)]^T$ from the corresponding data set \mathbf{y} . To restate the problem more concisely, the complete measurement model in vector notation is given by ¹

$$\mathbf{y} = \mathbf{z} + \mathbf{e}. \quad (2)$$

¹Surely a similar analysis to what follows can and should be carried out for more general inverse problems such as deconvolution, interpolation, etc. We give a very brief hint of this direction in Appedix D.

It has been realized for some time now that effective restoration of signals will require methods which either model the signal a priori (i.e. are Bayesian) or learn the underlying characteristics of the signal from the given data (i.e. learning, non-parametric, or empirical Bayes methods.) Most recently, the latter category of approaches has become exceedingly popular. Perhaps the most striking recent example is the popularity of patch-based methods [7], [8], [9], [10]. This new generation of algorithms exploit both local and non-local redundancies or “self-similarities” in the signals being treated. Earlier on, the bilateral filter [8] was developed with very much the same idea in mind, as were its spiritually close predecessors: the Susan filter [11], normalized convolution [12], and the filters of Yaroslavsky [13]. The common philosophy among these and related techniques is the notion of measuring and making use of affinities between a given data point (or more generally patch or region) of interest, and others in the given measured signal \mathbf{y} . These similarities are then used in a filtering context to give higher weights to contributions from more similar data values, and to properly discount data points that are less similar. The pattern recognition literature has also been a source of parallel ideas. In particular, the celebrated mean-shift algorithm [14], [15] is in principle an iterated version of point-wise regression as also described in [2], [1]. In the machine learning community, the general regression problem has been carefully studied, and deep connections between regularization, least-squares regression, and the support vector formalism have also been established [16], [17], [18], [19].

Despite the voluminous recent literature on techniques based on these ideas, simply put, the key differences between the resulting practical filtering methods have been relatively minor, yet somewhat poorly understood. In particular, the underlying framework for each of these methods is distinct only to the extent that the weights assigned to different data points is decided upon differently. To be more concrete and mathematically precise, let us consider the denoising problem (2) again. The estimate of the signal $z(x)$ at the position x is found using a (non-parametric) *point estimation* framework; namely, the weighted least squares problem

$$\hat{z}(x_j) = \arg \min_{z(x_j)} \sum_{i=1}^n [y_i - z(x_j)]^2 K(x_i, x_j, y_i, y_j) \quad (3)$$

where the weight (or kernel) function $K(\cdot)$ is a symmetric², positive, and unimodal function which measures the “similarity” between the samples y_i and y_j , at respective positions x_i and x_j . If the kernel function is restricted to be only a function of the spatial locations x_i and x_j , then the resulting formulation is what is known as (classical, or not data-adaptive) *kernel regression* in the non-parametric statistics

²with respect to the indices i and j

literature [20], [21]. Perhaps more importantly, the key difference between *local* and *non-local* patch-based methods lies essentially in the definition of the range of the sum in (3). Namely, indices covering a small spatial region around a pixel of interest define local methods, and vice-versa.

Interestingly, in the early 1980's the essentially identical concept of *moving least-squares* emerged independently [22], [23] in the graphics community. This idea has since been widely adopted in computer graphics [24] as a very effective tool for smoothing and interpolation of data in 3-D. Surprisingly, despite the obvious connections between moving least-squares and the adaptive filters based on similarity, their kinship has remained largely hidden so far.

II. EXISTING ALGORITHMS

Over the years, the measure of similarity $K(x_i, x_j, y_i, y_j)$ has been defined in a number of different ways, leading to a cacophony of filters, including some of the most well-known recent approaches to image denoising [8], [9], [7]. Figure 1 gives a graphical illustration of how different choices of similarity kernels lead to different classes of filters, some of which we discuss next.

A. Classical Regression Filters [20], [21], [13]:

Naturally, the most naive way to measure the “distance” between two pixels is to simply consider their spatial Euclidean distance; namely, using a Gaussian kernel,

$$K(x_i, x_j, y_i, y_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{h_x^2}\right).$$

Such filters essentially lead to (possibly space-varying) Gaussian filters which are quite familiar from traditional image processing [25]. It is possible to adapt the variance (or bandwidth) parameter h_x to the local image statistics, and obtain a relatively modest improvement in performance. But the lack of stronger adaptivity to the underlying structure of the signal of interest is a major drawback of these classical approaches.

B. The Bilateral Filter (BL) [8], [3]:

Another manifestation of the formulation in (3) is the bilateral filter where the spatial *and* photometric distances between two pixels are taken into account in separable fashion as follows:

$$K(x_i, x_j, y_i, y_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{h_x^2}\right) \exp\left(\frac{-(y_i - y_j)^2}{h_y^2}\right) = \exp\left\{\frac{-\|x_i - x_j\|^2}{h_x^2} + \frac{-(y_i - y_j)^2}{h_y^2}\right\} \quad (4)$$

As can be observed in the exponent on the right-hand side, and in Figure 1, the similarity metric here is a weighted Euclidean distance between the vectors (x_i, y_i) and (x_j, y_j) . This approach has several advantages. Namely, while the kernel is easy to construct, and computationally simple to calculate, it yields useful local adaptivity to the given data. In addition, it has only two control parameters (h_x, h_y) , which make it very convenient to use. However, as is well-known, this filter does not provide effective performance in low signal-to-noise scenarios [3].

C. Non-Local Means (NLM) [7], [26], [27]:

The non-local means algorithm, originally proposed in [28] and [29], has stirred a great deal of interest in the community in recent years. At its core, however, it is a relatively simple generalization of the bilateral filter; namely, the photometric term in the bilateral similarity kernel, which is measured point-wise, is simply replaced with one that is patch-wise. A second difference is that (at least in theory) the geometric distance between the patches (corresponding to the first term in the bilateral similarity kernel), is essentially ignored, leading to strong contribution from patches that may not be physically near the pixel of interest (hence the name *non-local*). To summarize, the NLM kernel is

$$K(x_i, x_j, y_i, y_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{h_x^2}\right) \exp\left(\frac{-\|\mathbf{y}_i - \mathbf{y}_j\|^2}{h_y^2}\right) \quad \text{with} \quad h_x \longrightarrow \infty, \quad (5)$$

where \mathbf{y}_i and \mathbf{y}_j refer now to *patches* of pixels centered at pixels y_i and y_j , respectively. In practice, two implementation details should be observed. First, the patch-wise photometric distance $\|\mathbf{y}_i - \mathbf{y}_j\|^2$ in the above is in fact measured as $(\mathbf{y}_i - \mathbf{y}_j)^T \mathbf{G} (\mathbf{y}_i - \mathbf{y}_j)$ where \mathbf{G} is a fixed diagonal matrix containing Gaussian weights which give higher importance to the center of the respective patches. Second, it is computationally rather impractical to compare *all* the patches \mathbf{y}_i to \mathbf{y}_j , so although the non-local means approach in Buades et al. [28] theoretically forces h_x to be infinite, in practice typically the search is limited to a reasonable spatial neighborhood of y_j . Consequently, in effect the NLM filter too is more or less local; or said another way, h_x is never infinite in practice. The method in Awate et al. [29], on the other hand, proposes a *Gaussian-distributed* sample which comes closer to the exponential weighting on Euclidean distances in (5).

Despite its popularity, the performance of the NLM filter leaves much to be desired. The true potential of this filtering scheme was demonstrated only later with the Optimal Spatial Adaptation (OSA) approach of Boulanger and Kervrann [26]. In their approach, the photometric distance was refined to include estimates of the local noise variances within each patch. Namely, they computed a local diagonal covariance matrix, and defined the locally adaptive photometric distance as $(\mathbf{y}_i - \mathbf{y}_j)^T \mathbf{V}_j^{-1} (\mathbf{y}_i - \mathbf{y}_j)$ in such a way as to

minimize an estimate of the local mean-squared error. Furthermore, they considered iterative application of the filter as discussed in Section V.

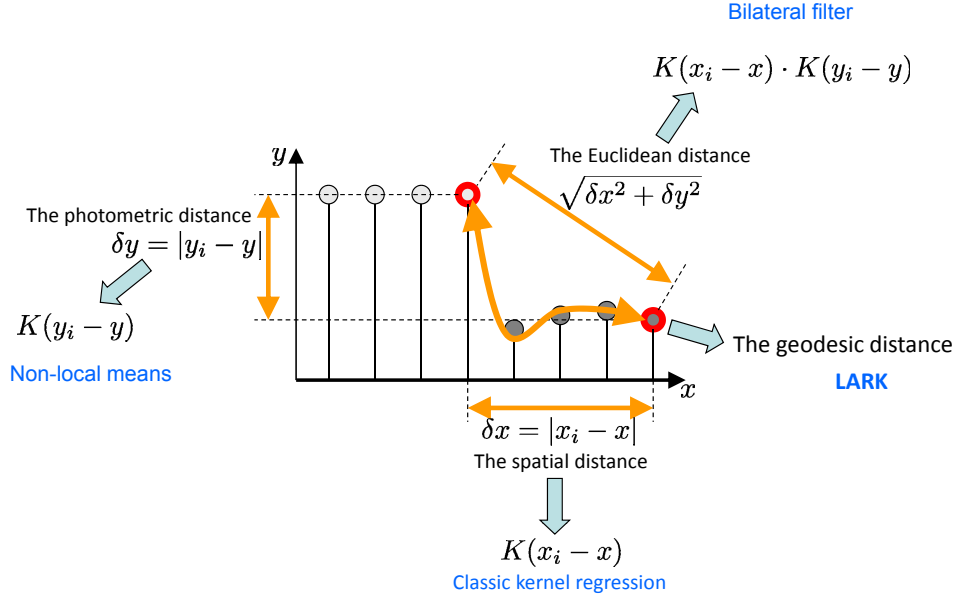


Fig. 1. Similarity Metrics and the Resulting Filters

D. Locally Adaptive Regression (Steering) Kernels (LARK) [9]:

The key idea behind this measure of similarity, originally proposed in [9] is to robustly obtain the local structure of images by analyzing the photometric (pixel value) differences based on estimated gradients, and to use this structure information to adapt the shape and size of a canonical kernel. The LARK kernel is defined as follows:

$$K(x_i, x_j, y_i, y_j) = \exp \left\{ -(x_i - x_j)^T \mathbf{C}_i (x_i - x_j) \right\}, \quad (6)$$

where the matrix $\mathbf{C}_i = \mathbf{C}(y_i, y_j)$ is estimated from the given data as

$$\mathbf{C}_i = \sum_j \begin{bmatrix} z_{x_{i,1}}^2(x_j) & z_{x_{i,1}}(x_j)z_{x_{i,2}}(x_j) \\ z_{x_{i,1}}(x_j)z_{x_{i,2}}(x_j) & z_{x_{i,2}}^2(x_j) \end{bmatrix}.$$

Specifically, $z_{x_{i,*}}(x_j)$ are the estimated gradients of the underlying signal at point x_i , computed from the given measurements y_j in a patch around the point of interest. In particular, the gradients used in

the above expression are estimated from the given noisy image by applying classical (i.e. non-adaptive) locally linear kernel regression. Details of this estimation procedure are given in [30]. The reader may recognize the above matrix as the well-studied “structure tensor” [31]. The advantage of the LARK descriptor is that it is exceedingly robust to noise and perturbations of the data. The formulation is also theoretically well-motivated since the quadratic exponent in (6) essentially encodes the local *geodesic* distance between the points (x_i, y_i) and (x_j, y_j) on the graph of the function $z(x, y)$ thought of as a 2-D surface (a manifold) embedded in 3-D. The geodesic distance was also used in the context of the Beltrami-flow kernel in [32], [33] in an analogous fashion.

III. GENERALIZATIONS AND PROPERTIES

The above discussion can be naturally generalized by defining the augmented data variable $\mathbf{t}_i = [x_i^T, \mathbf{y}_i^T]^T$ and a general Gaussian kernel as follows:

$$K(\mathbf{t}_i, \mathbf{t}_j) = \exp \left\{ -(\mathbf{t}_i - \mathbf{t}_j)^T \mathbf{Q} (\mathbf{t}_i - \mathbf{t}_j) \right\}, \quad (7)$$

$$\mathbf{Q}_{i,j} = \begin{bmatrix} \mathbf{Q}_x & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_y \end{bmatrix} \quad (8)$$

where \mathbf{Q} is symmetric positive definite (SPD).

Setting $\mathbf{Q}_x = \frac{1}{h_x^2} \mathbf{I}$ and $\mathbf{Q}_y = \mathbf{0}$ we have classical kernel regression, whereas one obtains the bilateral filter framework when $\mathbf{Q}_x = \frac{1}{h_x^2} \mathbf{I}$ and $\mathbf{Q}_y = \frac{1}{h_y^2} \text{diag}[0, 0, \dots, 1, \dots, 0, 0]$. The latter diagonal matrix picks out the center pixel in the element-wise difference of patches $\mathbf{t}_i - \mathbf{t}_j$. When $\mathbf{Q}_x = \mathbf{0}$ and $\mathbf{Q}_y = \frac{1}{h_y^2} \mathbf{G}$, we have the nonlocal means filter and its variants. Finally, the LARK kernel in (6) is obtained when $\mathbf{Q}_x = \mathbf{C}_i$ and $\mathbf{Q}_y = \mathbf{0}$. More generally, the matrix \mathbf{Q} can be selected so that it has non-zero off-diagonal blocks. However, no practical algorithms with this choice have been proposed so far. As detailed below, with a symmetric positive definite \mathbf{Q} , this general approach results in valid symmetric positive definite kernels, a property that is used throughout the rest of our discussion³.

The above concepts can be further extended using the powerful theory of reproducible kernels originated in functional analysis (and later successfully adopted in machine learning [34], [35]) in order to present a significantly more general framework for selecting the similarity functions. This will help us identify the wider class of admissible similarity kernels more formally, and to understand how to produce new

³The definition of \mathbf{t} given here is only one of many possible choices. Our treatment in this paper is equally valid when, for instance, $\mathbf{t} = \mathbf{T}(x, \mathbf{y})$ is any feature derived from a convenient linear or nonlinear transformation of the original data.

kernels from ones already defined [35]. Formally, a scalar-valued function $K(\mathbf{t}, \mathbf{s})$ over a compact region of its domain \mathbb{R}^n , is called an admissible kernel if

- K is symmetric: $K(\mathbf{t}, \mathbf{s}) = K(\mathbf{s}, \mathbf{t})$
- K is positive definite. That is, for any collection of points $\mathbf{t}_i, i = 1, \dots, n$, the *Gram* matrix with elements $\mathbf{K}_{i,j} = K(\mathbf{t}_i, \mathbf{t}_j)$ is positive definite.

Such kernels satisfy some useful properties such as positivity, $K(\mathbf{t}, \mathbf{t}) \geq 0$, and the Cauchy-Schwartz inequality $K(\mathbf{t}, \mathbf{s}) \leq \sqrt{K(\mathbf{t}, \mathbf{t})K(\mathbf{s}, \mathbf{s})}$.

With the above definitions in place, there are numerous ways to construct new valid kernels from existing ones. We list some of the most useful ones below, without proof [35]. Given two valid kernels $K_1(\mathbf{t}, \mathbf{s})$ and $K_2(\mathbf{t}, \mathbf{s})$, the following constructions yield admissible kernels:

- 1) $K(\mathbf{t}, \mathbf{s}) = \alpha K_1(\mathbf{t}, \mathbf{s}) + \beta K_2(\mathbf{t}, \mathbf{s})$ for any pair $\alpha, \beta \geq 0$
- 2) $K(\mathbf{t}, \mathbf{s}) = K_1(\mathbf{t}, \mathbf{s}) K_2(\mathbf{t}, \mathbf{s})$
- 3) $K(\mathbf{t}, \mathbf{s}) = k(\mathbf{t}) k(\mathbf{s})$, where $k(\cdot)$ is a scalar-valued function.
- 4) $K(\mathbf{t}, \mathbf{s}) = p(K_1(\mathbf{t}, \mathbf{s}))$, where $p(\cdot)$ is a polynomial with positive coefficients.
- 5) $K(\mathbf{t}, \mathbf{s}) = \exp(K_1(\mathbf{t}, \mathbf{s}))$

Regardless of the choice of the kernel function, the weighted least-square optimization problem (3) has a simple solution. In matrix notation we can write

$$\hat{z}(x_j) = \arg \min_{z(x_j)} [\mathbf{y} - z(x_j)\mathbf{1}_n]^T \mathbf{K}_j [\mathbf{y} - z(x_j)\mathbf{1}_n] \quad (9)$$

where $\mathbf{1}_n = [1, \dots, 1]^T$, and $\mathbf{K}_j = \text{diag} [K(x_1, x_j, y_1, y_j), K(x_2, x_j, y_2, y_j), \dots, K(x_n, x_j, y_n, y_j)]$.

