STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **12**, November 2024, pp 1775–1794.
Published online in International Academic Press (www.IAPress.org)

Image Denoising Using the Geodesics' Gramian of the Manifold Underlying Patch-Space

Kelum Gajamannage[1,*], Randy Paffenroth[2], Anura P. Jayasumana[3]

[1]*Department of Mathematics and Applied Mathematical Sciences, University of Rhode Island, Kingston, RI-02881, USA*
[2]*Department of Mathematical Sciences, Department of Computer Science, Data Science Program, Worcester Polytechnic Institute, Worcester, MA-01609, USA*
[3]*Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO-80525, USA*

Abstract With the proliferation of sophisticated cameras in modern society, the demand for accurate and visually pleasing images is increasing. However, the quality of an image captured by a camera may be degraded by noise. Thus, some processing of images is required to filter out the noise without losing vital image features. Even though the current literature offers a variety of denoising methods, the fidelity and efficacy of their denoising are sometimes uncertain. Thus, here we propose a novel and computationally efficient image denoising method that is capable of producing accurate images. To preserve image smoothness, this method inputs patches partitioned from the image rather than pixels. Then, it performs denoising on the manifold underlying the patch-space rather than that in the image domain to better preserve the features across the whole image. We validate the performance of this method against benchmark image processing methods.

Keywords Gramian, Graph Geodesic, Manifold, Nonlocal Denoising, Patch-Base

AMS 2010 subject classifications 68U10, 94A08, 94A12, 32C09

DOI: 10.19139/soic-2310-5070-2124

1. Introduction

Modern systems such as satellites and medical imaging instruments rely on cameras operating in diverse environments to capture high-quality images of interest [26, 31]. Those images are often contaminated with significant noise during acquisition, compression, and transmission that leads to distortion and loss of image information [11]. Noisy images degrade the performance of subsequent image processing tasks such as image analysis, tracking, and even video processing. Thus, before processing images, an extra step should be performed to denoise images. Since noise, edges, and texture are high-frequency components of an image, distinguishing each component, especially noise, is a non-trivial task [11]. This limitation frequently causes loss of some vital features of the recovered image. Thus, the recovery of high-quality images without losing vital features is an essential attribute of the denoising process.

Deep learning based image denoising methods, such as [8], [43], and [44], learn a mapping function on a training set that contains clean image pairs by optimizing a loss function [12]. Recently, these methods have received surged attention as they have performed well in many computer vision tasks [12]. However, the deep learning image denoising frameworks suffer from major drawbacks, that are prominent in typical neural networks, such as difficulties in training when the noise contamination is high, vanishing gradient when the network is considerably deep, and high computational cost due to repeated training [22].

*Correspondence to: Kelum Gajamannage (Email: kelum.gajamannage@uri.edu). Department of Mathematics and Applied Mathematical Sciences, University of Rhode Island, 45 Upper College Rd., Kingston, RI-02881, USA.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Other than a few deep learning based image denoising frameworks that became existed in recent years, the image denoising literature is significantly dominated by non deep learning based image denoising frameworks. Non deep learning based image denoising methods can be categorized into two types: patch-based and pixels-based. Patch-based image denoising methods, such as [7, 28, 42, 38, 45, 39, 23, 21], partition an input image into blocks, called patches, and process these patches locally in "patch-space" to estimate the true pixel values of the original image [6]. Patch-based image denoising methods are well known for better performance in contrast to that of pixel-based methods [3]. Patch-based image denoising approaches possess noteworthy advantages such as their efficiently smooth flat regions due to overlaps between patches, and their ability to preserve fine image details and sharp edges [3]. Sparse 3-D transform-domain collaborative filtering [7], abbreviated as BM3D, is the most popular denoising method that consists of two-stage non-local collaborative filtering in the transform domain. BM3D stacks similar patches into 3D groups by block matching and then these 3D groups are transformed into the wavelet domain. Then, hard thresholding or Wiener filtering with coefficients is employed in the wavelet domain followed by applying an inverse transformation of coefficients. Finally, the denoised version of the image is constructed by aggregating all the estimated patches. However, when the noise increases gradually, the denoising performance of BM3D decreases greatly and artifacts are introduced, especially in flat areas [11].

Sparse and redundant representations over learned dictionaries [10], abbreviated as KSVD, is another popular denoising method that is based on sparse and redundant representations over trained dictionaries. Using the KSVD algorithm, a dictionary that describes the image content is effectively obtained. Since this denoising method is primarily based on the KSVD algorithm, it is named as KSVD denoising (for convenience, we simply call KSVD for this denoising method later). The training is performed either using the corrupted image itself or training on a corpus of high-quality image database. Here, the user needs to threshold the required depth of the sparsity that ensures the sparse representation uses no more than this many columns of the dictionary to reconstruct every image patch instance. Since the KSVD is limited in handling small image patches, authors extend its deployment to arbitrary image sizes by defining a global image prior in a Bayesian reconstruction framework that forces sparsity over patches in every location in the image. However, this method has computational deficiencies since both the choice of an appropriate dictionary for a dataset is a non-convex problem and the implementation of KSVD is an iterative approach that does not guarantee to find the global optimum [35]. Wavelets denoising with empirical Bayes thresholding, abbreviated as BWD (Bayes wavelet denoising), [25], has also shown good denoising performance. Since wavelets localize features of an image to different scales, important signal or image features can be preserved while removing the noise. Specifically, the wavelet transformation leads to a sparse representation for many real-world images where this representation concentrates image features in a few large magnitude wavelet coefficients. Small wavelet coefficients typically represent noise that can be shrunk or removed without affecting the image quality. Empirical Bayes approach is used as the threshold rule that is based on the assumption that the image has an independent prior distribution given by a mixture model. The inverse wavelet transform is applied on shrunk coefficients to generate the noise-free version of the image. However, such wavelet denoising approaches suffer from high computational cost, less natural denoising, and less robustness.

Here, we propose a novel denoising method that uses eigenvectors of the Gramian matrix of geodesic distances. In our method, first, we partition the noisy image into partially overlapping moving square-patches with a known length, say ρ, such that each patch is centered at one unique pixel of the image. Each patch is a point in a ρ^2-dimensional space where a low-dimensional manifold underlies [17, 18]. This manifold representation is similar to the wavelet domain representation of BM3D and BWD, and redundant dictionary representation of KSVD, where the image features are concentrated. Revealing this hidden manifold helps better explain the geometry of the patch-set and then helps identify the features in the image. Learning such a manifold is the salient step in the discipline of *Dimensionality Reduction* from where we borrow the basic concept for the proposed method for denoising images. In the context of image processing, the process of projecting the high-dimensional data of the patch-space into a low-dimensional manifold is technically the same as eliminating the noise in the image since extra dimensions mostly represent noise and minor information. This projection is conducted using the Gramian matrix of geodesic distances [20, 19]. Due to this manifold approach, our method can be considered to be a non-local denoising method that performs denoising in the patch-space rather than that in the image domain to preserve the features across the image.

Geodesics on this manifold are approximated using a custom-made graph structure over the patch-space. This graph structure is made in two steps: 1) we represent all the patches as vertices, 2) for a given neighborhood parameter δ, we search δ many nearest neighbor patches for each patch and join each pair of neighbors with an edge having the weight equal to the Euclidean distance between them. This neighborhood search constructs a graph structure on the dataset where each point is treated as a vertex. This graph structure closely mimics the geometry of the manifold hidden in the patch-space. Then, we compute the shortest path distance between each pair of vertices as an approximation to its geodesic and construct the geodesic distance matrix. Since nearby points on this manifold represent similar patches of the image regardless of their physical location in the image, geodesic distances encode similarity between patches. Thus, this graph structure approximating geodesic distances is similar to the 3D stacking of the similar patches in BM3D. The advantage of using geodesic distance over the trivial Euclidean distance as a proximity for the manifold distance is that the geodesic distance is nonlinear whereas Euclidean distance is linear. Nonlinear proximity of a manifold mimics the manifold closely so that it helps capturing the true geometry of the manifold underlying the image patch-space. Thus, the adoption of the geodesic distance over the adoption of Euclidian distance ensures better quality of the denoised images that retains essential image features.