The closed form solution to the above is

$$\begin{aligned} \hat{z}(x_j) &= (\mathbf{1}_n^T \mathbf{K}_j \mathbf{1}_n)^{-1} \mathbf{1}_n^T \mathbf{K}_j \mathbf{y} & (10) \\ &= \left(\sum_i K(x_i, x_j, y_i, y_j) \right)^{-1} \left(\sum_i K(x_i, x_j, y_i, y_j) y_i \right) \\ &= \sum_i \frac{K_{ij}}{\sum_i K_{ij}} y_i \\ &= \sum_i W_{ij} y_i \\ &= \mathbf{w}_j^T \mathbf{y}. & (11) \end{aligned}$$

So in general, the estimate $\hat{z}(x_j)$ of the signal at position x_j is given by a weighted sum of *all* the given data points $y(x_i)$, each contributing a weight commensurate with its similarity as indicated by $K(\cdot)$, with the measurement $y(x_j)$ at the position of interest. Furthermore, as should be apparent in (10), the

weights sum to one. To control computational complexity, or to design local versus non-local filters, we may choose to set the weight for some “sufficiently far-away” pixels to zero or a small value, leading to a weighted sum involving a relatively small number of data points in a properly defined vicinity of the sample of interest. This is essentially the only distinction between locally adaptive processing methods (such as BL, and LARK) and so-called *non-local* methods such as NLM. It is worth noting that in the formulation above, despite the simple form of (10), in general we have a nonlinear estimator since the weights $W_{ij} = W(x_i, x_j, y_i, y_j)$ depend on the noisy data⁴.

To summarize the discussion so far, we have presented a general framework which absorbs many existing algorithms as special cases. This was done in several ways, including a general description of the set of admissible similarity kernels, which allows the construction of a wide variety of new kernels not considered before in the image processing literature. Next, we turn our attention to the matrix formulation of the non-parametric filtering approach. As we shall see, this provides a framework for more in-depth and intuitive understanding of the resulting filters, their subsequent improvement, and for their respective asymptotic and numerical properties.

Before we end this section, it is worth saying a few words about computational complexity. In general, patch-based methods are quite computationally intensive. Recently, many works have aimed at both efficiently searching for similar patches, and more cleverly computing the resulting weights. Among these, notable recent work appears in [36] and [37].

IV. THE MATRIX FORMULATION AND ITS PROPERTIES

In this section, we analyze the filtering problems posed earlier in the language of linear algebra and make several theoretical and practical observations. In particular, we are able to not only study the numerical/algebraic properties of the resulting filters, but also to analyze some of their fundamental statistical properties.

To begin, recall the convenient vector form of the filters:

$$\hat{z}(x_j) = \mathbf{w}_j^T \mathbf{y}. \quad (12)$$

where $\mathbf{w}_j^T = [W(x_1, x_j, y_1, y_j), W(x_2, x_j, y_2, y_j), \dots, W(x_n, x_j, y_n, y_j)]$ is a vector of weights for

⁴The non-parametric approach in (3) can be further extended to include a more general expansion of the signal $z(x)$ in a desired basis. We briefly discuss this case in the appendix, but leave its full treatment for future research.

each j . Writing the above at once for all j we have

$$\hat{\mathbf{z}} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_n^T \end{bmatrix} \mathbf{y} = \mathbf{W} \mathbf{y}. \quad (13)$$

As such, the filters defined by the above process can be analyzed as the product of a (square, $n \times n$) matrix of weights \mathbf{W} with the vector of the given data \mathbf{y} . First, a notational matter: \mathbf{W} is in general a function of the data, so strictly speaking, the notation $\mathbf{W}(\mathbf{y})$ would be more descriptive. But as we will describe later in more detail, the typical process for computing these weights in practice involves first computing a ‘‘pilot’’ or preliminary denoised version of the image, from which the weights are then calculated. This pre-processing, done only for the purposes of computing the parameters of the filter, reduces the effective variance of the input noise and stabilizes the calculation of the weights so that they are not a strong function of the input noise level. As such, it is practically more useful to think of the weights as depending more directly on the underlying (unknown) image $\mathbf{W}(\mathbf{z})$. As such, we shall simply denote the weight matrix going forward as \mathbf{W} . A more rigorous justification for this is provided later in the next Section, and supported in Appendix C.

Next, we highlight some important properties of the matrix \mathbf{W} , which lend much insight to our analysis. Referring to (10), \mathbf{W} can be written as the product

$$\mathbf{W} = \mathbf{D}^{-1} \mathbf{K}, \quad (14)$$

where $[\mathbf{K}]_{ij} = K(x_i, x_j, y_i, y_j) = K_{ij}$ and \mathbf{D} is a positive definite diagonal matrix with the normalizing factors $\mathbf{D}_{jj} = \text{diag}\{\sum_i K_{ij}\}$ along its diagonals. We can write

$$\mathbf{W} = \mathbf{D}^{-1} \mathbf{K} = \mathbf{D}^{-1/2} \underbrace{\mathbf{D}^{-1/2} \mathbf{K} \mathbf{D}^{-1/2}}_{\mathbf{L}} \mathbf{D}^{1/2} \quad (15)$$

It is evident that since \mathbf{K} and \mathbf{D} are symmetric positive definite, then so is \mathbf{L} . Meanwhile, \mathbf{W} and \mathbf{L} are related through a diagonal similarity transformation, and therefore have the same eigenvalues. Hence, interestingly, \mathbf{W} has real, non-negative eigenvalues (even though it is not symmetric!)⁵ An alternative, though not necessarily more intuitive, description of the action of \mathbf{W} is possible in terms of graphical models. Namely, if we consider the data (x_i, y_i) to be nodes on a (finite, undirected, weighted) graph

⁵That is, \mathbf{W} is positive definite but *not* symmetric.

with weights between nodes i and j given by the kernel values $K(x_i, x_j, y_i, y_j)$, the matrix $\mathcal{L} = \mathbf{L} - \mathbf{I}$ is precisely the “graph Laplacian” [38], [5], [39]⁶. This is illustrated in Fig. 2.

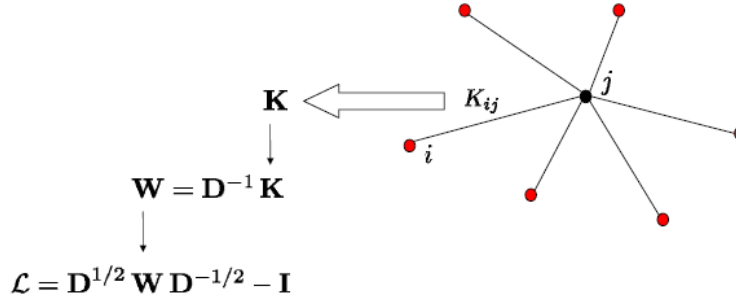


Fig. 2. Construction of the graph Laplacian \mathcal{L} from the kernel weights \mathbf{K} .

Next, we note that \mathbf{W} is a positive row-stochastic matrix⁷ – namely, by definition, the sum of each of its rows is equal to one, as should be clear from (10). Hence, the classical Perron-Frobenius theory is at our disposal [40], [41], which completely characterizes the spectra of such matrices. In particular [42], [43], we have that \mathbf{W} has spectral radius $0 \leq \lambda(\mathbf{W}) \leq 1$. Indeed, the largest eigenvalue of \mathbf{W} is simple (unrepeated) and is exactly $\lambda_1 = 1$, with corresponding eigenvector $\mathbf{v}_1 = (1/\sqrt{n})[1, 1, \dots, 1]^T = (1/\sqrt{n})\mathbf{1}_n$. Intuitively, this means that filtering by \mathbf{W} preserves the average grey level, and will leave a constant signal (i.e., a “flat” image) unchanged. In fact, with its spectrum inside the unit disk, \mathbf{W} is an *ergodic* matrix [40], [41], which is to say that its powers converge to a matrix of rank one, with identical rows. More explicitly, using the eigen-decomposition of \mathbf{W} we can write

$$\mathbf{W}^k = \mathbf{V} \mathbf{S}^k \mathbf{V}^{-1} = \mathbf{V} \mathbf{S}^k \mathbf{U}^T = \sum_{i=1}^n \lambda_i^k \mathbf{v}_i \mathbf{u}_i^T,$$

where we have defined $\mathbf{U}^T = \mathbf{V}^{-1}$, while denoting by \mathbf{u}_i^T the left eigenvectors of \mathbf{W} which are the columns of \mathbf{U} . Therefore,

$$\lim_{k \rightarrow \infty} \mathbf{W}^k = \mathbf{1}_n \mathbf{u}_1^T.$$

⁶The name is not surprising. In the canonical case, \mathbf{W} is a low-pass filter (though possibly nonlinear and space-varying.) Since \mathbf{W} and \mathbf{L} share the same spectrum, \mathbf{L} is also a low-pass filter. Therefore, $\mathcal{L} = \mathbf{L} - \mathbf{I}$ is generically a high-pass filter, justifying the *Laplacian* label from the point of view of filtering.

⁷Furthermore, the structure of the pixel data as samples of a function on a finite, fixed, lattice means that \mathbf{W} can also be interpreted [5], [39] as the transition matrix of an aperiodic and irreducible Markov chain defined on the entries of the data vector \mathbf{y} .

So \mathbf{u}_1 summarizes the asymptotic effect of applying the filter \mathbf{W} many times.

As we made clear earlier, $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$ is generically not a symmetric matrix, though it always has real, positive eigenvalues. As such, it should not come as a surprise that \mathbf{W} is quite close to a symmetric positive definite matrix. As we illustrate in the Appendix, this is indeed the case. As such, we shall find it useful to approximate \mathbf{W} with such a symmetric positive definite matrix. To effect this approximation requires a bit of care, as the resulting matrix must still be row-stochastic. Fortunately, this is not difficult to do. Sinkhorn’s algorithm [44], [45] for matrix scaling provides a provably optimal [46] way to accomplish this task⁸. The obtained matrix is a close approximation of a given \mathbf{W} which is both symmetric positive definite, and now doubly (i.e., row- *and* column-) stochastic. Working with a symmetric (or rather “symmetrized”) \mathbf{W} makes possible much of the analysis that follows in the remainder of this paper⁹.

From this point forward therefore, we shall consider \mathbf{W} to be a symmetric positive definite, and (doubly-) stochastic matrix. The spectrum of \mathbf{W} determines the effect of the filter on the noisy signal \mathbf{y} . We write the eigen-decomposition:

$$\mathbf{W} = \mathbf{V}\mathbf{S}\mathbf{V}^T \tag{16}$$

where $\mathbf{S} = \text{diag}[\lambda_1, \dots, \lambda_n]$ contains the eigenvalues in decreasing order $0 \leq \lambda_n \leq \dots \leq \lambda_1 = 1$, and \mathbf{V} is an orthogonal matrix. As an example, we illustrate the spectrum of the LARK [9] kernel in Figure 3 for different types of patches. As can be observed, the spectrum of \mathbf{W} decays rather quickly for “flat” patches, indicating that the filter is very aggressive in denoising these types of patches. As we consider increasingly more complex anisotropic patches containing edges, corners, and various textures, the spectrum of the filter is (automatically) adjusted to be less aggressive in particular directions. To take the specific case of a patch containing a simple edge, the filter is adjusted in unsupervised fashion to perform strong denoising *along* the edge, and to preserve and enhance the structure of the patch *across* the edge. This type of behavior is typical of the adaptive non-parametric filters we have discussed so far, but it is particularly striking and stable in the case of LARK kernels shown in Figure 3.

Of course, the matrix \mathbf{W} is a function of the noisy data \mathbf{y} and hence it is strictly speaking a stochastic variable. But another interesting property of \mathbf{W} is that when the similarity kernel $K(\cdot)$ is of the general

⁸We refer the reader to the appendix and citations therein, where we illustrate and justify both the approximation procedure, and its fidelity in detail.

⁹Interestingly, there is an inherent and practical advantage in using symmetrization. As we will see in the experimental results, given any \mathbf{W} , employing its symmetrized version generally *improves* performance in the mean-squared error sense. We do not present a proof of this observation here, and leave its analysis for another occasion.

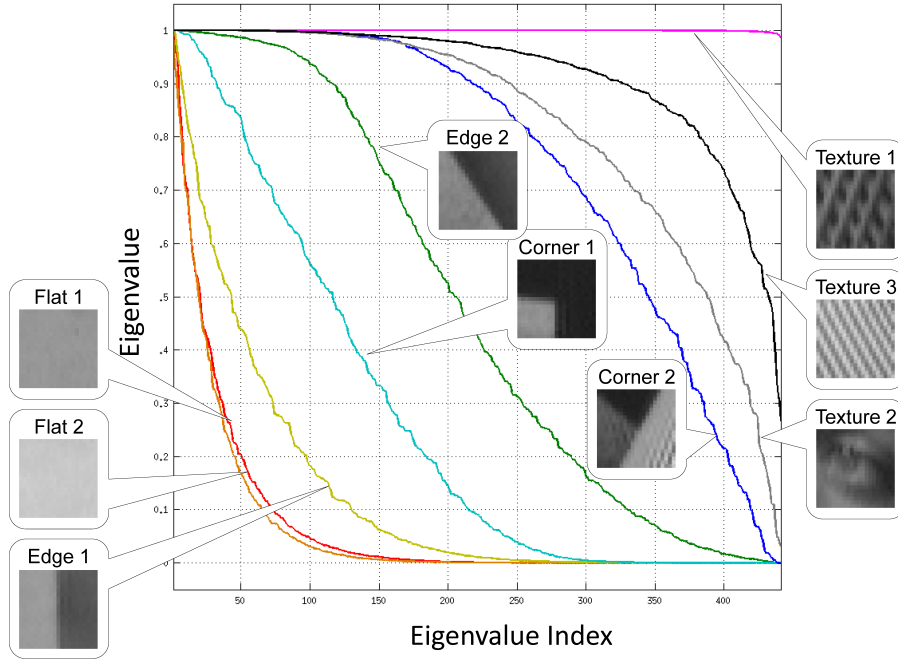


Fig. 3. Spectrum of the LARK filter on different types of patches.

Gaussian type as in (7), as described in the Appendix, the resulting filter coefficients \mathbf{W} are quite stable to perturbations of the data by noise of modest size [47]. From a practical point of view, and insofar as the computation of the matrix \mathbf{W} is concerned, it is always reasonable to assume that the noise variance is relatively small, because in practice we typically compute \mathbf{W} on a “pre-processed” version of the noisy image \mathbf{y} anyway¹⁰. Going forward, therefore, we consider \mathbf{W} as fixed. Some experiments in Section V confirm the validity of this assumption.

A. Statistical Analysis of the Filters’ Performance

We are now in a position to carry out an analysis of the performance of filters defined earlier. As we just explained, to carry out this analysis, we first note that \mathbf{W} is, strictly speaking, a function of the noisy data \mathbf{y} . Indeed, the dependence on the given data is what gives strength to these adaptive filters. On the other hand, one may legitimately worry that the effect of noise on the computation of these weights may be dramatic, resulting in too much sensitivity for the resulting filters to be effective. This issue has

¹⁰This small noise assumption is made only for the analysis of the filter coefficients, and is not invoked in rest of the paper.

indeed been considered in the context of most patch-based adaptive filters. The consensus approach, in keeping with similar thinking in the non-parametric statistics literature, is to compute the weights from a “pilot” (or pre-filtered) version $\hat{\mathbf{z}}_0$ of the noisy data; that is $\mathbf{W}(\hat{\mathbf{z}}_0)$. The pre-processing step yielding the pilot estimate $\hat{\mathbf{z}}_0$ can be carried out with any one of many simpler (non-adaptive) filters, and is done only for the purpose of stably computing the parameters of the filter. The net effect of this step, however, is significant: it reduces the effective variance of the input noise and stabilizes the calculation of the weights so that they are *not* a strong function of the input noise level any longer. In this setting, the noise affecting the calculation of the weights can, for all practical intents, be assumed small. As such, the stability results in [47] (summarized in Appendix C) give strong guarantees that the weights are consequently disturbed relatively little. That is, $\mathbf{W}(\hat{\mathbf{z}}_0) \approx \mathbf{W}(\mathbf{z})$. For the purposes of performance analysis below, it is therefore reasonable (and practically verified in Figures 9, 10, 15, and 16) to consider the weights as depending more directly on the underlying (unknown) image $\mathbf{W}(\mathbf{z})$, rather than $\mathbf{W}(\mathbf{y})$. Therefore, in what follows we will consider the weight matrix \mathbf{W} to be, at least approximately, non-random¹¹.

Now, let us compute an approximation to the bias and variance of the estimator $\hat{\mathbf{z}} = \mathbf{W}\mathbf{y}$. The bias in the estimate is

$$\text{bias} = \mathbf{E}(\hat{\mathbf{z}}) - \mathbf{z} = \mathbf{E}(\mathbf{W}\mathbf{y}) - \mathbf{z} \approx \mathbf{W}\mathbf{z} - \mathbf{z} = (\mathbf{W} - \mathbf{I})\mathbf{z}.$$

Recalling that the matrix \mathbf{W} has a unit eigenvalue corresponding to a (constant) vector, we note that $\hat{\mathbf{z}}$ is an unbiased estimate if \mathbf{z} is a constant image, but is biased for all other underlying images. The squared magnitude of the bias is given by

$$\|\text{bias}\|^2 = \|(\mathbf{W} - \mathbf{I})\mathbf{z}\|^2. \quad (17)$$

Writing the image \mathbf{z} in the column space of the orthogonal matrix \mathbf{V} (which contains the eigenvectors as its columns) as $\mathbf{z} = \mathbf{V}\mathbf{b}_0$, we can rewrite the squared bias magnitude as

$$\|\text{bias}\|^2 = \|(\mathbf{W} - \mathbf{I})\mathbf{z}\|^2 = \|\mathbf{V}(\mathbf{S} - \mathbf{I})\mathbf{b}_0\|^2 = \|(\mathbf{S} - \mathbf{I})\mathbf{b}_0\|^2 = \sum_{i=1}^n (\lambda_i - 1)^2 b_{0i}^2. \quad (18)$$

We note that the last sum can be expressed over the indices $i = 2, \dots, n$, because the first term in fact vanishes since $\lambda_1 = 1$. Next, we consider the variance of the estimate. We have

$$\text{cov}(\hat{\mathbf{z}}) = \text{cov}(\mathbf{W}\mathbf{y}) \approx \text{cov}(\mathbf{W}\mathbf{e}) = \sigma^2 \mathbf{W}\mathbf{W}^T \quad \implies \quad \text{var}(\hat{\mathbf{z}}) = \text{tr}(\text{cov}(\hat{\mathbf{z}})) = \sigma^2 \sum_{i=1}^n \lambda_i^2.$$

¹¹As such, we also drop the dependence on its argument for the sake of simplifying the notation. But the reader is advised to always remember that the weights are adapted to the (estimated) structure of the underlying image.

Overall, the mean-squared-error (MSE) is given by

$$\mathbf{MSE} = \|\text{bias}\|^2 + \text{var}(\hat{\mathbf{z}}) = \sum_{i=1}^n (\lambda_i - 1)^2 b_{0i}^2 + \sigma^2 \lambda_i^2. \quad (19)$$

a) *The ideal spectrum: BM3D [48] Explained:* Let us pause for a moment and imagine that we can optimally *design* the matrix \mathbf{W} from scratch in such a way as to minimize the mean-squared error above. That is, consider the set of eigenvalues for which (19) is minimized. Differentiating the expression for MSE with respect to λ_i and setting equal to zero leads to a familiar expression:

$$\lambda_i^* = \frac{b_{0i}^2}{b_{0i}^2 + \sigma^2} = \frac{1}{1 + \mathbf{snr}_i^{-1}}. \quad (20)$$

This is of course nothing but a manifestation of the Wiener filter with $\mathbf{snr}_i = \frac{b_{0i}^2}{\sigma^2}$ denoting the signal-to-noise ratio at each component i of the signal. This observation naturally raises the question of whether any existing filters in fact get close to this optimal performance [49]. Two filters that are designed to approximately achieve this goal are BM3D [48], and PLOW [50] which are at present considered to be the state of the art in denoising. The BM3D algorithm can be briefly summarized by the following three steps:

- 1) Patches from an input image are classified according to their similarity, and are grouped into 3-D clusters accordingly.
- 2) A so-called “3D collaborative Wiener filtering” is implemented to process each cluster.
- 3) Filtered patches inside all the clusters are aggregated to form an output.

The core process of the BM3D algorithm is the collaborative Wiener filtering that transforms a patch cluster through a fixed orthonormal basis \mathbf{V} (e.g. DCT) [48], where the coefficients of each component are then scaled by some “shrinkage” factor $\{\lambda_i\}$, which are precisely chosen according to (20). Following this shrinkage step, the data are transformed back to the spatial domain to generate denoised patches for the aggregation process. In practice, of course, one does not have access to $\{b_{0i}^2\}$, so they are roughly estimated from the given noisy image \mathbf{y} , as is done also in [48]. The BM3D is not based on the design of a specific kernel $K(\cdot)$, or even on a regression framework, as are other patch-based methods (such as NLM, LARK). In fact, there need not exist *any* closed form kernel which gives rise to the BM3D filter. Still, with the present framework, the diagonal Wiener shrinkage matrix $\mathbf{S} = \text{diag}[\lambda_1, \dots, \lambda_n]$, and the overall 3-D collaborative Wiener filtering process can now be understood using a symmetric, positive-definite filter matrix \mathbf{W} with orthonormal eigen-decomposition $\mathbf{V}\mathbf{S}\mathbf{V}^T$. If the orthonormal basis \mathbf{V} contains a constant vector $\mathbf{v}_{i'} = \frac{1}{\sqrt{n}}\mathbf{1}_n$, one can easily make \mathbf{W} doubly-stochastic by setting its corresponding shrinkage factor $\lambda_{i'} = 1$.

To summarize this section, we have presented a widely applicable matrix formulation $\hat{\mathbf{z}} = \mathbf{W} \mathbf{y}$ for the denoising problem, where \mathbf{W} is henceforth considered *symmetric, positive-definite, and doubly-stochastic*. This formulation has thus far allowed us to characterize the performance of the resulting filters in terms of their spectra, and to obtain insights as to the relative statistical efficiency of existing filters such as BM3D. One final observation worth making here is that the Birkhoff – von Neumann Theorem [41] notes that \mathbf{W} is doubly stochastic if and only if it is a convex combination of permutation matrices. Namely, $\mathbf{W} = \sum_{l=1}^{\bar{n}} \alpha_l \mathbf{P}_l$, where \mathbf{P}_l are permutation matrices; α_l are non-negative scalars with $\sum \alpha_l = 1$, and \bar{n} is at most $(n - 1)^2 + 1$. This means that regardless of the choice of (an admissible) kernel, each pixel of the estimated image

$$\hat{\mathbf{z}} = \mathbf{W} \mathbf{y} = \sum_{l=1}^{\bar{n}} \alpha_l (\mathbf{P}_l \mathbf{y})$$

is always a convex combination of at most $\bar{n} = (n - 1)^2 + 1$ (not necessarily nearby) pixels of the noisy input.

Going forward, we first use the insights gained thus far to study further improvements of the non-parametric approach. Then, we describe the relationship between the non-parametric approaches and more familiar parametric Bayesian methods.

V. IMPROVING THE ESTIMATE BY ITERATION

As we described above, the *optimal* spectrum for the filter matrix \mathbf{W} is dictated by the Wiener condition (20), which requires clairvoyant knowledge of the signal or at least careful estimation of the SNR at each spatial frequency. In practice, however, nearly all similarity-based methods we described so far (with the notable exception of the BM3D) design a matrix \mathbf{W} based on a choice of the kernel function, and without regard to the ultimate consequences of this choice on the spectrum of the resulting filter. Hence, at least in the mean-squared error sense, such choices of \mathbf{W} are always deficient¹². So one might naturally wonder if such estimates can be improved through some post-facto iterative mechanism. The answer is an emphatic yes, and that is the subject of this section.

To put the problem in more practical terms, the estimate derived from the application of a non-parametric denoising filter \mathbf{W} will not only remove some noise, but invariably some of the underlying signal as well. A number of different approaches have been proposed to reintroduce this “lost” component back to the estimate. Interestingly, as we describe later in Section VI, similar to the steepest descent

¹²As the former US Secretary of Defense D. Rumsfeld might have said it, we filter with the spectrum that we have, not the spectrum that we would *like* to have!

methods employed in the solution of more classical Bayesian optimization approaches, the iterative method employed here involve multiple applications of a filter (or sequence of filters) on the data, and somehow pooling the results. Next we discuss the two most well-known and frequently used approaches. The first is *anisotropic diffusion* [51], whereas the second class includes the recently popularized *Bregman iterations* [52], which is closely related to ℓ_2 -boosting [53]. The latter is a generalization of *twicing* introduced by Tukey [54] more than thirty years ago. In fact, as we will illustrate later, this latter class of iterations is also related to *biased anisotropic diffusion* [55], otherwise known as anisotropic *reaction-diffusion*.

A. Diffusion

So far, we have described a general formulation for denoising as a spatially adaptive, data-dependent filtering procedure (3) amounting to

$$\hat{\mathbf{z}} = \mathbf{W} \mathbf{y}.$$

Now consider applying the filter multiple times. That is, define $\hat{\mathbf{z}}_0 = \mathbf{y}$, and the iteration

$$\hat{\mathbf{z}}_k = \mathbf{W}\hat{\mathbf{z}}_{k-1} = \mathbf{W}^k \mathbf{y}. \quad (21)$$

The net effect of each application of \mathbf{W} is essentially a step of anisotropic diffusion [51], [5]. This can be seen as follows. From the iteration (21) we have

$$\hat{\mathbf{z}}_k = \mathbf{W}\hat{\mathbf{z}}_{k-1}, \quad (22)$$

$$= \hat{\mathbf{z}}_{k-1} - \hat{\mathbf{z}}_{k-1} + \mathbf{W}\hat{\mathbf{z}}_{k-1}, \quad (23)$$

$$= \hat{\mathbf{z}}_{k-1} + (\mathbf{W} - \mathbf{I}) \hat{\mathbf{z}}_{k-1}, \quad (24)$$

which we can rewrite as

$$\hat{\mathbf{z}}_k - \hat{\mathbf{z}}_{k-1} = (\mathbf{W} - \mathbf{I}) \hat{\mathbf{z}}_{k-1}. \quad (25)$$

Recall that $\mathbf{W} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{1/2}$. Hence we have

$$\mathbf{W} - \mathbf{I} = \mathbf{D}^{-1/2} (\mathbf{L} - \mathbf{I}) \mathbf{D}^{1/2} = \mathbf{D}^{-1/2} \mathcal{L} \mathbf{D}^{1/2}, \quad (26)$$

where \mathcal{L} is the graph Laplacian operator mentioned earlier [39], [5]. Defining the normalized variable $\bar{\mathbf{z}}_k = \mathbf{D}^{1/2} \hat{\mathbf{z}}_k$, (25) embodies a discrete version of anisotropic diffusion:

$$\bar{\mathbf{z}}_k - \bar{\mathbf{z}}_{k-1} = \mathcal{L} \bar{\mathbf{z}}_{k-1}, \quad \longleftrightarrow \quad \frac{\partial \bar{\mathbf{z}}(t)}{\partial t} = \nabla^2 \bar{\mathbf{z}}(t) \quad \text{Diffusion Eqn.} \quad (27)$$

where the left-hand side of the above is a discretization of the derivative operator $\frac{\partial \bar{\mathbf{z}}(t)}{\partial t}$.

Returning to the diffusion estimate (21), the bias in the estimate after k iterations is

$$\text{bias}_k = \mathbf{E}(\widehat{\mathbf{z}}_k) - \mathbf{z} = \mathbf{E}(\mathbf{W}^k \mathbf{y}) - \mathbf{z} = \mathbf{W}^k \mathbf{z} - \mathbf{z} = (\mathbf{W}^k - \mathbf{I})\mathbf{z}.$$

Recalling that the matrix \mathbf{W} has a unit eigenvalue corresponding to a (constant) vector, we note that since $\mathbf{W}^k - \mathbf{I}$ has the same eigenvectors as \mathbf{W} , the above sequence of estimators produce unbiased estimates of constant images.

The squared magnitude of the bias is given by

$$\|\text{bias}_k\|^2 = \|(\mathbf{W}^k - \mathbf{I})\mathbf{z}\|^2. \quad (28)$$

As before, writing the image \mathbf{z} in the column space of \mathbf{V} as $\mathbf{z} = \mathbf{V}\mathbf{b}_0$, we have:

$$\|\text{bias}_k\|^2 = \|(\mathbf{W}^k - \mathbf{I})\mathbf{z}\|^2 = \|\mathbf{V}(\mathbf{S}^k - \mathbf{I})\mathbf{b}_0\|^2 = \sum_{i=1}^n (\lambda_i^k - 1)^2 b_{0i}^2. \quad (29)$$

As k grows, so does the magnitude of the bias, ultimately converging to $\|\mathbf{b}_0\|^2 - b_{01}^2$. This behavior is consistent with what is observed in practice; namely, increasing iterations of diffusion produce more and more blurry (biased) results. Next, we derive the variance of the diffusion estimator:

$$\text{cov}(\widehat{\mathbf{z}}_k) = \text{cov}(\mathbf{W}^k \mathbf{y}) = \text{cov}(\mathbf{W}^k \mathbf{e}) = \sigma^2 \mathbf{W}^k (\mathbf{W}^k)^T,$$

which gives

$$\text{var}(\widehat{\mathbf{z}}_k) = \text{tr}(\text{cov}(\widehat{\mathbf{z}}_k)) = \sigma^2 \sum_{i=1}^n \lambda_i^{2k}.$$

As k grows, the variance tends to a constant value of σ^2 . Overall, the mean-squared-error (MSE) is given by

$$\text{MSE}_k = \|\text{bias}_k\|^2 + \text{var}(\widehat{\mathbf{z}}_k) = \sum_{i=1}^n (\lambda_i^k - 1)^2 b_{0i}^2 + \sigma^2 \lambda_i^{2k}. \quad (30)$$

Diffusion experiments for denoising three types of patches (shown in Fig. 4) are carried out to illustrate the evolution of MSE. The variance of input (Gaussian) noise is set to $\sigma^2 = 25$. Filters based on both LARK [9] and NLM [7] estimated from the latent noise-free patches are tested, and Sinkhorn algorithm (see Appendix B) is implemented to make the filter matrices doubly-stochastic. Predicted MSE_k , $\text{var}(\widehat{\mathbf{z}}_k)$ and $\|\text{bias}_k\|^2$ according to (30) through the diffusion iteration are shown in Fig. 5 (LARK) and Fig. 7 (NLM), where we can see that diffusion monotonically reduces the estimation variance and increases the bias. In some cases (such as (a) and (b) in Fig. 5) diffusion further suppresses MSE and improves the estimation performance. True MSEs for the standard (asymmetric) filters and their symmetrized versions computed by Monte-Carlo simulations are also shown, where in each simulation 100 independent noise realizations are averaged. We see that the estimated MSEs of the symmetrized filters match quite well

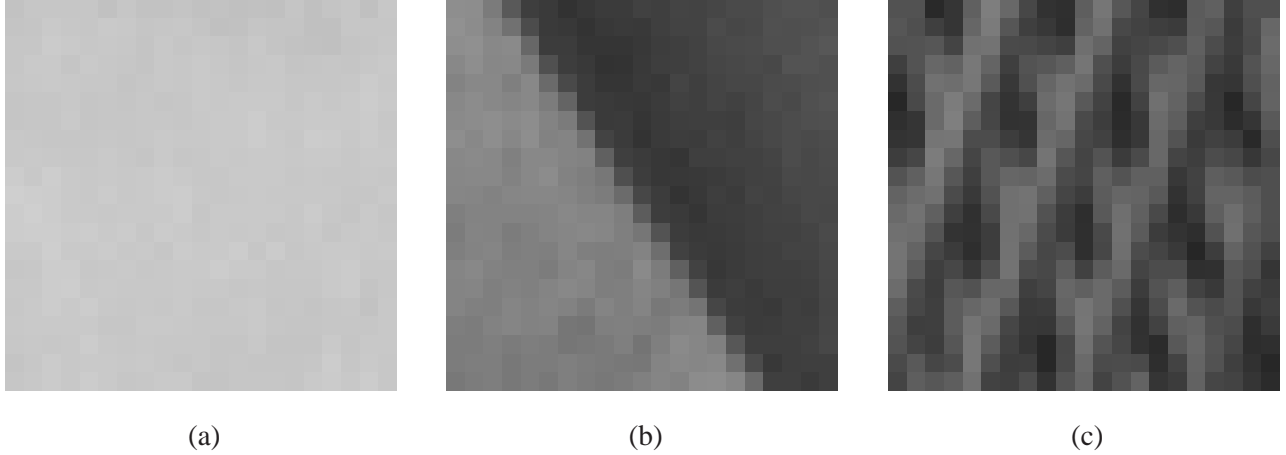


Fig. 4. Example patches: (a) flat, (b) edge, (c) texture.

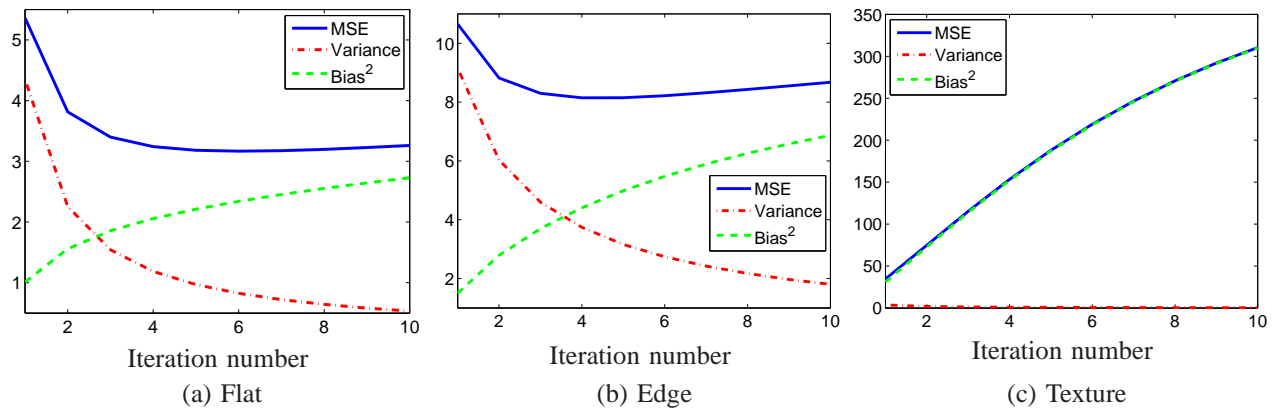


Fig. 5. Plots of predicted MSE_k , $\text{var}(\hat{z}_k)$ and $\|\text{bias}_k\|^2$ using (30) for LARK [9] filters in diffusion process.

with the predicted ones (see Figs. 6, 8). The asymmetric filters, meanwhile generate slightly higher MSE, but behave closely to the symmetrized ones especially in regions around the optimal MSEs.