The geodesic distance matrix is transformed into its Gramian matrix by double centering. Eigenvalues and eigenvectors of the Gramian matrix of a dataset explain the geometric structures of the dataset [17, 4]. Thus, those eigenvalues and eigenvectors are used to reduce the dimensionality of high-dimensional datasets in Dimensionality Reduction methods such as [19, 20]. Bigger the eigenvalue of the Gramian, more prominent the features presented by that eigenvalue and its eigenvector. As the noise in an image is less prominent than that of the texture or cartoon, the big eigenvalues and their eigenvectors are dominated by the texture, while the smaller ones mostly represent the noise. Thus, we utilize a subset of fewer eigenvectors of the Gramian matrix to construct the noise-free patches. Usage of fewer prominent eigenvectors of the Gramian matrix is similar to hard thresholding or Wiener filtering on the wavelet domain in BM3D, thresholding to define the sparsity on the dictionary in KSVD, and empirical Bayes thresholding on the wavelet coefficients in BWD. Finally, we merge these noise-free patches to generate the noise-free version of the original noisy image. Since our method performs the denoising task using the Gramian matrix of the shortest graph distances mimicking the manifold geodesics, we name our method as *Geodesic Gramian Denoising* and abbreviate it as GGD.

Our paper is organized as follows: First, we provide the detailed information and theory associated with the development of GGD in Sec. 2. Then, we analyze the sensitivity of GGD with respect to its parameters and compare the performance of GGD against benchmark denoising methods with respect to both the input parameters and different levels of corruption in Sec. 3. Finally, we provide a summary along with conclusions in Sec. 4. Table 1 and Table 2 state the notations and abbreviations, respectively, used in this article along with their descriptions.

1.1. Contributions

Our proposed denoising method, GGD, makes the following contributions to the literature:

- GGD is a novel denoising algorithm that leverages eigenvectors of the Gramian matrix of geodesic distances between patches. This geodesic's Gramian approach helps denoising the original noisy image with a small subset of eigenvectors.
- GGD is a non-local denoising scheme that performs denoising in the patch-space rather than that in the image domain. Patch-based methods are well known for preserving image smoothness, fine image details, and sharp edges, across the entire image than those of pixel-based methods.
- In contrast to the general non-local denoising methods that are well known to have many parameters, GGD possesses only three parameters that can easily be pre-determined.

Table 1. Nomenclature

Notation	Description
$d(k,k')$	Distance between patches $\boldsymbol{u}(\boldsymbol{x}_k)$ and $\boldsymbol{u}(\boldsymbol{x}_{k'})$
L	Eigenvector threshold such that $l = 1,\dots,L$
k	Index of the ij-th pixel such that $k = n(i-1)+j$
λ_l	l-th eigenvalue
n	Length and width of the image
δ	Nearest neighbor parameter
ρ	Patch size
Δ	Reconstruction error
$\tilde{\mathcal{U}}$	Denoised Image
$\mathcal{D}$	Geodesic distance matrix
$\mathcal{G}$	Gramian matrix
$\mathcal{I}$	Original image
I	Identity matrix
$\mathcal{U}$	Input image for the algorithm (often noisy)
Γ	Weights of Shepard's method
$\tilde{\boldsymbol{u}}(\boldsymbol{x}_{ij})$	Denoised version of the patch $\boldsymbol{u}(\boldsymbol{x}_{ij})$
V	Eigenvectors of the matrix $\mathcal{G}$ such that $V = [\boldsymbol{\nu}_1\vert\dots\vert\boldsymbol{\nu}_l\vert\dots\vert\boldsymbol{\nu}_{n^2}]^T$
Λ	Eigenvalues of the matrix $\mathcal{G}$ such that $\Lambda = diag(\lambda_1,\dots,\lambda_l,\dots,\lambda_{n^2})$
$\boldsymbol{x}_{ij}$	ij-th pixel of the image
$\boldsymbol{\nu}_l$	l-th eigenvector
$\boldsymbol{u}(\boldsymbol{x}_{ij})$	Patch centered at the point $\boldsymbol{x}_{ij}$
$G(V,E)$	Graph G with the vertex set V and edge set E
$\mathcal{N}(\boldsymbol{x}_k)$	Neighborhood at the pixel k

Table 2. Abbreviations

Notation	Description
BM3D	Sparse 3-D transform-domain collaborative filter denoising [7]
KSVD	Denoising with sparse and redundant representations over learned dictionaries [10]
BWD	Wavelet denoising with empirical Bayes thresholding [25]
GGD	Geodesic Gramian denoising
NLB	Nonlocal Bayesian image denoising [28]
AD	Anisotropic diffusion [41]
ID	Isotropic diffusion [34, 5]
RMSE	Root mean square error
PSNR	Peak signal to noise ratio
SSIM	Structural similarity index measure

2. Geodesic Gramian denoising

In this section, first, we describe the partition of an image into patches as well as the low-dimensional structure of the patch-space. Second, we state both the construction of a graph structure in the patch-space and the approximation of geodesics. Then, we provide the formulation of the Gramian matrix of the geodesic distances. We explain the technique of denoising patches next. Finally, we explain the construction of the noise-reduced version of the input image from the denoised patches.

2.1. *Patch-set*

While there are several approaches to partition a noisy image into patches, we partition the given image, denoted as $\mathcal{U}$, of size $n \times n$ into equal-sized square-shaped patches, denoted as $\boldsymbol{u}(\boldsymbol{x}_{ij})$'s; $i, j = 1, \ldots, n$, of odd length, denoted as ρ, as defined in Definition 1. We create square-shaped and odd length patches for the convenience of the formulation of GGD. As we make a patch centered at each pixel, the neighboring patches overlap each other. We replicate the boundary for the required number of times to partition a patch of $\rho \times \rho$ pixels that is centered on a pixel either at the boundary of the image or as close as $\lceil \rho/2 \rceil$ pixels to the boundary. This process creates n^2 patches for an image of size $n \times n$ that we call the patch-set. Each patch of $\rho \times \rho$ dimensions is treated as a point in a space of ρ^2-dimensions; thus, the extrinsic dimensionality of this patch-set is ρ^2.

Definition 1
Let $\boldsymbol{x}_{ij}$ be a pixel of the image with horizontal and vertical displacements i and j units, respectively, from the top-left corner of the image domain. We define a square-shaped patch of an odd length as the square-shaped block of pixels $\boldsymbol{u}(\boldsymbol{x}_{ij})$ of length ρ centered at $\boldsymbol{x}_{ij}$ given by

$$\boldsymbol{u}(\boldsymbol{x}_{ij}) = \begin{bmatrix} u_1(\boldsymbol{x}_{ij}) \\ u_2(\boldsymbol{x}_{ij}) \\ \vdots \\ u_{\rho^2}(\boldsymbol{x}_{ij}) \end{bmatrix} = \begin{bmatrix} \mathcal{U}(i - \lfloor \rho/2 \rfloor, j - \lfloor \rho/2 \rfloor) \\ \mathcal{U}(i - \lfloor \rho/2 \rfloor + 1, j - \lfloor \rho/2 \rfloor) \\ \vdots \\ \mathcal{U}(i + \lfloor \rho/2 \rfloor, j + \lfloor \rho/2 \rfloor) \end{bmatrix}. \tag{1}$$

For simplicity, sometimes we write $\boldsymbol{u}(\boldsymbol{x}_k)$ for $\boldsymbol{u}(\boldsymbol{x}_{ij})$ where $k = n(i-1) + j$ and $1 \le k \le n^2$.

The space that the path-set represents is known as the high-dimensional input space in the field of *Dimensionality Reduction* where the geometry of the patch-set is governed by an underlying low-dimensional manifold structure. Dimensionality reduction methods project this high-dimensional data onto this low-dimensional manifold in order to learn the prominent features in the dataset [17, 16]. The extra dimensions that the high-dimensional data possesses than that of the manifold are mostly the non-prominent features including noise. We use this concept of dimensionality reduction to eliminate noise in the patch-set. For that, first, we define geodesic distance matrix in Sec. 2.2

2.2. *Geodesic distance matrix*

Geodesics are defined as the true distances on the manifold. However, the computation of such true manifold distances is infeasible due to finite sampling. Thus, we approximate the geodesic distances as the shortest paths on a graph structure that we create in the patch-set. Specifically, we run the neighborhood search algorithm given in [2] with a user input neighborhood parameter, defined as δ, on the patch-set. This algorithm searches δ nearest neighbors for each patch, representing a ρ^2-dimensional point, using Euclidean distance. Then, we create a graph structure $G(V, E)$ on this dataset by defining the points, $\{\boldsymbol{u}(\boldsymbol{x}_k) | k = 1, \ldots, n^2\}$, as vertices, V. We define the edge set, E, by joining each pair of nearest neighbor points, say $\boldsymbol{u}(\boldsymbol{x}_k)$ and $\boldsymbol{u}(\boldsymbol{x}_{k'})$, with an edge having the weight equal to the Euclidean distance, denoted as $d(k, k')$,

$$d(k, k') = \|\boldsymbol{u}(\boldsymbol{x}_k) - \boldsymbol{u}(\boldsymbol{x}_{k'})\|_2, \tag{2}$$

between them.

The geodesic distance between two points in the dataset or two patches in the patch-set is approximated as the *shortest path distance* between the corresponding two vertices in the graph $G(V, E)$. The shortest path distance between any two vertices can be computed in many ways, including Dijkstra's algorithm [9]. However, we employ Floyd's algorithm [13] for this task as it computes the shortest paths between all the pairs of vertices in one batch, and is more efficient than Dijkstra's algorithm in this case. We approximate all the geodesic distances between patches and formulate the geodesic distance matrix $\mathcal{D} \in \mathbb{R}^{n^2 \times n^2}_{\ge 0}$ of the patch-set. Then, we transform this geodesic distance matrix into its Gramian matrix in Sec. 2.3.

2.3. Gramian matrix

We transform the geodesic distance matrix $\mathcal{D} \in \mathbb{R}_{\geq 0}^{n^2 \times n^2}$ into its Gramian matrix, denoted by $\mathcal{G}_{n^2 \times n^2}$, using

$$\mathcal{G}[i,j] = -\frac{1}{2}\big[\mathcal{D}[i,j] - \mu_i(\mathcal{D}) - \mu_j(\mathcal{D}) + \mu_{ij}(\mathcal{D})\big], \tag{3}$$

where $\mu_i(\mathcal{D})$, $\mu_j(\mathcal{D})$, and $\mu_{ij}(\mathcal{D})$ are the means of the i-th row of the matrix $\mathcal{D}$, j-th column of that matrix, and the mean of the full matrix, respectively, [30]. Gramian matrix of geodesic distances is *real-valued*, *symmetric*, and *positive semi-definite*, see Definition 2 [30].

The eigenvalues and eigenvectors of the Gramian matrix can be used to describe the properties of the underlying manifold of the patch-set and then that manifold can be used to describe the features of the image. While big eigenvalues and their corresponding eigenvectors of the Gramian matrix represent prominent features of the image including edges and corners of the texture, small eigenvalues and eigenvectors of that represent non-prominent features such as fine image details and noise. This statement is supported by the observations in Fig 1 where we analyze the information retention of the eigenvalues and eigenvectors of the Gramian matrix. Therein, we produce the Gramian matrix of an image of size 100×100 as described in Secs. 2.1, 2.3, and 2.2. Fig. 1(a) shows that only a few eigenvectors, out of 100^2, of the Gramian matrix is sufficient to reconstruct the image to a good level of quality. Figs. 1(b) and 1(c) show that only a few eigenvalues of the Gramian matrix retain most of the information of the image, while most of them retain less information. Consider that the denoising of patches that we will present in Sec. 2.4 was not performed for this example; however, the technical details of the image reconstruction using eigenvectors will be provided in Sec. 2.5.

Definition 2
Let $\mathcal{G}$ be a square-shaped matrix of order $n^2 \times n^2$. The eigenvalue decomposition of $\mathcal{G}$ is

$$\mathcal{G} = V\Lambda V^T, \tag{4}$$

where $V = [\boldsymbol{\nu}_1|\dots|\boldsymbol{\nu}_l|\dots|\boldsymbol{\nu}_{n^2}]^T$ is a matrix that represents eigenvectors $\boldsymbol{\nu}_l$'s by its rows and $\Lambda = diag(\lambda_1, \dots, \lambda_l, \dots, \lambda_{n^2})$ is a diagonal matrix that represents eigenvalues λ_l's. The matrix $\mathcal{G}$ is positive semi-definite, if and only if $\lambda_l \geq 0$ for all l.

2.4. Denoising patches

The patches are denoised using only a few, say L (eigenvector threshold), prominent eigenvectors of the Gramian matrix as they represent essential features of the image. For $l = 1, \dots, n^2$ and $\lambda_1 \geq \dots \geq \lambda_l \geq \dots \lambda_{n^2}$, $(\lambda_l, \boldsymbol{\nu}_l)$ represents eigenvalue and eigenvector pairs of the Gramian matrix. We denote the noise-reduced version of the patch $\boldsymbol{u}_k$ as $\tilde{\boldsymbol{u}}_k$ that we produce by

$$\tilde{\boldsymbol{u}}(\boldsymbol{x}_k) = \sum_{l=1}^{L} \langle \boldsymbol{u}(\boldsymbol{x}_k), \boldsymbol{\nu}_l \rangle \boldsymbol{\nu}_l, \tag{5}$$

where $l = 1, \dots, L$. Here, $\langle \cdot, \cdot \rangle$ denotes the inner product according to Definition 3. Note that this k is related to row index i and the column index j, both measured from the top-left corner of the image, by $k = n(i-1) + j$. Denoised patches are merged using Shepard's method as stated in Sec. 2.5.

Definition 3
Let $\boldsymbol{\nu}^{(1)} = \left(\nu_1^{(1)}, \dots, \nu_l^{(1)}, \dots, \nu_{n^2}^{(1)}\right)$ and $\boldsymbol{\nu}^{(2)} = \left(\nu_1^{(2)}, \dots, \nu_l^{(2)}, \dots, \nu_{n^2}^{(2)}\right)$ be two vectors, the inner product of these vectors is defined as

$$\langle \boldsymbol{\nu}^{(1)}, \boldsymbol{\nu}^{(2)} \rangle = \sum_{l=1}^{n^2} \nu_l^{(1)} \nu_l^{(2)}. \tag{6}$$

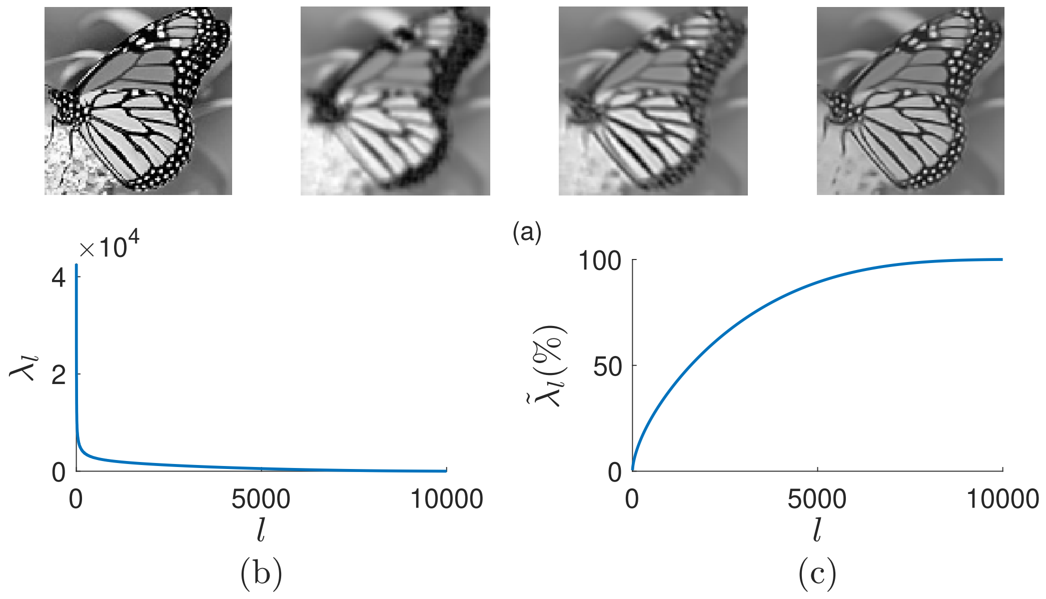

Figure 1. Information retention of eigenvalues and eigenvectors of the Gramian matrix constructed for an image of size 100×100. First, we construct the patch-set of 100^2 patches, each with the length $\rho = 5$, from the leftmost image in (a). Second, we create a graph structure in the patch-space with the neighborhood size $\delta = 10$ and approximate the geodesic distances using Floyd's algorithm. Third, we formulate the Gramian matrix of the geodesic distances. We reconstruct the leftmost image in (a) using the eigenvector associated with the biggest eigenvalue, see the second image from the left of (a), using the eigenvectors associated with the five biggest eigenvalues, see the third image from the left of (a), and using the eigenvectors associate with the 100 biggest eigenvalues, see the rightmost image in (a). (b) The magnitude of eigenvalues, denoted by λ_l for $l = 1, \dots, n^2$, from the biggest to the smallest, versus eigenvalue index, denoted by l, of the Gramian matrix and, (c) cumulative percentage of eigenvalues, denoted by $\tilde{\lambda}_l$ for $l = 1, \dots, n^2$, versus the index.