In the next set of Monte-Carlo simulations, we illustrate the effect of \mathbf{W} being computed from clean vs. noisy images (see Figs. 9 and 10). Noise variance $\sigma^2 = 0.5$, which is relatively small, simulating the situation where “pre-processing” has been applied to suppress estimation variance. It can be observed that the MSEs estimated from Monte-Carlo simulations are quite close to the ones predicted from the ideal filters, which confirms the assumption in Section IV that the filter matrix \mathbf{W} can be treated as deterministic under most circumstances.

To further analyze the change of MSE in the diffusion process, let us consider the contribution of MSE

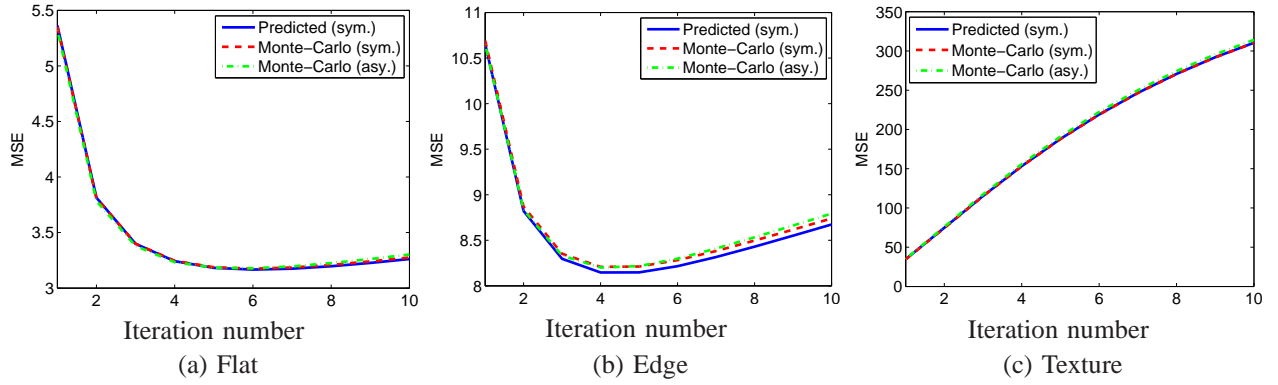


Fig. 6. Plots of predicted MSE and Monte-Carlo estimated MSE in diffusion process using LARK [9] filters. In the Monte-Carlo simulations, results for row-stochastic (asymmetric) filters and their symmetrized versions are both shown.

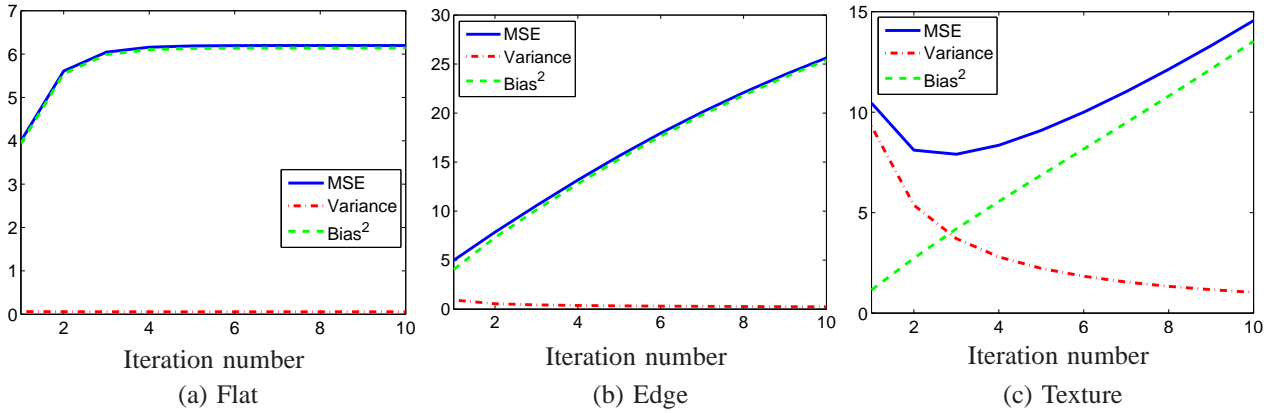


Fig. 7. Plots of predicted MSE_k , $\text{var}(\hat{z}_k)$ and $\|\text{bias}_k\|^2$ using (30) for NLM [7] filters in diffusion process.

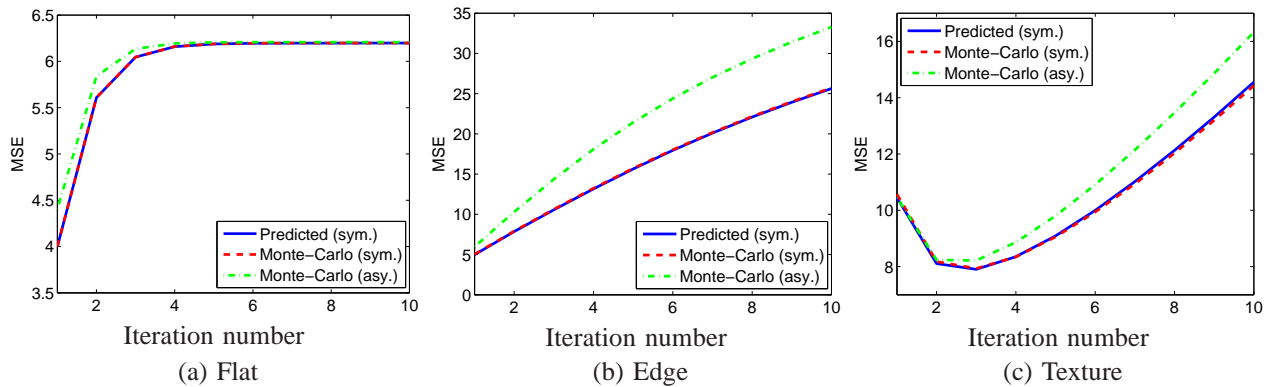


Fig. 8. Plots of predicted MSE and Monte-Carlo estimated MSE in diffusion process using NLM [7] filters. In the Monte-Carlo simulations, result for row-stochastic (asymmetric) filters and their symmetrized versions are both shown.

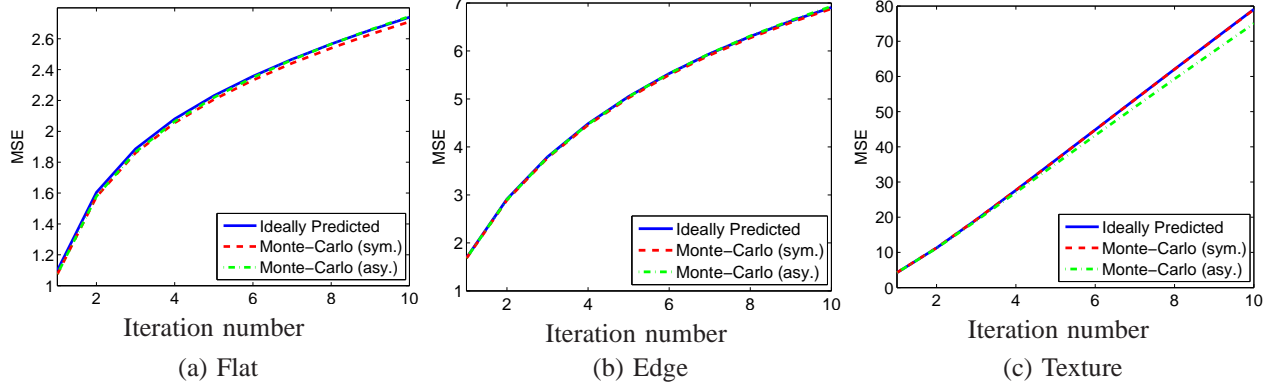


Fig. 9. Plots of predicted MSE in diffusion based on ideally estimated symmetric LARK filters, and MSE of noisy symmetric/asymmetric LARK filters through Monte-Carlo simulations. The input noise variance $\sigma^2 = 0.5$, and for each simulation 100 independent noise realizations are implemented.

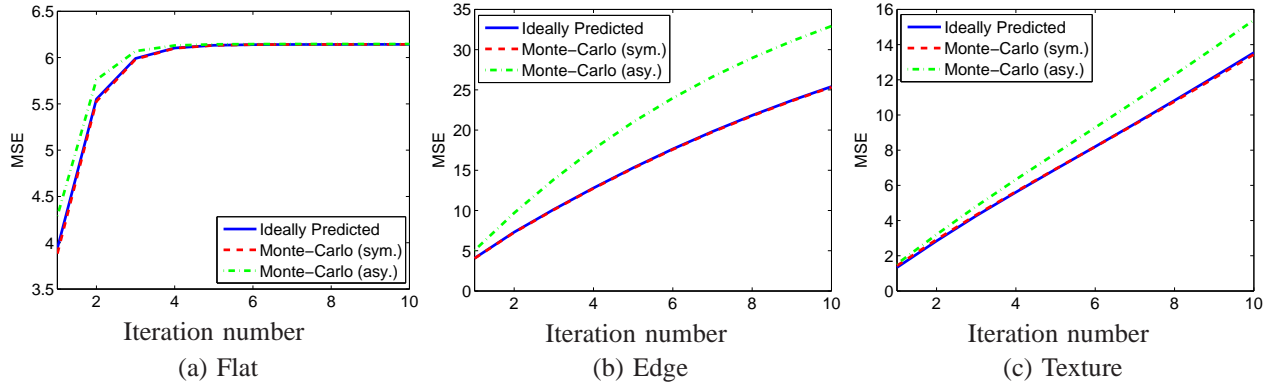


Fig. 10. Plots of predicted MSE in diffusion based on ideally estimated symmetric NLM filters, and MSE of noisy symmetric/asymmetric NLM filters through Monte-Carlo simulations. The input noise variance $\sigma^2 = 0.5$, and for each simulation 100 independent noise realizations are implemented.

in each component (or *mode*) separately:

$$\mathbf{MSE}_k = \sum_{i=1}^n \mathbf{MSE}_k^{(i)} \quad (31)$$

where $\mathbf{MSE}_k^{(i)}$ denotes the MSE of the i -th mode in the k th diffusion iteration, which is given by:

$$\mathbf{MSE}_k^{(i)} = (\lambda_i^k - 1)^2 b_{0i}^2 + \sigma^2 \lambda_i^{2k}. \quad (32)$$

Equivalently, we can write

$$\overline{\mathbf{MSE}}_k^{(i)} = \frac{\mathbf{MSE}_k^{(i)}}{\sigma^2} = \mathbf{snr}_i (\lambda_i^k - 1)^2 + \lambda_i^{2k}, \quad (33)$$

where $\mathbf{snr}_i = \frac{b_{0i}^2}{\sigma^2}$ is defined as the signal to noise ratio in the i -th mode.

One may be interested to know at which iteration $\text{MSE}_k^{(i)}$ is minimized. As we have seen experimentally, with an arbitrary filter \mathbf{W} , there is no guarantee that even the very first iteration of diffusion will improve the estimate in the MSE sense. The derivative of (33) with respect to k is

$$\frac{\partial \overline{\text{MSE}}_k^{(i)}}{\partial k} = 2\lambda_i^k \log \lambda_i \left[(\text{snr}_i + 1)\lambda_i^k - \text{snr}_i \right]. \quad (34)$$

For $i \geq 2$, since $\lambda_i < 1$ we note that the sign of the derivative depends on the term inside the brackets, namely, $(\text{snr}_i + 1)\lambda_i^k - \text{snr}_i$. To guarantee that in the k th iteration the MSE of the i -th mode decreases (i.e. negative derivative), we must have:

$$(\text{snr}_i + 1)\lambda_i^q - \text{snr}_i > 0 \quad \text{for any } 0 \leq q < k. \quad (35)$$

In fact the condition

$$(\text{snr}_i + 1)\lambda_i^k - \text{snr}_i > 0 \quad (36)$$

guarantees that the derivative is always negative for $q \in (0, k)$ because given any scalar $t = \frac{q}{k} \in (0, 1)$:

$$(\text{snr}_i + 1)\lambda_i^k - \text{snr}_i > 0 \Rightarrow \lambda_i^{tk} > \left(\frac{\text{snr}_i}{\text{snr}_i + 1} \right)^t \Rightarrow (\text{snr}_i + 1)\lambda_i^{tk} > \text{snr}_i \left(\frac{\text{snr}_i}{\text{snr}_i + 1} \right)^{t-1} > \text{snr}_i \quad (37)$$

The condition (36) has an interesting interpretation. Rewriting (36) we have:

$$\log(1 + \text{snr}_i) < \underbrace{\log \left(\frac{1}{1 - \lambda_i^k} \right)}_{\epsilon_i}, \quad (38)$$

where $\lambda_i^k = \lambda_i^k$ denotes the i -th eigenvalue of \mathbf{W}^k . The left-hand side of the inequality is Shannon's channel capacity [56] of the i -th mode for the problem $\mathbf{y} = \mathbf{z} + \mathbf{e}$. We name the right-hand side expression ϵ_i the *entropy* of the i -th mode of the filter \mathbf{W}^k . The larger ϵ_i , the more variability can be expected in the output image produced by the i -th mode of the denoising filter. The above condition implies that if the filter entropy exceeds or equals the channel capacity of the i -th mode, then the k -th iteration of diffusion will in fact produce an improvement in the corresponding mode. One may also be interested in identifying an approximate value of k for which the overall MSE_k is minimized. This depends on the signal energy distribution ($\{b_{0i}^2\}$) over all the modes, which is generally not known, but for which we may have some prior in mind. But for a given mode i , the minimum $\text{MSE}^{(i)}$ is achieved in the k^* -th iteration, where

$$k^* = \log \left(\frac{\text{snr}_i}{\text{snr}_i + 1} \right) / \log \lambda_i. \quad (39)$$

Of course, in practice we can seldom find a single iteration number k^* minimizing $\text{MSE}^{(i)}$ for *all* the modes simultaneously; but in general, the take-away lesson is that optimizing a given filter through

diffusion can (under proper conditions (38)) make its corresponding eigenvalues closer to those of an ideal Wiener filter that minimizes MSE.

Another interesting question is this: given a particular \mathbf{W} , how much further MSE reduction can be achieved by implementing diffusion? For example, can we use diffusion to improve the 3D collaborative Wiener filter in BM3D [48]? Let us assume that this filter has already approximately achieved the ideal Wiener filter condition in each mode, namely

$$\lambda_i^* \approx \frac{\text{snr}_i}{\text{snr}_i + 1}. \quad (40)$$

Then we can see that:

$$(\text{snr}_i + 1)(\lambda_i^*)^k - \text{snr}_i < 0 \quad \text{for any } k > 1, \quad (41)$$

which means for all the modes, diffusion will definitely worsen (increase) the MSE. In other words, multi-iteration diffusion is not useful for filters where the Wiener condition has been (approximately) achieved.

B. Twicing, ℓ_2 -Boosting, Reaction-Diffusion, and Bregman Iterations

An alternative to repeated applications of the filter \mathbf{W} is to consider the *residual* signals, defined as the difference between the estimated signal and the measured signal. The use of the residuals in improving estimates has a rather long history, dating at least back to the work of John Tukey, who in his landmark book [54], termed the idea “twicing”. More recently, the idea has been proposed in the applied mathematics community for application to image restoration problems under the rubric of *Bregman* iterations [52]. And in the machine learning and statistics literature, [53] recently proposed the idea of ℓ_2 -*boosting*, which may be viewed as an (ℓ_2) regression counterpart of the more familiar notion of boosting in classification, originally proposed by Freund and Schapire [57]. All of the aforementioned ways of looking at the use of residuals are independently quite rich and worth exploring on their own. Indeed, for a wide-angle view of the boosting approach to regression and regularization, and its connections to functional gradient descent interpretations, we can refer the interested reader to the nice treatment in [58]. The general idea of Bregman divergence, and its use in solving large-scale optimization problems is also a very deep area of work, for which we can refer the interested reader to the book [59] in which both the foundations and some applications to signal processing are discussed. Also, in [60], the ideas of Bregman divergence and boosting are connected.