2.5. *Merging denoised patches*

In our approach, each pixel in the image domain is overlapped with ρ^2 patches. This overlapping makes each pixel location in the image also exist in nearby ρ^2 patches. These nearby pixels are within the radius of $\rho/2$ units from the target pixel, say $\boldsymbol{x}_k$. We denote this neighborhood as $\mathcal{N}(\boldsymbol{x}_k)$ and define as

$$\mathcal{N}(\boldsymbol{x}_k) = \{\boldsymbol{x}_t \mid \ \|\boldsymbol{x}_k - \boldsymbol{x}_t\|_\infty \leq \rho/2\}, \tag{7}$$

[33].

In order to reconstruct the image intensity at a pixel location of the noise-reduced version of the image, we have to get an estimate from the same pixel location in ρ^2 nearby patches. For each pixel $\boldsymbol{x}_t \in \mathcal{N}(\boldsymbol{x}_k)$, there exists a new index t_n such that the extrinsic pixel location (i, j) at that new index of the patch $\tilde{\boldsymbol{u}}(\boldsymbol{x}_t)$, denoted by $[\tilde{\boldsymbol{u}}(\boldsymbol{x}_t)]_{t_n}$, is the same as the extrinsic pixel location of $\boldsymbol{x}_k$. We use this new index in the patch merging step below. We combine all these estimates using a moving least square approximation given by Shepard's method [37] and construct the pixel $\boldsymbol{x}_k$ of the denoised version of the image as

$$\tilde{\mathcal{U}}(\boldsymbol{x}_k) = \sum_{\boldsymbol{x}_t \in \mathcal{N}(\boldsymbol{x}_k)} \Gamma(\boldsymbol{x}_k, \boldsymbol{x}_t)[\tilde{\boldsymbol{u}}(\boldsymbol{x}_t)]_{t_n}, \tag{8}$$

where the weights $\Gamma(\boldsymbol{x}_k, \boldsymbol{x}_t)$ are defined as

$$\Gamma(\boldsymbol{x}_k, \boldsymbol{x}_t) = \frac{e^{-\|\boldsymbol{x}_k - \boldsymbol{x}_t\|^2}}{\sum_{\boldsymbol{x}_{t'} \in \mathcal{N}(\boldsymbol{x}_k)} e^{-\|\boldsymbol{x}_k - \boldsymbol{x}_{t'}\|^2}}. \tag{9}$$

The weighting term in Eqn. (9) weights close by pixels with more weight while the faraway pixels with less weight. Thus, according to Eqn. (8), merging assures that the pixel $\boldsymbol{x}_k$ of the reconstructed image is highly influenced by the pixels at the same location of the nearby patches. The main steps of GGD are summarized in Algorithm 1.

Algorithm 1 *Geodesic Gramian Denoising (GGD).*

```
Inputs:   noisy image ($\mathcal{U}_{n\times n}$), patch length ($\rho$), nearest neighborhood size ($\delta$),
          and eigenvector threshold ($L$).
Outputs:  noise-reduced image ($\tilde{\mathcal{U}}_{n\times n}$).

1: Construct $n^2$ overlapping square-shaped patches each with the length $\rho$ from the noisy image $\mathcal{U}_{n\times n}$ and denote
   the patch-set as $\{\boldsymbol{u}(\boldsymbol{x}_k) | k = 1, \ldots, n^2\}$, (Sec. 2.1).
2: Produce the graph structure $G(V, E)$ from the patch-set using the nearest neighbor search algorithm in [2].
   Use Floyd's algorithm in [13], to approximate the geodesic distances in the patch-space and then produce the
   geodesic distance matrix $\mathcal{D}$, (Sec. 2.2).
3: Construct the Gramian matrix $\mathcal{G}$ from the geodesic distance matrix $\mathcal{D}$ using Eqn. (3), (Sec. 2.3).
4: Compute the eigenvectors $\{\nu_l | l = 1, \ldots L\}$ corresponding to the $L$ biggest eigenvalues of the Gramian matrix
   $\mathcal{G}$ and use Eqn. (5) to produce noise-free patches $\{\tilde{\boldsymbol{u}}(\boldsymbol{x}_k) | k = 1, \ldots, n^2\}$, (Sec. 2.4).
5: Merge noise-free patches using Eqns. (8) and (9), and generate the denoise image $\tilde{\mathcal{U}}_{n\times n}$, (Sec. 2.5)
```

Computational complexity of GGD is dominated by Step 4 of Algorithm 1 where it uses the eigenvalue decomposition to generate eigenvectors of the Gramian matrix. For an image of $n \times n$ pixels, Step 1 of the algorithm generates a patch-space of size n^2. Then, while steps 2 measures geodesic distances between patches to creates a distance matrix of size $n^2 \times n^2$, step 3 converts this distance matrix to a Gramian matrix of the same size. Since the computational complexity of the eigenvalue decomposition of a matrix of size $m \times m$ is $\mathcal{O}(m^3)$ [29], the computational complexity of the eigenvector computation of the Gramian matrix is $\mathcal{O}(n^6)$. Thus, the computational complexity of GGD is $\mathcal{O}(n^6)$ for denoising an image of $n \times n$. The Python implementation of this algorithm is available in Ref. [15].

3. Performance analysis

We analyze the performance of GGD by both visual perception and three similarity metrics, namely, Root Mean Square Error (RMSE), Peak Signal to Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM). First, we analyze the sensitivity of GGD to both the noise contamination (ϵ) of an input image and the parameters of GGD, namely, patch size (ρ), neighborhood size (δ), and eigenvector threshold (L). Then, we compare the performance of GGD with six benchmark image denoising methods.

RMSE, see Def. 4, ranges between 0 and ∞, is a primary measure to detect the numerical trad-off of an image from a reference image, e.g, while the original noise-free image is the reference image, the other image is a denoised approximation of a noisy version of the original image [24]. PSNR, see Def. 5, measures the numerical difference of an images from a reference image, with respect to the maximum possible pixel value of the reference image [24] where PSNR ranges between 0 and ∞. If the original noise-free image is the reference image and a denoised approximation of its noisy version is the other image, a higher PSNR value provides better quality of the approximation since it approaches infinity as the RMSE approaches zero [24]. SSIM, see Def. 6, ranging between -1 and 1 measures the structural similarity of an image to a reference image [40]. Ideal approximation provides 1 for SSIM since such approximation ensures $\mu_{\mathcal{I}} = \mu_{\tilde{\mathcal{U}}}$, $\sigma_{\mathcal{I}} = \sigma_{\tilde{\mathcal{U}}}$, and $\sigma_{\mathcal{I}\tilde{\mathcal{U}}} = \sigma_{\mathcal{I}}\sigma_{\tilde{\mathcal{U}}}$ in Def. 6. Instead of using traditional error summation methods, SSIM is designed by modeling any image distortion as a combination of three factors, namely, luminance distortion, contrast distortion, and loss of correlation.

Definition 4
Let, two-dimensional matrix $\mathcal{I}$ represents a reference image of size $n \times n$ and $\tilde{\mathcal{U}}$ represents any other image of interest. Root Mean Square Error [24], abbreviated as RMSE, of the image $\tilde{\mathcal{U}}$ with respect to the reference image $\mathcal{I}$ is defined as

$$RMSE(\mathcal{I},\tilde{\mathcal{U}}) = \sqrt{\frac{\sum_{(i,j)\in\mathbb{N}^{n\times n}}(\mathcal{I}[i,j]-\tilde{\mathcal{U}}[i,j])^2}{n^2}}. \tag{10}$$

Definition 5
Let, two-dimensional matrix $\mathcal{I}$ represents a reference image of size $n \times n$ and $\tilde{\mathcal{U}}$ represents any other image of interest. Peak Signal to Noise Ratio [24], abbreviated as PSNR, of the image $\tilde{\mathcal{U}}$ with respect to the reference image $\mathcal{I}$ is defined as

$$PSNR(\mathcal{I},\tilde{\mathcal{U}}) = 20\log_{10}\left(\frac{\max(\mathcal{I})}{RMSE(\mathcal{I},\tilde{\mathcal{U}})}\right). \tag{11}$$

Here, $\max(\mathcal{I})$ represents the maximum possible pixel value of the image $\mathcal{I}$. Since the pixels in our images of interest are represented in 8-bit digits, $\max(\mathcal{I})$ is 255.