In whatever guise, the basic notion – to paraphrase Tukey, is to use the filtered residuals to enhance the estimate by adding some “roughness” to it. Put another way, if the residuals contain some of the underlying signal, filtering them should recover this left-over signal at least in part.

Formally, the residuals are defined as the difference between the estimated signal and the measured signal: $\mathbf{r}_k = \mathbf{y} - \hat{\mathbf{z}}_{k-1}$, where here we define the initialization¹³ $\hat{\mathbf{z}}_0 = \mathbf{W}\mathbf{y}$. With this definition, we write the iterated estimates as

$$\hat{\mathbf{z}}_k = \hat{\mathbf{z}}_{k-1} + \mathbf{W} \mathbf{r}_k = \hat{\mathbf{z}}_{k-1} + \mathbf{W}(\mathbf{y} - \hat{\mathbf{z}}_{k-1}). \quad (42)$$

These estimates too trade off bias against variance with increasing iterations, though in a fundamentally different way than the diffusion estimator we discussed earlier. Indeed, we immediately notice that this iteration has a very different asymptotic behavior than diffusion. Namely, as $k \rightarrow \infty$, the estimate here tends back towards the original data y , whereas the diffusion tends to a constant!

For an intuitive interpretation, note that for $k = 1$, the (residual) iteration above gives

$$\hat{\mathbf{z}}_1 = \hat{\mathbf{z}}_0 + \mathbf{W}(\mathbf{y} - \hat{\mathbf{z}}_0) = \mathbf{W}\mathbf{y} + \mathbf{W}(\mathbf{y} - \mathbf{W}\mathbf{y}) = (2\mathbf{W} - \mathbf{W}^2) \mathbf{y}.$$

This first iterate $\hat{\mathbf{z}}_1$ is in fact precisely the original “twicing” estimate of Tukey [54]. More recently, this particular filter has been suggested (i.e. rediscovered) in other contexts. In [39], for instance, Coifman et al. suggested it as an ad-hoc way to enhance the effectiveness of diffusion. The intuitive justification given [5] was that $(2\mathbf{W} - \mathbf{W}^2) = (2\mathbf{I} - \mathbf{W}) \mathbf{W}$ can be considered as a two step process; namely blurring, or diffusion (as embodied by \mathbf{W}), followed by an additional step of *inverse* diffusion or sharpening (as embodied by $2\mathbf{I} - \mathbf{W}$.) As can be seen, the ad-hoc suggestion has a clear interpretation here. Furthermore, we see that the estimate $\hat{\mathbf{z}}_1$ can also be thought of as a kind of nonlinear, adaptive *unsharp masking* process, further clarifying its effect. In [61] this procedure and its relationship to the Bregman iterations were studied in detail¹⁴.

To further understand the iteration (42) in comparison to the earlier diffusion framework, consider

¹³Note that this initialization is a *once-filtered* version of the noisy data using some initial filter \mathbf{W} , and different from $(\mathbf{z}_0 = \mathbf{y})$ used in the diffusion iterations.

¹⁴For the sake of completeness, we also mention in passing that in the non-stochastic setting and where \mathbf{W} is rank 1, the residual-based iterations (42) are known as *matching pursuit* [62]. This approach that has found many applications and extensions for fitting over-complete dictionaries.

again a very similar algebraic manipulation of (42):

$$\widehat{\mathbf{z}}_k = \widehat{\mathbf{z}}_{k-1} + \mathbf{W}(\mathbf{y} - \widehat{\mathbf{z}}_{k-1}), \quad (43)$$

$$= (\mathbf{I} - \mathbf{W}) \widehat{\mathbf{z}}_{k-1} + \mathbf{W}\mathbf{y}, \quad (44)$$

$$\widehat{\mathbf{z}}_k - \widehat{\mathbf{z}}_{k-1} = (\mathbf{I} - \mathbf{W}) \widehat{\mathbf{z}}_{k-1} + (\widehat{\mathbf{z}}_0 - \widehat{\mathbf{z}}_{k-1}). \quad (45)$$

Redefining the graph Laplacian ¹⁵

$$\mathbf{I} - \mathbf{W} = \mathbf{D}^{-1/2} (\mathbf{I} - \mathbf{L}) \mathbf{D}^{1/2} = \mathbf{D}^{-1/2} \mathcal{L} \mathbf{D}^{1/2},$$

with a change of variable $\bar{\mathbf{z}}_k = \mathbf{D}^{1/2} \widehat{\mathbf{z}}_k$, we can write

$$\underbrace{\bar{\mathbf{z}}_k - \bar{\mathbf{z}}_{k-1}}_{\text{diffusion}} = \underbrace{\mathcal{L} \bar{\mathbf{z}}_{k-1}}_{\text{diffusion}} + \underbrace{(\bar{\mathbf{z}}_0 - \bar{\mathbf{z}}_{k-1})}_{\text{reaction term}}, \quad \longleftrightarrow \quad \underbrace{\frac{\partial \bar{\mathbf{z}}(t)}{\partial t}}_{\text{diffusion}} = \underbrace{\nabla^2 \bar{\mathbf{z}}(t)}_{\text{diffusion}} + \underbrace{(\bar{\mathbf{z}}(0) - \bar{\mathbf{z}}(t))}_{\text{reaction term}} \quad (46)$$

The above is a modified version of the diffusion equation studied earlier, where the modification takes the form of a “forcing” function, which is in fact the residual between the estimate at time t and a *prefiltered* version of the original data. This additional term reacts against the strict smoothing effect of the standard diffusion, and results in a non-trivial steady-state. Not coincidentally, the notion of a “biased” anisotropic diffusion [55] is essentially identical to the above formulation. More recently, Farbman et al. [63] proposed a variational formulation which also leads precisely to the present setup. This will become more clear as we discuss the connection to (empirical) Bayesian approaches in Section VI.

The residual-based iterations also have fundamentally different statistical properties than diffusion. To study the statistical behavior of the estimates, we rewrite the iterative process in (42) more explicitly in terms of the data \mathbf{y} . We have [40], [53]:

$$\widehat{\mathbf{z}}_k = \sum_{j=0}^k \mathbf{W}(\mathbf{I} - \mathbf{W})^j \mathbf{y} = \left(\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1} \right) \mathbf{y}. \quad (47)$$

Clearly the above iteration does not monotonically blur the data; but a rather more interesting behavior for the bias-variance tradeoff emerges. Analogous to our earlier derivation, the bias in the estimate after k iterations is

$$\text{bias}_k = \mathbf{E}(\widehat{\mathbf{z}}_k) - \mathbf{z} = \left(\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1} \right) \mathbf{E}(\mathbf{y}) - \mathbf{z} = -(\mathbf{I} - \mathbf{W})^{k+1} \mathbf{z}.$$

The squared magnitude of the bias is given by

$$\|\text{bias}_k\|^2 = \|(\mathbf{I} - \mathbf{W})^{k+1} \mathbf{z}\|^2 = \sum_{i=1}^n (1 - \lambda_i)^{2k+2} b_{0i}^2. \quad (48)$$

¹⁵Note that this definition of the graph Laplacian differs by a negative sign from the earlier one in (26).

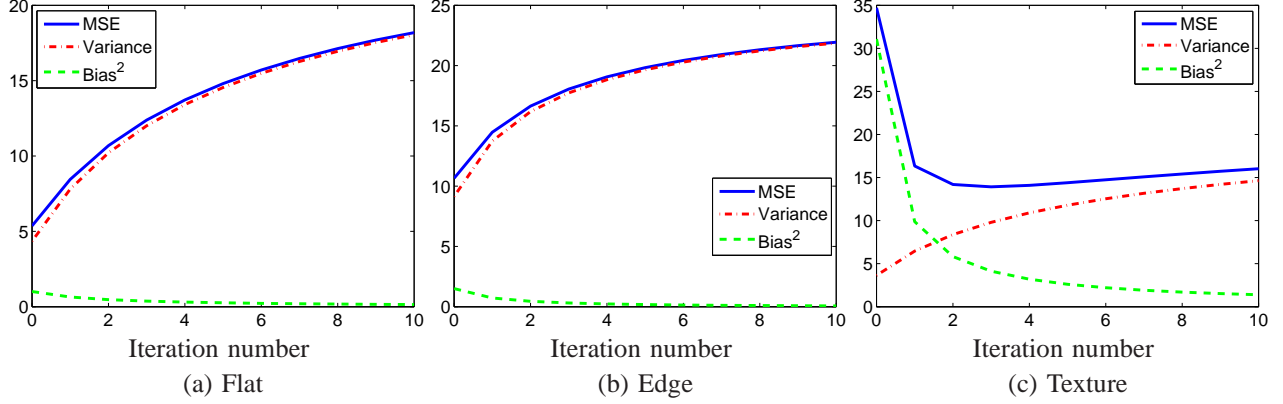


Fig. 11. Plots of predicted MSE_k , $\text{var}(\hat{\mathbf{z}}_k)$ and $\|\text{bias}_k\|^2$ using (49) for LARK [9] filters in the twicing process.

The behavior of the (decaying) bias in this setting is in stark contrast to the (increasing) bias of the diffusion process.

The variance of the estimator and the corresponding mean-squared error are:

$$\text{cov}(\hat{\mathbf{z}}_k) = \text{cov} \left[\left(\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1} \right) \mathbf{y} \right] = \sigma^2 \left(\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1} \right) \left(\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1} \right)^T,$$

which gives

$$\text{var}(\hat{\mathbf{z}}_k) = \text{tr}(\text{cov}(\hat{\mathbf{z}}_k)) = \sigma^2 \sum_{i=1}^n \left(1 - (1 - \lambda_i)^{k+1} \right)^2.$$

As k grows, the variance tends to a constant value of σ^2 . Overall, the mean-squared-error (MSE) is

$$\text{MSE}_k = \|\text{bias}_k\|^2 + \text{var}(\hat{\mathbf{z}}_k) = \sum_{i=1}^n (1 - \lambda_i)^{2k+2} b_{0i}^2 + \sigma^2 \left(1 - (1 - \lambda_i)^{k+1} \right)^2. \quad (49)$$

Experiments for denoising the patches given in Fig. 4 using the “twicing” approach are carried out, where the same doubly-stochastic LARK and NLM filters used for the diffusion tests in Fig. 5 and 7 are employed. Plots of predicted MSE_k , $\text{var}(\hat{\mathbf{z}}_k)$ and $\|\text{bias}_k\|^2$ with respect to iteration are illustrated in Fig. 11 (LARK) and Fig. 13 (NLM). It can be observed that in contrast to diffusion, twicing monotonically reduces the estimation bias and increases the variance. So twicing may in fact reduce the MSE in cases where diffusion fails to do so (such as (c) in Fig. 5). True MSEs for the asymmetric filters and their symmetrized versions are estimated through Monte-Carlo simulations, and they all match very well with the predicted ones (see Fig. 12, 14).

Similar to the diffusion analysis before, we plot MSE resulting from the use of \mathbf{W} filters directly estimated from noisy patches through Monte-Carlo simulations with noise variance $\sigma^2 = 0.5$, and compare

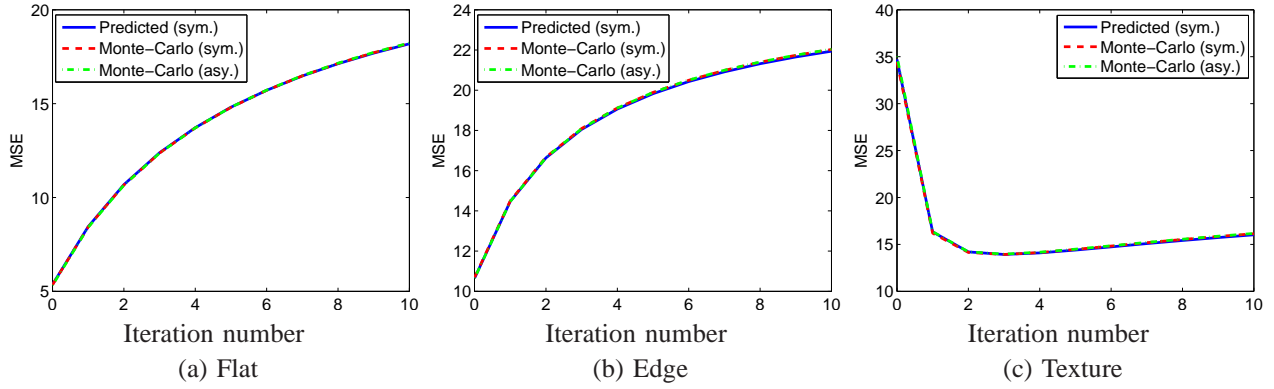


Fig. 12. Plots of predicted MSE and Monte-Carlo estimated MSE in twicing process using LARK [9] filters. In the Monte-Carlo simulations, row-stochastic (asymmetric) filters and their symmetrized versions are all tested.

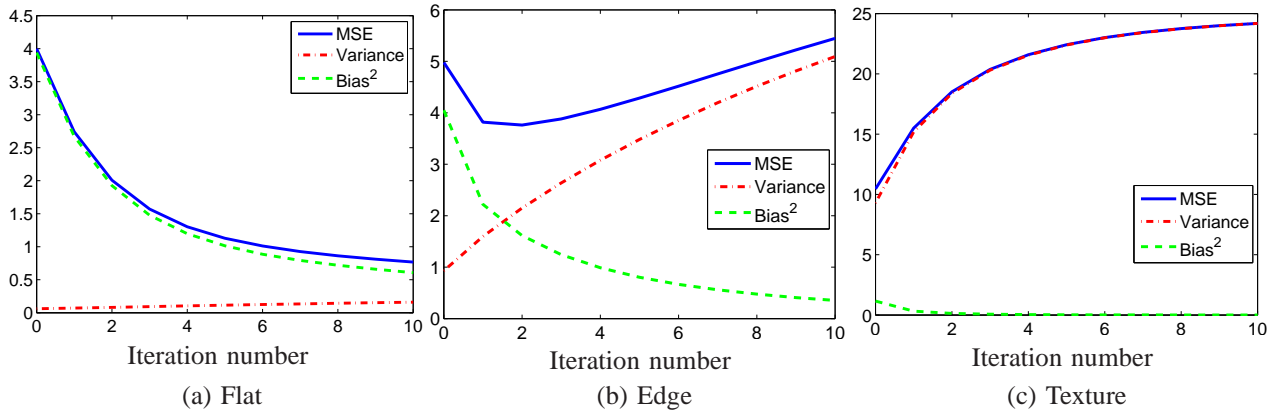


Fig. 13. Plots of predicted MSE_k , $\text{var}(\hat{z}_k)$ and $\|\text{bias}_k\|^2$ using (49) for NLM [7] filters in twicing process.

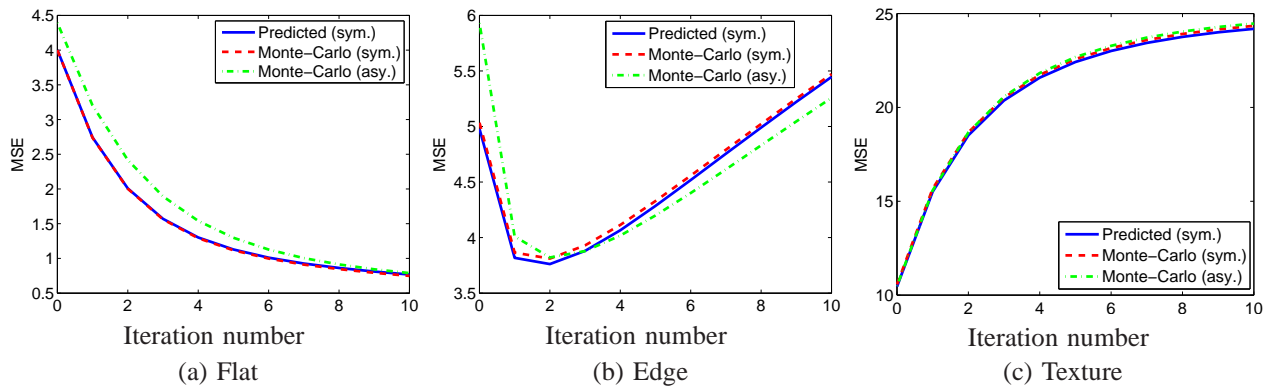


Fig. 14. Plots of predicted MSE and Monte-Carlo estimated MSE in twicing process using NLM [7] filters. In the Monte-Carlo simulations, row-stochastic (asymmetric) filters and their symmetrized versions are all tested.