Definition 6
Let, two-dimensional matrix $\mathcal{I}$ represents a reference image of size $n \times n$ and $\tilde{\mathcal{U}}$ represents an image of interest. Structural Similarity Index Measure [40], abbreviated as SSIM, of the image $\tilde{\mathcal{U}}$ with respect to the reference image $\mathcal{I}$ is defined as the product of luminance distortion (I), contrast distortion (C), and loss of correlation (S), such as

$$SSIM(\mathcal{I},\tilde{\mathcal{U}}) = I(\mathcal{I},\tilde{\mathcal{U}})\ C(\mathcal{I},\tilde{\mathcal{U}})\ S(\mathcal{I},\tilde{\mathcal{U}}), \tag{12}$$

where

$$\begin{aligned} I(\mathcal{I},\tilde{\mathcal{U}}) &= \frac{2\mu_{\mathcal{I}}\mu_{\tilde{\mathcal{U}}}+c_1}{\mu_{\mathcal{I}}^2+\mu_{\tilde{\mathcal{U}}}^2+c_1},\\ C(\mathcal{I},\tilde{\mathcal{U}}) &= \frac{2\sigma_{\mathcal{I}}\sigma_{\tilde{\mathcal{U}}}+c_2}{\sigma_{\mathcal{I}}^2+\sigma_{\tilde{\mathcal{U}}}^2+c_2},\\ S(\mathcal{I},\tilde{\mathcal{U}}) &= \frac{\sigma_{\mathcal{I}\tilde{\mathcal{U}}}+c_3}{\sigma_{\mathcal{I}}\sigma_{\tilde{\mathcal{U}}}+c_3}. \end{aligned} \tag{13}$$

Here, $\mu_{\mathcal{I}}$ and $\mu_{\tilde{\mathcal{U}}}$ are means of $\mathcal{I}$ and $\tilde{\mathcal{U}}$, respectively; $\sigma_{\mathcal{I}}$ and $\sigma_{\tilde{\mathcal{U}}}$ are standard deviations of $\mathcal{I}$ and $\tilde{\mathcal{U}}$, respectively; and $\sigma_{\mathcal{I}\tilde{\mathcal{U}}}$ is the covariance between $\mathcal{I}$ and $\tilde{\mathcal{U}}$. Moreover, c_1, c_2, and c_3 are very small positive constants to avoid the case of division by zero.

3.1. Sensitivity analysis

Here, first, we analyze the sensitivity of GGD with respective to the parameters patch size (ρ) and neighborhood size (δ). Then, we analyze the sensitivity of GGD with respect to the corruption level (ϵ) of the input images followed by analyzing the sensitivity of GGD with respect to the parameter eigenvector threshold (L).

3.1.1. Patch size and neighborhood size We use an image of Lena Forsén, which is commonly used for evaluating image processing algorithms, to analyze the influence of patch size and neighborhood size for the performance. Since the original image is colored, we transform it into a gray image by taking the average across three color channels. The original noise-free image, that we denote by $\mathcal{I}$, is of the size 100×100. We make three noisy versions, denoted by $\mathcal{U}_{1,2,3}$, of this gray image by imposing additive Gaussian noise with three different levels of standard deviations, $\epsilon =$, 40, 60, and 80, such that

$$\mathcal{U} = \mathcal{I} + \mathcal{N}(0,\epsilon^2), \tag{14}$$

where $\mathcal{N}(0, \epsilon^2)$ denotes a Gaussian random distribution with mean 0 and standard deviation ϵ. Since $\mathcal{U}$'s represent images, the values of the pixels in them should be between 0–255. Thus, we adjust $\mathcal{U}$ by replacing the values less than zero with zeros and the values more than 255 with 255's.

We run our denoising method 36 times with 36 pairs of parameter values (ρ, δ) where $\rho =$ 3, 5, 7, 9, 11, 13 and $\delta =$ 5, 10, 15, 20, 25, 30 on each of the three noised versions of the image. We set the eigenvector threshold arbitrarily as $L = 50$ in GGD and compute the noise-reduced version $\tilde{\mathcal{U}}$ of the input image. We run this experiment four more times each with a different random seed (each random seed contributes one sample realization) of the Gaussian distribution in Eqn. (14) to avoid the random effect. For each pair of parameters and for each sample realization, the reconstruction error is computed using the three similarity metrics, RMSE (Def. 4), PSNR (Def. 5), and SSIM (Def. 6) with the noise-free original image as the reference image and the denoised image as the image of interest. For each parameter pair, one value for each metric is computed after averaging the five values obtained for different sample realizations. The corruption levels of the input noisy images are also assessed in terms of the same similarity metrics with the noise-free original image as the reference image and the noisy images as the images of interest. For each corruption level and for each metric, we average the metric values obtained across five different sample realizations and compute a single value.

The corruption levels of the input noisy images are $RMSE = 38.76$, $PSNR = 16.36$, and $SSIM = .3537$ for $\epsilon = 40$; $RMSE = 53.87$, $PSNR = 13.50$, and $SSIM = .2444$ for $\epsilon = 60$; $RMSE = 66.82$, $PSNR = 11.63$, and $SSIM = .1820$ for $\epsilon = 80$. Figs. 2(a-i) show that GGD performs better in denoising the image for all the 36 parameter pairs and all the three noise levels than the corresponding input corruption level. Moreover, we observe that the denoising performance can be made substantially better with some parameter values. We observe that the reconstruction error increases when the noise increases. We also observe in Figs. 2(a-i) that when the corruption levels are $\epsilon = 40$, 60, and 80, the best denoise performances, in terms of all the three metrics, are observed at $\rho = 9, 11$, $\rho = 11, 13$, and $\rho \geq 13$, respectively. Figs. 2(j-l) provides visual evidence for the above trade-offs between performance and parameters where we see that the best denoising is attained around aforesaid ρ's for aforesaid noise levels. This is because if the input image possesses high noise, big patches help smooth the image more, whereas bigger patches might smooth the image too much so that underlying image features might also be distorted. The denoising performance slightly decreases, in terms of all the metrics, when the neighborhood size increases. The reason for that is big neighborhood sizes add more edges into the graph structure $G(V, E)$ presented in Sec. 2.2 which may cause underestimation of the true geodesic distances on the manifold.

3.1.2. Corruption level of the input image Here, we analyze the performance of GGD with respect to the corruption level of the input image. We vary the corruption level of images by imposing different levels of Gaussian random noise into the image of Lena Forsén of size 100×100. For that, we vary $\epsilon = 0, 10, 20, \ldots, 100$ in Eqn. (14) and create 10 noisy images. We run GGD over these noisy images with arbitrary parameter values (ρ, δ, L) where $\delta = 10$, $\rho = 3, 7$, and $L = 10$, 10^2, 10^3, 10^4, and denoise them. We compute RMSE (Def. 4), PSNR (Def. 5), and SSIM (Def. 6) for each denoised image with the noise-free original image as the reference image and the denoised image as the image of interest. Moreover, we compute the corruption level of each noisy image using the same three metrics, with the noise-free original image as the reference image and the noisy images as the images of interest, and use that to compare the denoising results of GGD. Since we added Gaussian random noise onto images, to eliminate the random effect on the results, we run the same experiment 10 times, e.i. realizations, each with a different seed (each seed contributes one sample realization) in the Gaussian random number generator. We average the results over the 10 realizations for each combination of parameters.

Fig. 3(a-c) represent, 1) means of the three reconstruction error metrics RMSE, PSNR, and SSIM of the denoised images across 10 realizations with respect to the noise levels (ϵ's) for each combination of the parameters (ρ, L) where $\rho =$ 3,7 and $L = 10, 10^2, 10^3, 10^4$; and 2) the means of the same three metric values of the noisy images across 10 realizations with respect to the noise levels. The errorbars in the figures represent the standard deviation of the reconstruction errors across 10 realizations. We observe that the errorbars of the noisy images are bigger than that of the denoised images associated with small eigenvector thresholds due to the fact that small eigenvector thresholds remove more noise that will eventually reduce the influence of the random seed to the performance.