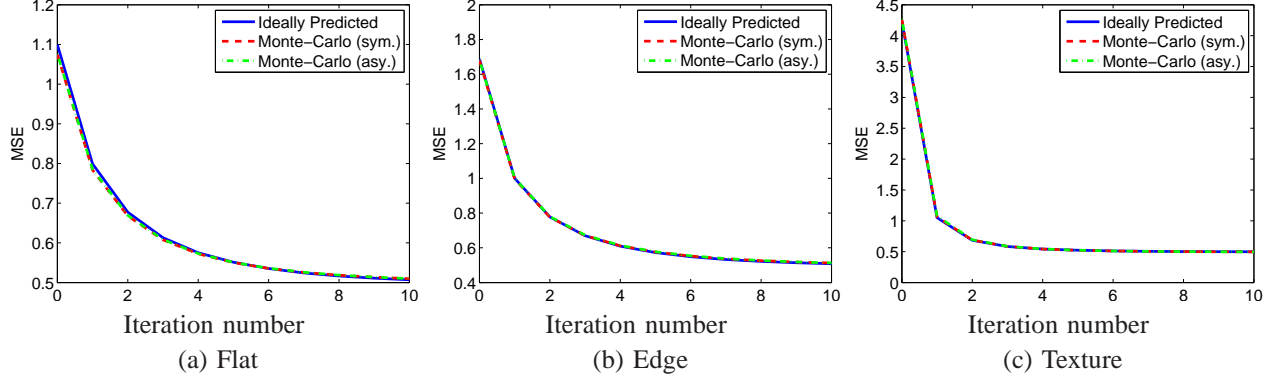


Fig. 15. Plots of predicted MSE in twicing based on ideally estimated symmetric LARK filters, and MSE of noisy symmetric/asymmetric LARK filters through Monte-Carlo simulations. The input noise variance $\sigma^2 = 0.5$, and for each simulation 100 independent noise realizations are implemented.

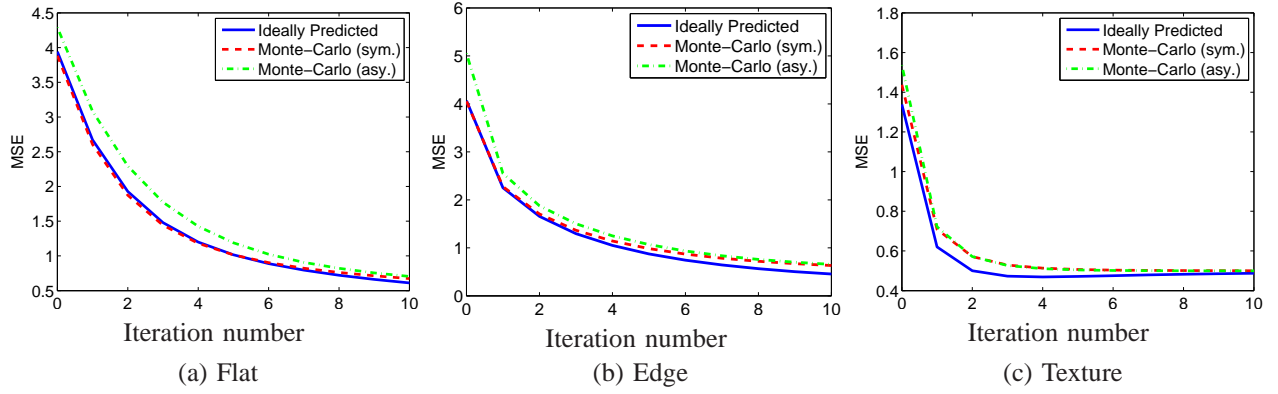


Fig. 16. Plots of predicted MSE in diffusion based on ideally estimated symmetric NLM filters, and MSE of noisy symmetric/asymmetric NLM filters through Monte-Carlo simulations. The input noise variance $\sigma^2 = 0.5$, and for each simulation 100 independent noise realizations are implemented.

them with the predicted MSEs from the ideally estimated filters. Again, the MSEs estimated from Monte-Carlo simulations are quite close to the predicted data. (See Figs. 15 and 16.)

Here again, the contribution to \mathbf{MSE}_k of the i -th mode can be written as:

$$\mathbf{MSE}_k^{(i)} = (1 - \lambda_i)^{2k+2} b_{0i}^2 + \sigma^2 \left(1 - (1 - \lambda_i)^{k+1} \right)^2 \quad (50)$$

Proceeding in a fashion similar to the diffusion case, we can analyze the derivative of $\overline{\mathbf{MSE}}_k^{(i)}$ with respect to k to see whether the iterations improve the estimate. We have:

$$\frac{\partial \overline{\mathbf{MSE}}_k^{(i)}}{\partial k} = 2(1 - \lambda_i)^{k+1} \log(1 - \lambda_i) \left[(\mathbf{snr}_i + 1)(1 - \lambda_i)^{k+1} - 1 \right]. \quad (51)$$

Reasoning along parallel lines to the earlier discussion, the following condition guarantees that the k -th

iteration improves the estimate:

$$(\mathbf{snr}_i + 1)(1 - \lambda_i)^{k+1} > 1 \tag{52}$$

Rewriting the above we have

$$\log(1 + \mathbf{snr}_i) > \log\left(\frac{1}{1 - \lambda'_i}\right) \tag{53}$$

where now $\lambda'_i = 1 - (1 - \lambda_i)^{k+1}$ is the i -th eigenvalue of $\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k+1}$, which is the equivalent filter matrix for the k -th twicing iteration. This is the reverse of the condition we derived earlier; namely, here we observe that if the entropy does *not* exceed the channel capacity, then iterating with the residuals will indeed produce an improvement. This makes reasonable sense because if the filter removes too much detail from the denoised estimate, this “lost” detail is contained in the residuals, which the iterations will attempt to return to the estimate. Again, the minimum $\text{MSE}_k^{(i)}$ is achieved at the k^* -th iteration, where

$$k^* = -\log(1 + \mathbf{snr}_i) / \log(1 - \lambda'_i) - 1, \tag{54}$$

which is when the i -th eigenvalue of the matrix $\mathbf{I} - (\mathbf{I} - \mathbf{W})^{k^*+1}$ becomes $\mathbf{snr}_i / (\mathbf{snr}_i + 1)$, the Wiener condition. In most cases we cannot optimize all the modes with a uniform number of iterations, but under proper conditions (53) twicing can make the eigenvalues closer to those of the ideal Wiener filter.

From the above analysis, we can see that it is possible to improve the performance of many existing denoising algorithms in the MSE sense by implementing iterative filtering. However, to choose either diffusion or twicing and to determine the optimal iteration number require prior knowledge (or estimation) of the latent signal energy $\{b_{0_i}^2\}$ (or the SNR,) and this is an ongoing challenge. Research on estimating the SNR of image or video data without a reference “ground truth” is of broad interest in the community as of late [64]. In particular, this problem points to potentially important connections with ongoing work in no-reference image quality assessment [65] as well.

VI. RELATIONSHIP TO (EMPIRICAL) BAYES AND REGULARIZATION APPROACHES

It is interesting to contrast the adaptive non-parametric framework described so far with the more classical Bayesian estimation approaches. Of course, Bayesian thinking requires that a “prior” be specified independently of the measured data, which characterizes the aggregate behavior of the underlying latent image of interest. In the context we have described so far, no specific prior information was used. Instead, data-dependant weights were specified in a least-squares framework, yielding locally adaptive filters. On the surface then, it may seem that there is no direct relationship between the two approaches. However, as we will illustrate below, specifying the kernel function or the corresponding weights is essentially

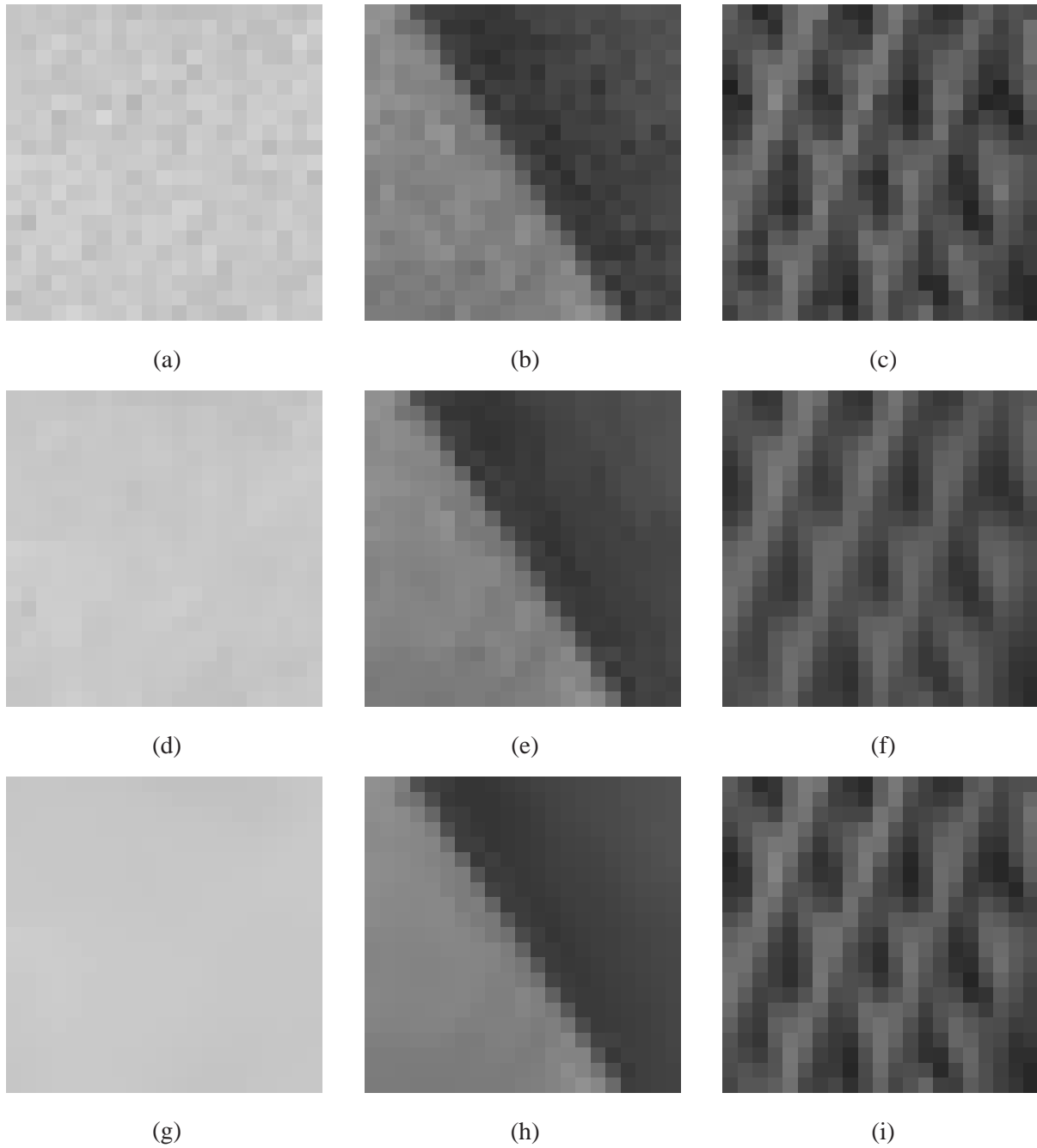


Fig. 17. Denoising example using iterative LARK filtering: (a)-(c) input noisy patches; noise variance $\sigma^2 = 25$. (d)-(f) output patches by filtering once. (g)-(i) MSE optimized outputs: (g) is the 6th diffusion output; (h) is the 4th diffusion output; (i) is the 3rd twicing output.

equivalent to *estimating* a particular type of “prior” from the given data¹⁶. This empirical Bayesian approach has in fact a long and useful history [66] in the statistics literature. Furthermore, recently another class of modern patch-based filtering methods based on the empirical Bayesian framework were pioneered in [67] and extended in [68] and [69] based on the notion of “adaptive” (i.e. data-dependent) priors. In the same spirit, we illustrate below that an empirical version of the popular maximum a-posteriori method has a data-dependent counterpart in the non-parametric techniques we described so far.

A canonical Bayesian approach is to specify a prior for the unknown \mathbf{z} , or to equivalently use a regularization functional. The so-called *maximum a-posteriori* estimate is then given by the solution of the following optimization problem:

$$\text{Maximum a-posteriori (MAP):} \quad \hat{\mathbf{z}} = \arg \min_{\mathbf{z}} \frac{1}{2} \|\mathbf{y} - \mathbf{z}\|^2 + \frac{\lambda}{2} \mathcal{R}(\mathbf{z}) \quad (55)$$

where the first term on the right-hand side is the data-fidelity (log-likelihood) term, and the second (regularization, or log-prior) term essentially enforces a soft constraint on the global structure and smoothness of the signal. In the MAP approach, the regularization term is a functional $\mathcal{R}(\mathbf{z})$ which is typically (but not always) convex, to yield a unique minimum for the overall cost. Particularly popular recent examples include $\mathcal{R}(\mathbf{z}) = \|\nabla \mathbf{z}\|$, and $\mathcal{R}(\mathbf{z}) = \|\mathbf{z}\|_1$. The above approach implicitly contains a global (Gaussian) model of the noise (captured by the quadratic data-fidelity term) and an explicit model of the signal (captured by the regularization term.)

Regardless of what choice we make for the regularization functional, this approach is *global* in the sense that the explicit assumptions about the noise and the signal constrain the degrees of freedom in the solution, hence limiting the global behavior of the estimate. This often results in well-behaved (but not always desirable) solutions. This is precisely the main motivation for using (local/non-local) adaptive non-parametric techniques in place of global regularization methods. Indeed, though the effect of regularization is not explicitly present in the non-parametric framework, its work is implicitly done by the design of the kernel function $K(\cdot)$, which affords us local control, and therefore more freedom and often better adaptation to the given data, resulting in more powerful techniques with broader applicability.

Another important advantage of the local non-parametric methods is that the resulting algorithms inherently operate in relatively small windows, and can hence be efficiently implemented and often easily parallelized. With global parametric and variational methods, the solutions are often implemented

¹⁶One cringes to call this estimate a “prior” any longer, but we will do so for lack of a better word.

using large-scale global optimization techniques, which can be hard (sometimes even impossible) to “kernelize,” for algorithmic efficiency [33].

Though we do not discuss them here, of course hybrid methods are also possible, where the first term in (55) is replaced by the corresponding weighted term (3), with regularization also applied explicitly [70]. As we shall see below, the iterative methods for improving the non-parametric estimates (such as diffusion, twicing, etc.) mirror the behavior of steepest descent iterations in the empirical Bayesian case, but with a fundamental difference. Namely, the corresponding implied regularization functionals deployed by the non-parametric methods are significantly different, adapting directly to the given data.

To connect MAP to the non-parametric setting, we proceed in similar lines as Elad [3] by considering the simplest iterative approach: the steepest descent (SD) method. The SD iteration for MAP, with fixed step size μ is

$$\text{MAP:} \quad \hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k - \mu [(\hat{\mathbf{z}}_k - \mathbf{y}) + \lambda \nabla \mathcal{R}(\hat{\mathbf{z}}_k)]. \quad (56)$$

A. Empirical Bayes Interpretation of Diffusion

Let us compare the diffusion iterations (25) to the steepest descent MAP iterations above in (56). Namely, we equate the right-hand sides of

$$\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k - \mu [(\hat{\mathbf{z}}_k - \mathbf{y}) + \lambda \nabla \mathcal{R}(\hat{\mathbf{z}}_k)] \quad (57)$$

$$\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k + (\mathbf{W} - \mathbf{I}) \hat{\mathbf{z}}_k \quad (58)$$

which gives

$$-\mu [(\hat{\mathbf{z}}_k - \mathbf{y}) + \lambda \nabla \mathcal{R}(\hat{\mathbf{z}}_k)] = (\mathbf{W} - \mathbf{I}) \hat{\mathbf{z}}_k.$$

Solving for $\nabla \mathcal{R}(\hat{\mathbf{z}}_k)$ we obtain

$$\nabla \mathcal{R}(\hat{\mathbf{z}}_k) = \frac{1}{\mu\lambda} (\mathbf{W} - (1 - \mu)\mathbf{I}) (\mathbf{y} - \hat{\mathbf{z}}_k) - \frac{1}{\mu\lambda} (\mathbf{I} - \mathbf{W}) \mathbf{y} \quad (59)$$

When \mathbf{W} is symmetric¹⁷, integrating both sides we have (to within an additive constant:)

$$\mathcal{R}_{\text{diffusion}}(\mathbf{z}) = \frac{1}{2\mu\lambda} (\mathbf{y} - \mathbf{z})^T ((1 - \mu)\mathbf{I} - \mathbf{W}) (\mathbf{y} - \mathbf{z}) + \frac{1}{\mu\lambda} \mathbf{y}^T (\mathbf{I} - \mathbf{W}) \mathbf{z}. \quad (60)$$

We observe therefore that the equivalent regularization term is a quadratic function of the *data and residuals*. In particular, the function $\mathcal{R}_{\text{diffusion}}(\hat{\mathbf{z}}_k)$ can be thought of as an *estimated* log-prior at each

¹⁷Since the left-hand side of (59) is the gradient of a scalar-valued function, the right-hand side must be curl-free. This is possible if and only if \mathbf{W} is symmetric.

iteration as it depends on the data \mathbf{y} , the last estimate $\hat{\mathbf{z}}_k$, and weights \mathbf{W} which are computed from the data. More specifically, the implied empirical Bayes “prior” would be

$$\hat{p}(\mathbf{z}) = c \exp [-\mathcal{R}(\hat{\mathbf{z}})]$$

where c is a normalization constant. The empirical (MAP) Bayesian interpretation of the residual-based method follows similarly, but is somewhat surprisingly more straightforward, as we illustrate below.