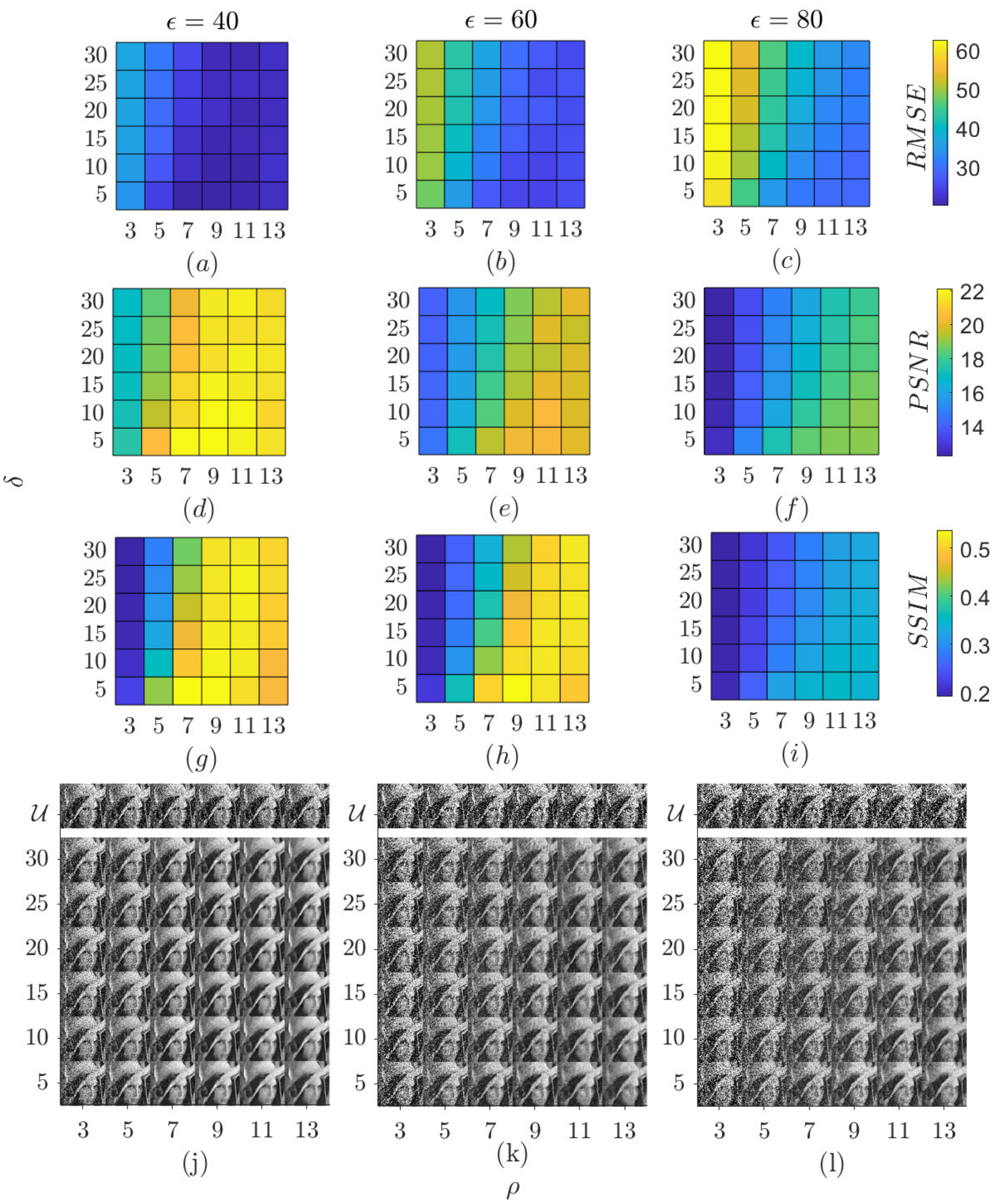

Figure 2. Denoising performance of GGD with respect to patch size (ρ) and neighborhood size (δ). We produce three noisy images from an image of Lena Forsén of size 100×100 by imposing three levels of noise sampled from the Gaussian distributions $\mathcal{N}(0, \epsilon^2)$ where $\epsilon = 40$, 60, and 80. We set the eigenvector threshold to an arbitrary 50 and run GGD with 36 pairs of parameter values $\{(\rho, \delta)|\ \rho = 3, 5, 7, 9, 11, 13; \delta = 5, 10, 15, 20, 25, 30\}$ on each of the three images. Then, we compute the reconstruction errors in terms of RMSE, PSNR, and SSIM of GGD associated with 36 parameter pairs and three noise levels $\epsilon = 40, 60, 80$. We run the same experiment four more times each with a different seed (each contributes a sample realization) in the Gaussian distribution that generates additive noise for the corrupted. For each metric, (a-c) RMSE, (d-f) PSNR, and (g-i) SSIM, and for each noise level $\epsilon = 40, 60$, and 80, we average the corresponding performance values over the five sample realizations. Each color-bar is scaled to interpret the values in all the three cases $\epsilon = 40, 60$, and 80 in the same row. We also compute the corruption level of the three noisy images using the same metric where $RMSE = 38.76$, $PSNR = 16.36$, and $SSIM = .3537$ for $\epsilon = 40$; $RMSE = 53.87$, $PSNR = 13.50$, and $SSIM = .2444$ for $\epsilon = 60$; $RMSE = 66.82$, $PSNR = 11.63$, and $SSIM = .1820$ for $\epsilon = 80$. (j-l) The appearance of the noisy images and their denoised images for all the 3×36 cases.

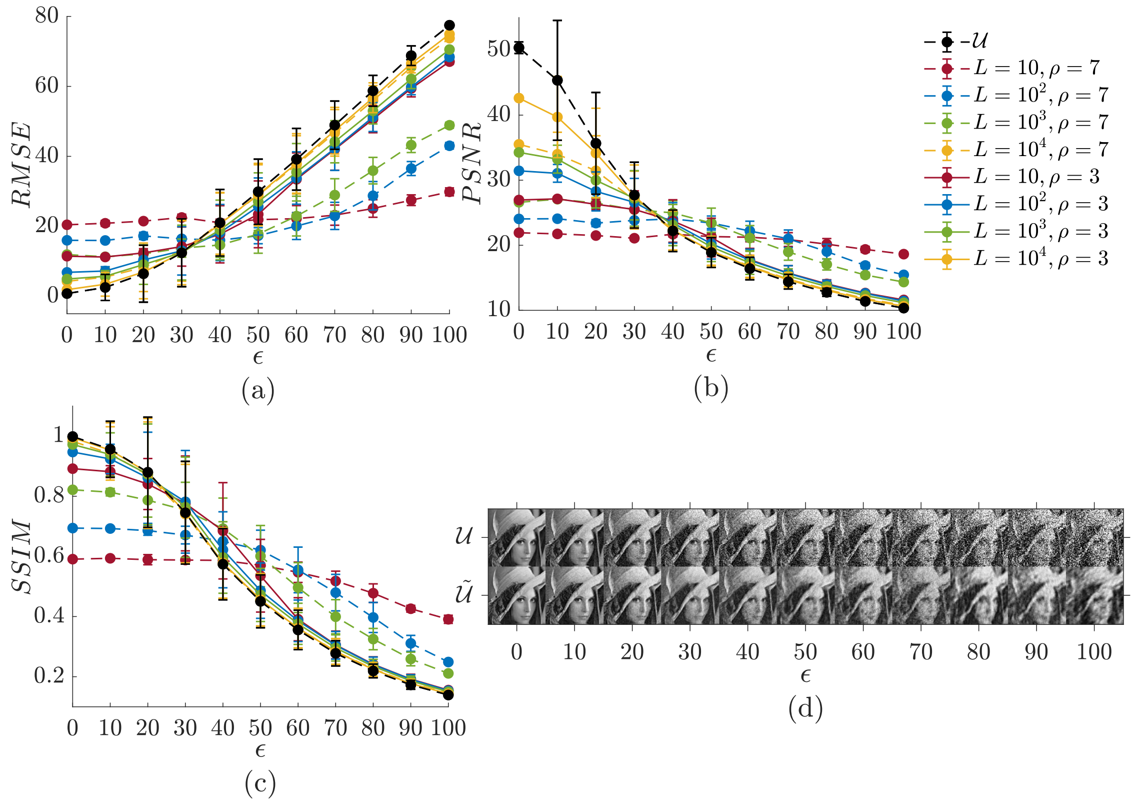

Figure 3. Image denoising performance of GGD with respect to the intensity of noise contamination (ϵ). We produce 11 images by imposing a variable noise into an image of Lena Forsén of size 100×100 that is sampled from the Gaussian distributions $\mathcal{N}(0, \epsilon^2)$ where $\epsilon = 0, 10, \ldots, 100$. We denoise these images using GGD with the parameters (ρ, δ, L) for $\rho = 3, 7$; $\delta = 10$; and $L = 10, 10^2, 10^3, 10^4$ where δ, ρ, and L denote neighborhood size, patch size, and eigenvector threshold, respectively. We compute the reconstruction errors using the three metrics RMSE, PSNR, and SSIM for all the denoised versions of the noisy images. We run GGD with each of the parameter combinations nine more times with different random seeds (each random seed provides a sample realization) in the Gaussian distribution. For each parameter combination, the means of the reconstruction errors across 10 sample realizations with respect to ϵ is presented in (a-c) where errorbars represent the standard deviations of the reconstruction errors computed across 10 sample realizations. The corruption levels of the input images, denoted by $\mathcal{U}$, with respect to the noise levels is also computed using the three metrics. We use the corruption levels to compare the denoising results of GGD. (d) The appearance of the noisy images and their denoised images for all the 11 noisy images.

Fig. 3(a-c) shows that GGD performs substantially better than the initial noisy image under all the noise levels for at least one of the combination of parameters (ρ, L). Moreover, we observe that GGD performs better with bigger patch sizes for higher noise levels whereas it performs better with smaller patch sizes for lower noise levels. This is because bigger patches have more overlapping pixels between nearby patches which makes a high dependency between points on the underlying manifold where it is capable of removing more noise in an image. We also observe that while smaller eigenvector thresholds increase denoising performance under higher noise levels, bigger eigenvector thresholds increase denoising performance under lower noise levels. More eigenvectors retain more features in the denoising of the images with low noise so that more eigenvectors increase denoising performance under low noise levels; however, more eigenvectors also retain more noise if the noise contamination is significant in the noisy images that will subsequently reduce the denoising performance. We also observe that the reconstruction errors of the denoising converge to that of the initial noisy image when all the eigenvectors, 10^4, are used for the denoising due to the fact that all the eigenvalues essentially produces back the initial noisy image.

3.1.3. Eigenvector threshold Here, we analyze the influence of the eigenvector threshold on the performance of GGD. We create three images of Lena Forsén of size 100×100 using Eqn. (14) with three corruption levels ϵ = 40, 60, and 80. We run GGD over each image with four different parameter sets (δ, ρ) = (10,5), (10,9), (20,5), and (20,9) that we chose arbitrarily to generate the denoising at a sequence of eigenvector thresholds such that $L = 50, 100, \ldots, 10000$. The reconstruction errors of all the denoised images are computed using three reconstruction performance metrics RMSE (Def. 4), PSNR (Def. 5), and SSIM (Def. 6) by treating the noise-free original image as the reference image and the denoised image as the image of interest. Figs. 4 (a-i) show the reconstruction errors computed using three performance metrics RMSE, PSNR, and SSIM for all the 12 experiments. We compute the corruption levels of the input noisy images using the same three performance metrics and compare GGD's denoising performance with them.

We observe that at some eigenvector thresholds the denoising performance with respect to all the three reconstruction performance metrics is substantially better than the initial corruption for all the three noise levels and all the four parameter combinations. We also observe that the reconstruction error converges to the corruption level when the eigenvector threshold reaches to its maximum due to the fact that GGD with all the eigenvectors includes not only all the features of the initial noisy image but also all its noise. Independent to the eigenvector threshold, bigger patch sizes improve denoising performance with respect to all the three metrics when the noise contamination is high. However, only for some eigenvector thresholds, smaller patch sizes improve denoising performance with respect to all the three metrics when the noise contamination is low. Neighborhood size doesn't have much influence on denoising performance. The best eigenvector threshold is immensely sensitive to the noise contamination of the image and parameter values in use. We observe that GGD performs significantly better denoising with respect to all the three performance metrics with fewer eigenvector thresholds. Moreover, we observe that the optimum eigenvector thresholds for the best denoising with respect to three performance metrics are close enough. These fewer optimum eigenvectors can be computed using some algebraic techniques rather than working on the entire data matrix that we will discuss with details in Sec. 4.

3.2. Comparison of GGD with benchmark image denoising methods

After the detailed sensitivity analysis of the parameters in GGD and the analysis of noise contamination of images, as presented in Sec. 3.1; here, we compare the performance of GGD with six benchmark denoising methods. Those methods are sparse 3-D transform-domain collaborative filtering (BM3D) [7], sparse and redundant representations over learned dictionaries (KSVD) [10], wavelets denoising with empirical Bayes thresholding (BWD) [25], nonlocal Bayesian image denoising (NLB) [28], anisotropic diffusion (AD) [41], isotropic diffusion (ID) [34, 5]. We run GGD along with the above six denoising methods on five famous test images, namely, Barbara, boat, cameramen, clown, and mandrill, of size 150 $\times$ 150 that are downloaded from [1]. We produce two versions of each test image by imposing two corruption levels ϵ = 40 and 80 using Eqn. (14). We run GGD over all the 10 test images with arbitrary parameter values $\delta = 20$, $\rho = 7$, and $L = 50$, and compute the reconstruction performance using the metrics RMSE (Def. 4), PSNR (Def. 5), and SSIM (Def. 6) by treating the noise-free original image as the reference image and the denoised image as the image of interest.

We set the parameters of the other six denoising methods to the recommended values in their literature or to the default values if nothing is recommended. The parameters denoising strength, patch size, sliding step size, the maximum number of similar blocks, radius for search block matching, step between two search locations, 2D thresholding, 3D thresholding, and threshold for the block-distance of BM3D are set to ϵ (the standard deviation of the imposed noise in GGD), 12, 4, 16, 39, 1, 2, 2.8, and 3000, respectively. Block size, dictionary size, number of training iterations, Lagrangian multiplier, noise gain, and number of non-zero coefficients of KSVD are set to 64, 244, 10, 30/ϵ, 1.55, and 2, respectively. The wavelet decomposition level of BWD is set to 3. Patch size, number of similar patches, search window size, truncated rank, collaborative filtering coefficient, and the minimum threshold for similar patches of NLB are set to 8, 7, 40, 1.05, 1, and 4, respectively. The standard deviation of the edge-stopping function, amount of diffusion, and maximum iterations of AD to 1.5, 2, and 300, respectively. The standard deviation of the edge-stopping function, amount of diffusion, and maximum iterations of ID are set to 5, 5, and 1000, respectively. We run the above six methods over these 10 test images with the above parameter values and compute the reconstruction errors using the same three performance metrics RMSE, PSNR, and SSIM