B. Empirical Bayes Interpretation of the Residual Iterations

We arrive at the empirical Bayes interpretation of the residual iterations (42) in a similar manner. In particular, comparing (42) to the steepest descent MAP iterations in (56) we have

$$\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k - \mu [(\hat{\mathbf{z}}_k - \mathbf{y}) + \lambda \nabla \mathcal{R}(\hat{\mathbf{z}}_k)] \quad (61)$$

$$\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k + \mathbf{W}(\mathbf{y} - \hat{\mathbf{z}}_k) \quad (62)$$

Again, equating the right-hand sides we get:

$$-\mu [(\hat{\mathbf{z}}_k - \mathbf{y}) + \lambda \nabla \mathcal{R}(\hat{\mathbf{z}}_k)] = \mathbf{W}(\mathbf{y} - \hat{\mathbf{z}}_k).$$

Solving for $\nabla \mathcal{R}(\hat{\mathbf{z}}_k)$ we obtain

$$\nabla \mathcal{R}(\hat{\mathbf{z}}_k) = \frac{1}{\mu\lambda} (\mathbf{W} - \mu\mathbf{I}) (\mathbf{y} - \hat{\mathbf{z}}_k), \quad (63)$$

With \mathbf{W} symmetric, we have (to within an additive constant:)

$$\mathcal{R}_{\text{residual}}(\mathbf{z}) = \frac{1}{2\mu\lambda} (\mathbf{y} - \mathbf{z})^T (\mu\mathbf{I} - \mathbf{W}) (\mathbf{y} - \mathbf{z}) \quad (64)$$

Therefore the (implicit) regularization term is an adaptive function of the data and residuals. Once again, the empirically *estimated* prior is of the same form $\hat{p}(\mathbf{z}) = c \exp [-\mathcal{R}(\hat{\mathbf{z}})]$ where $\mathcal{R}(\hat{\mathbf{z}})$ is computed directly on the residuals $\mathbf{y} - \hat{\mathbf{z}}$ of the filtering process at every iteration. It is especially interesting to note that this empirical prior is a quadratic form in the *residuals* alone where, not coincidentally, the matrix $\mu\mathbf{I} - \mathbf{W}$ is closely related to the Laplacian operator defined earlier. This hints directly at the edge-preserving behavior of this type of iteration; a property not shared by the diffusion process, which monotonically blurs the data.

VII. CONCLUDING REMARKS

It has been said that in both literature and in film, there are only seven basic plots (comedy, tragedy, etc.) – that all stories are combinations or variations on these basic themes. Perhaps it is taking the analogy too far to say that an exact parallel exists in our field as well. But it is fair to argue that the basic tools of our trade in recent years have revolved around a small number of key concepts as well, which I tried to highlight in this paper.

- The most successful modern approaches in image and video processing are **non-parametric**. We have drifted away from model-based methods which have dominated signal processing for decades.
- In one-dimensional signal processing, there is a long history of design and analysis of adaptive filtering algorithms. A corresponding line of thought has only recently become the dominant paradigm in processing higher-dimensional data. Indeed, **adaptivity** to data is a central theme of all the algorithms and techniques discussed here, which represent a snapshot of the state of the art.
- The traditional approach that every graduate student of our field learns in class is to carefully design a filter, apply it to the given data, and call it a day. In contrast, many of the most successful recent approaches involve repeated applications of a filter or sequence of filters to data, and aggregating the results. Such ideas have been around for some time in statistics, machine learning, and elsewhere, but we have just begun to make careful use of **sophisticated iteration and boosting** mechanisms.

As I highlighted here, deep connections between techniques used commonly in computer vision, image processing, graphics, and machine learning exist. The practitioners of these fields have been using each other's techniques either implicitly or explicitly for a while. The pace of this convergence has quickened, and this is not a coincidence. It has come about through many years of scientific iteration in what my late friend and colleague Gene Golub called the “serendipity of science” – an aptitude we have all developed for making desirable discoveries by happy accident.

VIII. ACKNOWLEDGEMENTS

I thank the Editorial Board of the IEEE Signal Processing Magazine, and Prof. Antonio Ortega in particular, for their support and valuable feedback. I am grateful to all my students in the MDSP research group at UC Santa Cruz for their insights and comments on earlier versions of this manuscript. In particular, I acknowledge the diligent help of Xiang Zhu with many of the simulations contained here. I am also thankful for feedback and encouragement of my friends and colleagues Prof. Michael Elad of the Technion and Prof. Bernd Girod of Stanford. This work was supported in part by the US Air Force Grant FA9550-07-1-0365 and the National Science Foundation Grant CCF-1016018.

APPENDIX A

GENERALIZATION OF THE NON-PARAMETRIC FRAMEWORK TO ARBITRARY BASES

The non-parametric approach in (3) can be further extended to include a more general model of the signal $z(x)$ in some appropriate basis. Namely, expanding the regression function $z(x)$ in a desired basis ϕ_l , we can formulate the following optimization problem:

$$\hat{z}(x_j) = \arg \min_{\beta_l(x_j)} \sum_{i=1}^n \left[y_i - \sum_{l=0}^N \beta_l(x_j) \phi_l(x_i, x_j) \right]^2 K(y_i, y, x_i, x_j), \quad (65)$$

where N is the model (or regression) order. For instance, with the basis set $\phi_l(x_i, x_j) = (x_i - x_j)^l$, we have the Taylor series expansion, leading to the local polynomial approaches of classical kernel regression [20], [71], [9]. Alternatively, the basis vectors could be learned from the given image using a method such as K-SVD [72]. In matrix notation we have

$$\hat{\beta}(x_j) = \arg \min_{\beta(x_j)} [\mathbf{y} - \Phi_j \beta(x_j)]^T \mathbf{K}_j [\mathbf{y} - \Phi_j \beta(x_j)], \quad (66)$$

where

$$\Phi_j = \begin{bmatrix} \phi_0(x_1, x_j) & \phi_1(x_1, x_j) & \cdots & \phi_N(x_1, x_j) \\ \phi_0(x_2, x_j) & \phi_1(x_2, x_j) & \cdots & \phi_N(x_2, x_j) \\ \vdots & \vdots & & \vdots \\ \phi_0(x_n, x_j) & \phi_1(x_n, x_j) & \cdots & \phi_N(x_n, x_j) \end{bmatrix} \quad (67)$$

is a matrix containing the basis vectors in its columns; and where $\beta(x_j) = [\beta_0, \beta_1, \dots, \beta_N]^T(x_j)$ is the coefficient vector of the local signal representation in this basis. Meanwhile, \mathbf{K}_j is the diagonal matrix of weights as defined earlier. In order to maintain the ability to represent the ‘‘DC’’ pixel values, we can insist that the matrix Φ_j contain the vector $\mathbf{1}_n = [1, 1, \dots, 1]^T$ as one of its columns. Without loss of generality, we assume this to be the first column so that by definition $\phi_0(x_i, x_j) = 1$ for all i , and j .

Denoting the j -th row of Φ_j as $\phi_j^T = [\phi_0(x_j, x_j), \phi_1(x_j, x_j), \dots, \phi_N(x_j, x_j)]$ we have the closed-form solution

$$\hat{z}(x_j) = \phi_j^T \hat{\beta}(x_j) \quad (68)$$

$$= \underbrace{\phi_j^T (\Phi_j^T \mathbf{K}_j \Phi_j)^{-1} \Phi_j^T \mathbf{K}_j}_{\mathbf{w}_j^T} \mathbf{y} = \mathbf{w}_j^T \mathbf{y} \quad (69)$$

This again is a weighted combination of the pixels, though a rather more complicated one than the earlier formulation. Interestingly, the filter vector \mathbf{w}_j still has elements that sum to 1. This can be seen as follows. We have

$$\mathbf{w}_j^T \Phi_j = \phi_j^T (\Phi_j^T \mathbf{K}_j \Phi_j)^{-1} \Phi_j^T \mathbf{K}_j \Phi_j = \phi_j^T.$$

Writing this explicitly, we observe

$$\mathbf{w}_j^T \begin{bmatrix} 1 & \phi_1(x_1, x_j) & \cdots & \phi_N(x_1, x_j) \\ 1 & \phi_1(x_2, x_j) & \cdots & \phi_N(x_2, x_j) \\ \vdots & \vdots & & \vdots \\ 1 & \phi_1(x_n, x_j) & \cdots & \phi_N(x_n, x_j) \end{bmatrix} = [1, \phi_1(x_j, x_j), \cdots, \phi_N(x_j, x_j)]$$

Considering the inner product of \mathbf{w}_j^T with the first column of Φ_j , we have $\mathbf{w}_j^T \mathbf{1}_n = 1$ as claimed.

The use of a general basis as described above carries one disadvantage, however. Namely, since the coefficient vectors \mathbf{w}_j are influenced now not only by the choice of the kernel, but also by the choice of the basis, the elements of \mathbf{w}_j can no longer be guaranteed to be positive. While this can be useful in the context of filtering as it affords additional degrees of freedom, it does significantly complicate the analysis of the resulting algorithms. Most notably, the resulting weight matrix \mathbf{W} will no longer have strictly positive elements, and hence the powerful machinery afforded by the Perron-Frobenius theory is no longer directly applicable. As such, we must resort to a broader definition of a *generalized* stochastic matrix [73].

APPENDIX B

SYMMETRIC APPROXIMATION OF \mathbf{W} AND ITS PROPERTIES

We approximate the matrix $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$ with a doubly-stochastic (symmetric) positive definite matrix. The algorithm we use to effect this approximation is due to Sinkhorn [44], [45]. He proved that given a matrix with strictly positive elements (such as \mathbf{W}), there exist diagonal matrices $\mathbf{R} = \text{diag}(\mathbf{r})$ and $\mathbf{C} = \text{diag}(\mathbf{c})$ such that

$$\widehat{\mathbf{W}} = \mathbf{R} \mathbf{W} \mathbf{C}$$

is doubly stochastic. That is,

$$\widehat{\mathbf{W}} \mathbf{1}_n = \mathbf{1}_n \quad \text{and} \quad \mathbf{1}_n^T \widehat{\mathbf{W}} = \mathbf{1}_n^T \quad (70)$$

Furthermore, the vectors \mathbf{r} and \mathbf{c} are unique to within a scalar (i.e. $\alpha \mathbf{r}$, \mathbf{c}/α .) Sinkhorn's algorithm for obtaining \mathbf{r} and \mathbf{c} in effect involves repeated normalization of the rows and columns (See Algorithm 1 for details) so that they sum to one, and is provably convergent and optimal in the cross-entropy sense [46]. To see that the resulting matrix $\widehat{\mathbf{W}}$ is symmetric positive definite, we note a corollary of Sinkhorn's result: When a symmetric matrix is considered, the diagonal scalings are in fact identical [44], [45]. Applied to \mathbf{K} , we see that the symmetric diagonal scaling $\widehat{\mathbf{K}} = \mathbf{M} \mathbf{K} \mathbf{M}$ yields a symmetric doubly-stochastic

matrix. But we have $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$, so

$$\widehat{\mathbf{W}} = \mathbf{R}\mathbf{D}^{-1}\mathbf{K}\mathbf{C}$$

Since the $\widehat{\mathbf{W}}$ and the diagonal scalings are unique (to within a scalar), we must have $\mathbf{R}\mathbf{D}^{-1} = \mathbf{C} = \mathbf{M}$, and therefore $\widehat{\mathbf{W}} = \widehat{\mathbf{K}}$. That is to say, applying Sinkhorn's algorithm to either \mathbf{K} or its row-sum normalized version \mathbf{W} yields the very same (symmetric positive definite doubly stochastic) result, $\widehat{\mathbf{W}}$.

Algorithm 1 Algorithm for scaling a matrix \mathbf{A} to a nearby doubly-stochastic matrix $\widehat{\mathbf{A}}$

Given a matrix \mathbf{A} , let $(n, n) = \text{size}(\mathbf{A})$ and initialize $\mathbf{r} = \text{ones}(n, 1)$;

for $k = 1 : \text{iter}$;

$$\mathbf{c} = 1./(\mathbf{A}^T \mathbf{r});$$

$$\mathbf{r} = 1./(\mathbf{A} \mathbf{c});$$

end

$$\mathbf{C} = \text{diag}(\mathbf{c}); \mathbf{R} = \text{diag}(\mathbf{r});$$

$$\widehat{\mathbf{A}} = \mathbf{R} \mathbf{A} \mathbf{C}$$

The convergence of this iterative algorithm is known to be linear [45], with rate given by the subdominant eigenvalue $\widehat{\lambda}_2$ of $\widehat{\mathbf{W}}$. It is of interest to know how much the diagonal scalings will perturb the eigenvalues of \mathbf{W} . This will indicate how accurate our approximate analysis in Section V will be, which uses the eigenvalues of $\widehat{\mathbf{W}}$ instead of \mathbf{W} . Fortunately, Kahan [74], [43] provides a nice result: Suppose that \mathbf{A} is a non-Hermitian matrix with eigenvalues $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n|$. Let $\widehat{\mathbf{A}}$ be a perturbation of \mathbf{A} that is Hermitian with eigenvalues $\widehat{\lambda}_1 \geq \widehat{\lambda}_2 \geq \dots \geq \widehat{\lambda}_n$, then

$$\sum_{i=1}^n |\lambda_i - \widehat{\lambda}_i|^2 \leq 2 \|\mathbf{A} - \widehat{\mathbf{A}}\|_F^2$$

Indeed, if \mathbf{A} and $\widehat{\mathbf{A}}$ are both positive definite (as is the case for both \mathbf{K} and \mathbf{W}) the factor 2 on the right-hand side of the inequality may be discarded. Specializing the result for \mathbf{W} , defining $\mathbf{\Delta} = \mathbf{W} - \widehat{\mathbf{W}}$ and normalizing by n , we have

$$\frac{1}{n} \sum_{i=1}^n |\lambda_i - \widehat{\lambda}_i|^2 \leq \frac{1}{n} \|\mathbf{\Delta}\|_F^2$$

The right-hand side can be further bounded as described in [75].

Hence for patches of reasonable size (say $n \geq 11$), the bound on the right-hand side, and the eigenvalues of the respective matrices are quite close, as demonstrated in Figure 18. Bounds may be established on

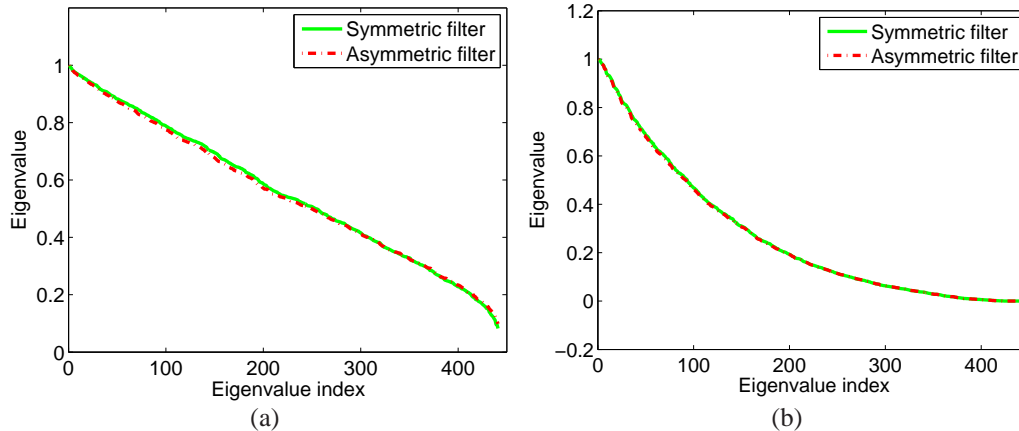


Fig. 18. Eigenvalues of row-stochastic (asymmetric) filter matrices and the corresponding doubly-stochastic (symmetric) filter matrices symmetrized using Sinkhorn's algorithm: (a) a NLM filter example; (b) a LARK filter example.

the distance between the eigenvectors of \mathbf{W} and $\widehat{\mathbf{W}}$ as well [75]. Of course, since both matrices are row-stochastic, they share their first right eigenvector $\widehat{\mathbf{v}}_1 = \mathbf{v}_1$, corresponding to a unit eigenvalue. The second eigenvectors are arguably more important, as they tell us about the dominant structure of the region being filtered, and the corresponding effect of the filters. Specifically, as indicated in [47], applying Theorem 5.2.8 from [43] gives a bound on the perturbation of the second eigenvectors:

$$\|\widehat{\mathbf{v}}_2 - \mathbf{v}_2\| \leq \frac{\|4\Delta\|_F}{\nu - \sqrt{2}\|\Delta\|_F},$$

where where ν is the gap between the second and the third eigenvalues. It is worth noting that the bound essentially does not depend on the dimension n of the data \mathbf{y} . This is encouraging, as it means that by approximating \mathbf{W} with $\widehat{\mathbf{W}}$, we get a uniformly bounded perturbation to the second eigenvector regardless of the filter window size. That is to say, the approximation is quite stable.

As a sidenote, we mention the interesting fact that, when applied to a symmetric squared distance matrix, Sinkhorn's algorithm has a rather nice geometric interpretation [76], [77]. Namely, if P is the set of points that generated the symmetric distance matrix A , then elements of its Sinkhorn scaled version $B = \mathbf{MAM}$ correspond to the square distances between the corresponding points in a set Q , where the points in Q are obtained by a *stereographic projection* of the points in P . The points in Q are confined to a hyper-sphere of dimension d embedded in \mathbb{R}^{d+1} , where d is the dimension of the subspace spanned by the points in P .

APPENDIX C

STABILITY OF FILTER COEFFICIENTS TO PERTURBATIONS DUE TO NOISE

The statistical analysis of the non-parametric filters \mathbf{W} is quite complicated when \mathbf{W} is considered random. Fortunately, however, the stability results in [47] allow us to consider the resulting filters as approximately deterministic. More specifically, we assume that the noise \mathbf{e} corrupting the data has i.i.d. samples from a symmetric (but not necessarily Gaussian) distribution, with zero-mean and finite variance σ^2 . The results in [47] imply that if the noise variance σ^2 is small relative to the clean data \mathbf{z} (i.e., when signal-to-noise ratio is high), the weight matrix $\widetilde{\mathbf{W}} = \mathbf{W}(\mathbf{y})$ computed from the noisy data $\mathbf{y} = \mathbf{z} + \mathbf{e}$ is near the latent weight matrix $\mathbf{W}(\mathbf{z})$. That is, with the number of samples n sufficiently large [47],

$$\|\widetilde{\mathbf{W}} - \mathbf{W}\|_F^2 \leq_p c_1 \sigma_e^{(2)} + c_2 \sigma_e^{(4)}, \quad (71)$$

for some constants c_1 and c_2 , where $\sigma_e^{(2)}$ and $\sigma_e^{(4)}$ denote the second and fourth order moments of $\|\mathbf{e}\|$, respectively, and “ \leq_p ” indicates that the inequality holds in probability. When the noise variance is sufficiently small, the fourth order moment is even smaller; hence, the change in the resulting coefficient matrix is bounded by a constant multiple of the small noise variance.

Approximations to the moments of the perturbation $d\mathbf{W} = \widetilde{\mathbf{W}} - \mathbf{W}$ are also given in [47], [78]:

$$\begin{aligned} \mathbb{E}[d\mathbf{W}] &\approx \mathbb{E}[d\mathbf{D}] \mathbf{D}^{-2} \mathbf{K} - \mathbf{D}^{-1} \mathbb{E}[d\mathbf{K}], \\ \mathbb{E}[d\mathbf{W}^2] &\approx \mathbb{E}[d\mathbf{D}^2] \mathbf{D}^{-4} \mathbf{K}^2 + \mathbf{D}^{-2} \mathbb{E}[d\mathbf{K}^2] - \mathbb{E}[d\mathbf{D} d\mathbf{K}] \mathbf{D}^{-3} \circ \mathbf{K}. \end{aligned}$$

where $d\mathbf{D} = \widetilde{\mathbf{D}} - \mathbf{D}$, and $d\mathbf{K} = \widetilde{\mathbf{K}} - \mathbf{K}$, and \circ denotes element-wise product. A thorough description of the perturbation analysis for the class of Gaussian data-adaptive weights is given in [47], [78]. To summarize, for a sufficiently small noise variance, the matrix of weights can be treated, with high confidence, as a deterministic quantity which depends only on the content of the underlying latent image \mathbf{z} .

APPENDIX D

EXTENSIONS TO MORE GENERAL RESTORATION PROBLEMS

Many of the ideas discussed in the context of denoising here can be extended to more general image restoration problems. As a simple example, we briefly note the case where the measurement model includes a linear distortion such as blur, expressed as follows:

$$\mathbf{y} = \mathbf{A} \mathbf{z} + \mathbf{e}. \quad (72)$$

One approach would be to employ the framework of the residual iterations defined earlier. More specifically, define

$$\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{z}}_k + \mathbf{A}^T \mathbf{W}(\mathbf{y} - \mathbf{A} \hat{\mathbf{z}}_k), \quad (73)$$

where we observe that the term $\mathbf{W}(\mathbf{y} - \mathbf{A} \hat{\mathbf{z}}_k)$ is filtering (i.e. denoising) the residuals from the measurement model in a spatially adaptive, non-parametric manner using \mathbf{W} . The operator \mathbf{A}^T is the adjoint to the blur, and applied to these filtered residuals, acts as a "backprojection" operation. The asymptotic behavior of this iterative process is interesting. Namely, as $k \rightarrow \infty$,

$$\mathbf{A}^T \mathbf{W} \mathbf{y} = \mathbf{A}^T \mathbf{W} \mathbf{A} \hat{\mathbf{z}}_\infty \quad (74)$$

or equivalently

$$\hat{\mathbf{z}}_\infty = (\mathbf{A}^T \mathbf{W} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{W} \mathbf{y}. \quad (75)$$

Without the adaptive weights in \mathbf{W} , this asymptotic solution is the familiar (*global*) least squares solution, which is unstable due to the direct inverse. The inclusion of the kernel weights, and stopping the process after a finite number of iterations can result in a stabilized solution with smaller MSE. The analysis of the stability and statistical characteristics of such estimates, along with the choice of the optimal stopping point will require in depth understanding of the spectrum of the operator $\mathbf{A}^T \mathbf{W}$, much in the same way as we did with the spectrum of \mathbf{W} in the denoising framework.

REFERENCES

- [1] D. Barash, "A fundamental relationship between bilateral filtering, adaptive smoothing and the nonlinear diffusion equation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 844–847, June 2002.
- [2] D. Barash and D. Comaniciu, "A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift," *Image Vis. Comput.*, vol. 22, no. 1, pp. 73–81, 2004.
- [3] M. Elad, "On the origin of the bilateral filter and ways to improve it," *IEEE Transactions on Image Processing*, vol. 11, no. 10, pp. 1141–1150, October 2002.
- [4] O. Scherzer and J. Weickert, "Relations between regularization and diffusion filtering," *Journal of Mathematical Imaging and Vision*, vol. 12, no. 1, pp. 43–63, Feb 2000.
- [5] A. Singer, Y. Shkolinsky, and B. Nadler, "Diffusion interpretation of nonlocal neighborhood filters for signal denoising," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 118–139, 2009.
- [6] L. Pizarro, P. Mrazek, S. Didas, S. Grewenig, and J. Weickert, "Generalised nonlocal image smoothing," *International Journal of Computer Vision*, vol. 90, no. 1, pp. 62–87, 2010.
- [7] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, vol. 4, no. 2, pp. 490–530, 2005.
- [8] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Proc. of the 1998 IEEE International Conference of Compute Vision, Bombay, India*, pp. 836–846, January 1998.

- [9] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, February 2007.
- [10] K. Dabov, A. Foi, V. Katkovnic, and K. Egiazarian, "Image denoising by sparse 3D transform-domain collaborative filtering," *IEEE Transactions of Image Processing*, vol. 16, no. 8, pp. 2080–2095, August 2007.
- [11] S. M. Smith and J. M. Brady, "Susan – a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.
- [12] H. Knutsson and C. F. Westin, "Normalized and differential convolution - methods for interpolation and filtering of incomplete and uncertain data," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 515–523, June 1993.
- [13] L. Yaroslavsky, *Digital Picture Processing*. Springer Verlag, 1987.
- [14] K. Fukunaga and L. D. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions of Information Theory*, vol. 21, pp. 32–40, 1975.
- [15] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, May 2002.
- [16] A. Smola and B. Scholkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.
- [17] A. Smola, B. Scholkopf, and K. Muller, "The connection between regularization operators and support vector kernels," *Neural Networks*, vol. 11, p. 637649, 1998.
- [18] R. Rosipal and L. J. Trejo, "Kernel partial least squares regression in reproducing kernel Hilbert space," *Journal of Machine Learning Research*, vol. 2, pp. 97–123, 2001.
- [19] F. Girosi, "An equivalence between sparse approximation and support vector machines," *Neural Computation*, vol. 10, no. 6, pp. 1455–1480, 1998.
- [20] M. P. Wand and M. C. Jones, *Kernel Smoothing*, ser. Monographs on Statistics and Applied Probability. London; New York: Chapman and Hall, 1995.
- [21] W. Hardle, *Applied Nonparametric Regression*. Cambridge [England] ; New York: Cambridge University Press, 1990.
- [22] P. Lancaster and K. Salkauskas, "Surfaces generated by moving least squares methods," *Mathematics of Computation*, vol. 87, pp. 141–158, 1981.
- [23] D. Levin, "The approximation power of moving least-squares," *Mathematics of Computation*, vol. 67, no. 224, pp. 1517–1531, 1998.
- [24] M. Alexa, J. Behr, D. Cohen-or, S. Fleishman, and D. Levin, "Computing and rendering point set surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 3–15, 2003.
- [25] R. Gonzalez and J. Woods, *Digital Image Processing, Third Edition*. Prentice Hall, 2008.
- [26] C. Kervrann and J. Boulanger, "Optimal spatial adaptation for patch-based image denoising," *Image Processing, IEEE Transactions on*, vol. 15, no. 10, pp. 2866–2878, Oct. 2006.
- [27] S. P. Awate and R. T. Whitaker, "Unsupervised, information-theoretic, adaptive image filtering for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 364–376, March 2006.
- [28] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," *CVPR*, pp. 60–65, June 2005.
- [29] S. Awate and R. Whitaker, "Higher-order image statistics for unsupervised, information-theoretic, adaptive, image filtering," *CVPR*, pp. 44–51, June 2005.

- [30] H. Takeda, S. Farsiu, and P. Milanfar, “Kernel regression for image processing and reconstruction,” *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, February 2007.
- [31] H. Knutsson and C.-F. Westin, “Normalized and differential convolution: Methods for interpolation and filtering of incomplete and uncertain data,” *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 515–523, June 16-19 1993.
- [32] N. Sochen, R. Kimmel, and A. Bruckstein, “Diffusions and confusions in signal and image processing,” *J. Math. Imaging and Vision*, vol. 14, pp. 195–209, 2001.
- [33] A. Spira, R. Kimmel, and N. Sochen, “A short time Beltrami kernel for smoothing images and manifolds,” *IEEE Trans. Image Processing*, vol. 16, no. 6, pp. 1628–1636, 2007.
- [34] J. Ham, D. Lee, S. Mika, and B. Schölkopf, “A kernel view of the dimensionality reduction of manifolds,” *Proc. of the 21st International Conference on Machine Learning*, 2004.
- [35] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” *The Annals of Statistics*, vol. 36, no. 3, pp. 1171–1220, 2008.
- [36] N. Dowson and O. Salvado, “Hashed nonlocal means for rapid image filtering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 485–499, 2011.
- [37] J. Baek and D. Jacobs, “Accelerating spatially varying Gaussian filters,” *ACM Transactions on Graphics*, vol. 29, no. 6, December 2010.
- [38] M. Belkin and P. Niyogi, “Towards a theoretical foundation for Laplacian-based manifold methods,” *Proc. of the 18th Conference on Learning Theory (COLT)*, pp. 486–500, 2005.
- [39] A. Szlam, M. Maggioni, and R. Coifman, “Regularization on graphs with function-adapted processes,” *Journal of Machine Learning Research*, vol. 9, pp. 1711–1739, 2008.
- [40] E. Seneta, *Non-negative Matrices and Markov Chains*, ser. Springer Series in Statistics. Springer, 1981.
- [41] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1991.
- [42] H. J. Landau and A. M. Odlyzko, “Bounds for eigenvalues of certain stochastic matrices,” *Linear Algebra and Its Applications*, vol. 38, pp. 5–15, June 1981.
- [43] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.
- [44] P. Knight, “The Sinkhorn-Knopp algorithm: Convergence and applications,” *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 1, pp. 261–275, 2008.
- [45] R. Sinkhorn, “A relationship between arbitrary positive matrices and doubly stochastic matrices,” *The Annals of Mathematical Statistics*, vol. 35, no. 2, pp. 876–879, 1964, 1964.
- [46] J. Darroch and D. Ratcliff, “Generalized iterative scaling for log-linear models,” *Annals of Mathematical Statistics*, vol. 43, pp. 1470–1480, 1972.
- [47] L. Huang, D. Yan, M. I. Jordan, and N. Taft, “Spectral clustering with perturbed data,” *Neural Information Processing Systems (NIPS)*, 2008.
- [48] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3-D transform-domain collaborative filtering,” *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, August 2007.
- [49] P. Chatterjee and P. Milanfar, “Is denoising dead?” *IEEE Trans. on Image Proc.*, vol. 19, no. 4, pp. 895–911, April 2010.
- [50] —, “Patch-based near-optimal denoising,” *IEEE Transactions on Image Processing*, 2011, to appear; Resources available at <http://tinyurl/plowdenoise>.

- [51] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions in Pattern Analysis and Machine Intelligence*, vol. 12, no. 9, pp. 629–639, July 1990.
- [52] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 460–489, 2005.
- [53] P. Buhlmann and B. Yu, "Boosting with the l_2 loss: Regression and classification," *Journal of the American Statistical Association*, vol. 98, no. 462, pp. 324–339, 2003.
- [54] J. W. Tukey, *Exploratory Data Analysis*. Addison Wesley, 1977.
- [55] N. Nordstrom, "Biased anisotropic diffusion - a unified regularization and diffusion approach to edge detection," *Image and Vision Computing*, vol. 8, pp. 318–327, 1990.
- [56] D. P. Palomar and S. Verdu, "Gradient of mutual information in linear vector Gaussian channels," *IEEE Trans. on Information Theory*, vol. 52, no. 1, pp. 141–154, 2006.
- [57] Y. Freund and R. E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, September 1999.
- [58] P. Buhlmann and T. Hothorn, "Boosting algorithms: regularization, prediction and model fitting," *Statistical Science*, vol. 22, no. 4, pp. 477–505, 2007.
- [59] Y. Censor and S. Zenios, *Parallel Optimization*, ser. Numerical Mathematics and Scientific Computation. New York: Oxford University Press, 1997.
- [60] M. Collins, R. Schapire, and Y. Singer, "Logistic regression, Adaboost and Bregman distances," *Machine Learning*, vol. 48, no. 1/2/3, 2002.
- [61] M. Charest and P. Milanfar, "On iterative regularization and its application," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 18, no. 3, pp. 406–411, March 2008.
- [62] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397 – 3415, Dec. 1993.
- [63] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, "Edge-preserving decompositions for multi-scale tone and detail manipulation," *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2008)*, vol. 27, no. 3, Aug. 2008.
- [64] F. Luisier, T. Blu, and M. Unser, "A new SURE approach to image denoising: Interscale orthonormal wavelet thresholding," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 593–606, March 2007.
- [65] Z. Wang and A. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, Jan 2009.
- [66] B. Carlin and T. Louis, *Bayes and Empirical Bayes Methods for Data Analysis*, ser. Texts in Statistical Science. Chapman and Hall, 2000.
- [67] T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and M. Weinberger, "Universal discrete denoising: known channel," *IEEE Trans. on Info. Theory*, vol. 51, no. 1, pp. 5–28, 2005.
- [68] S. Awate and R. Whitaker, "Feature-preserving MRI denoising: A nonparametric empirical bayes approach," *IEEE Trans. Med. Imaging*, vol. 26, no. 9, pp. 1242–1255, 2007.
- [69] K. Sivaramkrishnan and T. Weissman, "A context quantization approach to universal denoising," *IEEE Trans. on Signal Processing*, vol. 57, no. 6, pp. 2110–2129, 2009.
- [70] A. Bruckstein, "On globally optimal local modeling: From moving least square to overparametrization." SIAM, 2010, Conference on Imaging Science, Chicago, IL.
- [71] L. Wasserman, *All of Nonparametric Statistics*. Springer, 2005.

- [72] M. Aharon, M. Elad, and A. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, November 2006.
- [73] R. Khoury, "Closest matrices in the space of generalized doubly stochastic matrices," *Journal of Mathematical Analysis and Applications*, vol. 222, no. AY985970, pp. 562–568, 1998.
- [74] W. Kahan, "Spectra of nearly Hermitian matrices," *Proc. of the American Mathematical Society*, vol. 48, pp. 11–17, 1975.
- [75] P. Milanfar, "Asymptotic nearness of stochastic and doubly-stochastic matrices," *submitted*.
- [76] H. Jegou, C. Schmid, H. Harzallah, and J. Verbeek, "Accurate image search using the contextual dissimilarity measure," *IEEE Trans. on Pattern Analysis and Machine Intell.*, vol. 32, no. 1, pp. 2–11, January 2010.
- [77] C. Johnson, R. Masson, and M. Trosset, "On the diagonal scaling of Euclidean distance matrices to doubly stochastic matrices," *Linear Algebra and its Applications*, vol. 397, no. 1, pp. 253–264, 2005.
- [78] D. Yan, L. Huang, and M. I. Jordan, "Fast approximate spectral clustering," *15th ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2009.