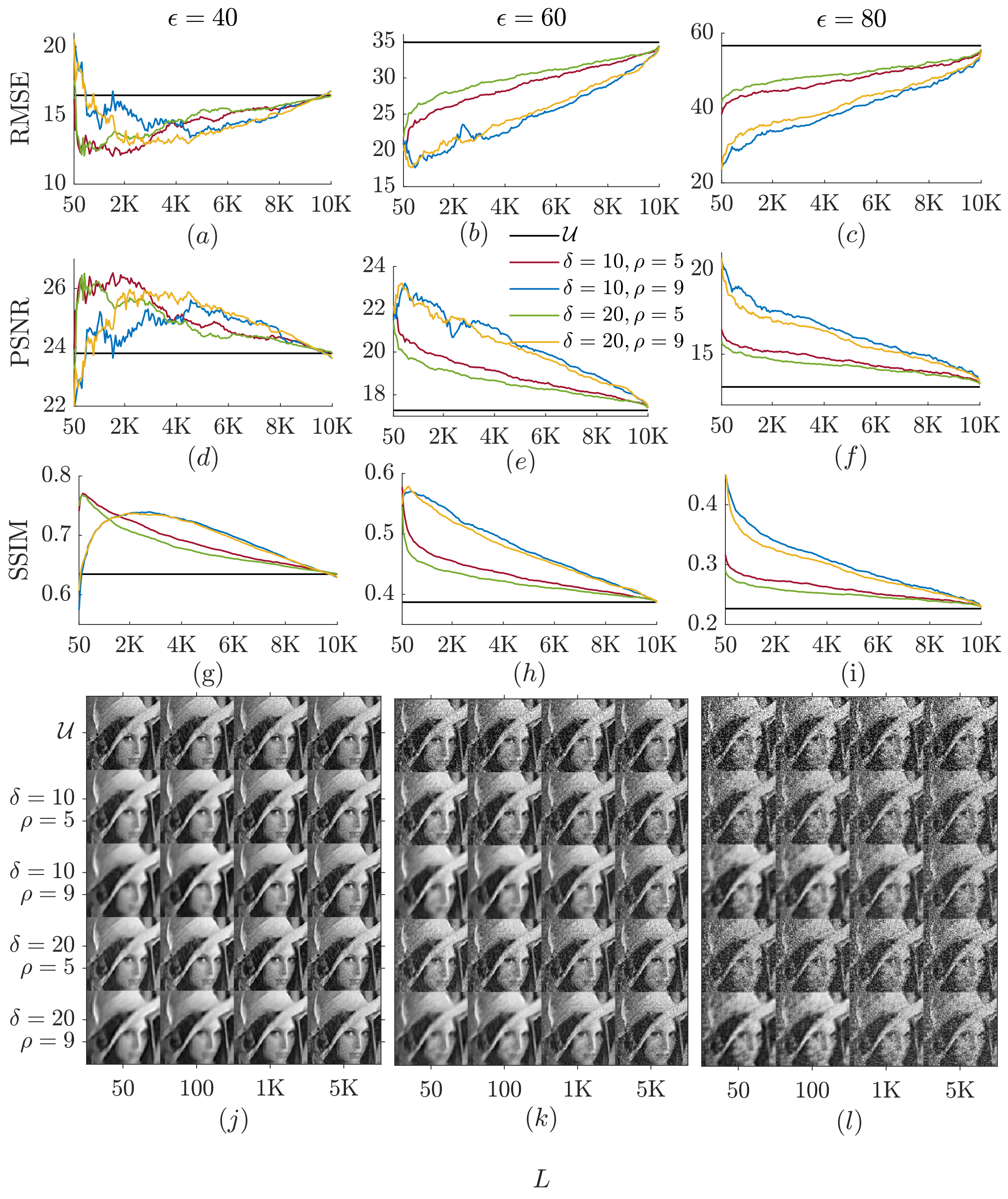

Figure 4. Denoising performance of GGD with respect to different eigenvector thresholds (L's). We impose random noise sampled from the Gaussian distributions $\mathcal{N}(0, \epsilon^2)$, where $\epsilon =$ 40, 60, and 80, into the image "Lena Forsén" of size 100×100, and generate three noisy test images. Each noisy image is denoised using GGD with four combinations of the parameters neighborhood size (δ) and patch size (ρ) such that (δ, ρ) = (10,5), (10,9), (20,5), and (20,9) for a sequence of eigenvector thresholds $L = 50, 100, \ldots, 10000$ (10000 is the total number of eigenvectors of an image of size 100×100). The reconstruction error of the denoising is computed using three performance metrics RMSE, PSNR, and SSIM that we present in (a-i). Consider that K's in these plots stand for thousand (e.i, $'$000). The corruption levels of the three test images, denoted by $\mathcal{U}$'s, are also computed using the same three performance metrics and used them to compare the denoising performance of GGD. In (a-i), even though the amount of corruption computed using any of the performance metrics for any noisy images is both a scalar and independent of the eigenvector threshold parameter, we represent it as a horizontal line at the metric value across eigenvector thresholds to provide a better comparison. (j-l) The appearance of the noisy images and their denoised images for three noise levels, four parameter combinations, and four eigenvector thresholds.

Table 3. Comparison of the denoising performance, quantified using the metrics RMSE (the first row of each block), PSNR (the second row of each block), and SSIM (the third row of each block), of GGD and other six benchmark denoising methods, namely, sparse 3-D transform-domain collaborative filtering (BM3D), sparse and redundant representations over learned dictionaries (KSVD), Bayes wavelet denoising (BWD), nonlocal Bayesian image denoising (NLB), anisotropic diffusion (AD), and isotropic diffusion (ID). For each of the five test images, namely, cameraman, mandrill, Barbara, boat, and clown, of size 150 $\times$ 150, two noisy image instances are made by imposing them with two Gaussian noise distributions of mean zero and standard deviations (ϵ's) 40 and 80. The corruption levels of the input noisy images, computed using the same three metrics are denoted by $\mathcal{U}$. While GGD is executed with the parameter values $\delta = 20$, $\rho = 7$, and $L = 50$, the other methods are executed using their recommended or default parameter values. For a given noisy image and a given performance metric, the colors red, blue, and green indicate the method performs the best, the second-best, and the third-best, respectively. We observed that the overall denoising performance of GGD is the best, that of BM3D is the second-best, and that of KSVD is the third-best.

Meth.	Images with $\epsilon = 40$					Images with $\epsilon = 80$				
	Bar.	Boa.	Cam.	Clo.	Man.	Bar.	Boa.	Cam.	Clo.	Man.
$\mathcal{U}$	40.37	44.18	38.83	36.15	41.46	69.71	72.46	67.58	62.68	70.78
	16.01	15.23	16.35	16.97	15.78	11.27	10.93	11.53	12.19	11.13
	.3444	.3067	.2533	.3676	.3443	.1627	.1466	.1359	.1978	.1628
GGD	17.11	25.70	18.18	18.75	21.94	25.60	27.26	28.18	27.28	28.87
	23.47	20.87	23.13	22.94	21.56	19.89	19.42	19.13	19.41	18.85
	.7209	.6070	.6313	.6603	.5448	.5436	.4578	.4776	.4958	.3883
BM3D	18.79	25.87	14.36	18.01	22.33	30.42	32.57	23.89	28.43	29.22
	22.65	19.87	24.99	23.02	21.15	18.47	17.88	20.57	19.06	18.78
	.7195	.6365	.7531	.6594	.4852	.5043	.4829	.6294	.4847	.3370
KSVD	18.88	26.06	17.09	18.83	21.97	27.96	33.17	25.91	27.93	29.54
	22.61	19.81	23.48	22.63	21.29	19.20	17.71	19.86	19.21	18.72
	.7008	.6053	.6088	.6348	.5347	.5251	.4299	.4131	.4789	.3336
BWD	21.91	28.53	19.59	21.64	24.67	30.22	35.09	27.92	31.28	30.05
	21.32	19.02	22.29	21.43	20.29	18.52	17.23	19.21	18.23	18.57
	.6373	.5691	.6438	.5871	.4041	.4987	.4491	.5287	.4381	.3292
NLB	22.62	29.56	21.07	20.41	25.74	36.30	41.12	34.16	33.38	38.00
	21.04	18.72	21.66	21.94	19.92	16.93	15.85	17.46	17.66	16.53
	.5647	.4696	.4032	.5527	.5005	.3449	.2822	.2533	.3630	.3007
AD	21.67	27.91	19.72	22.21	22.36	29.08	33.65	26.68	30.12	28.90
	21.42	19.22	22.23	21.19	21.14	18.86	17.59	19.61	18.55	18.84
	.6535	.5807	.5859	.5895	.5217	.4967	.4278	.4064	.4470	.3849
ID	24.27	29.53	19.63	23.54	24.47	29.87	33.95	25.88	30.37	29.03
	20.43	18.73	22.27	20.70	20.36	18.63	17.51	19.87	18.48	18.80
	.6204	.5698	.6884	.5738	.4222	.5189	.4538	.5134	.4770	.3856

by treating the noise-free original image as the reference image and the denoised image as the image of interest. We also compute the level of corruption of each test image using the same three performance metrics error metric and use them to compare the denoising results between methods.

Table 3 presents the corruption levels, computed using the metrics RMSE, PSNR, and SSIM, of the 10 noisy test images, see the row of $\mathcal{U}$, and the reconstruction errors, computed using the same metrics, of the images denoised by the eight methods including GGD. Therein, for a given noisy image and a given performance metric, the colors red, blue, and green represent the denoising method performing the best, the second-best, and the third-best performance, respectively. We observe that GGD obtains 20 reds, 4 blues, and 3 greens; BM3D obtains 10 reds, 8 blues, and 5 greens; and KSVD obtains 11 blues and 13 greens. This observation evidences that the overall denoising performance of GGD is the best, that of BM3D is the second-best, and that of KSVD is the third-best.

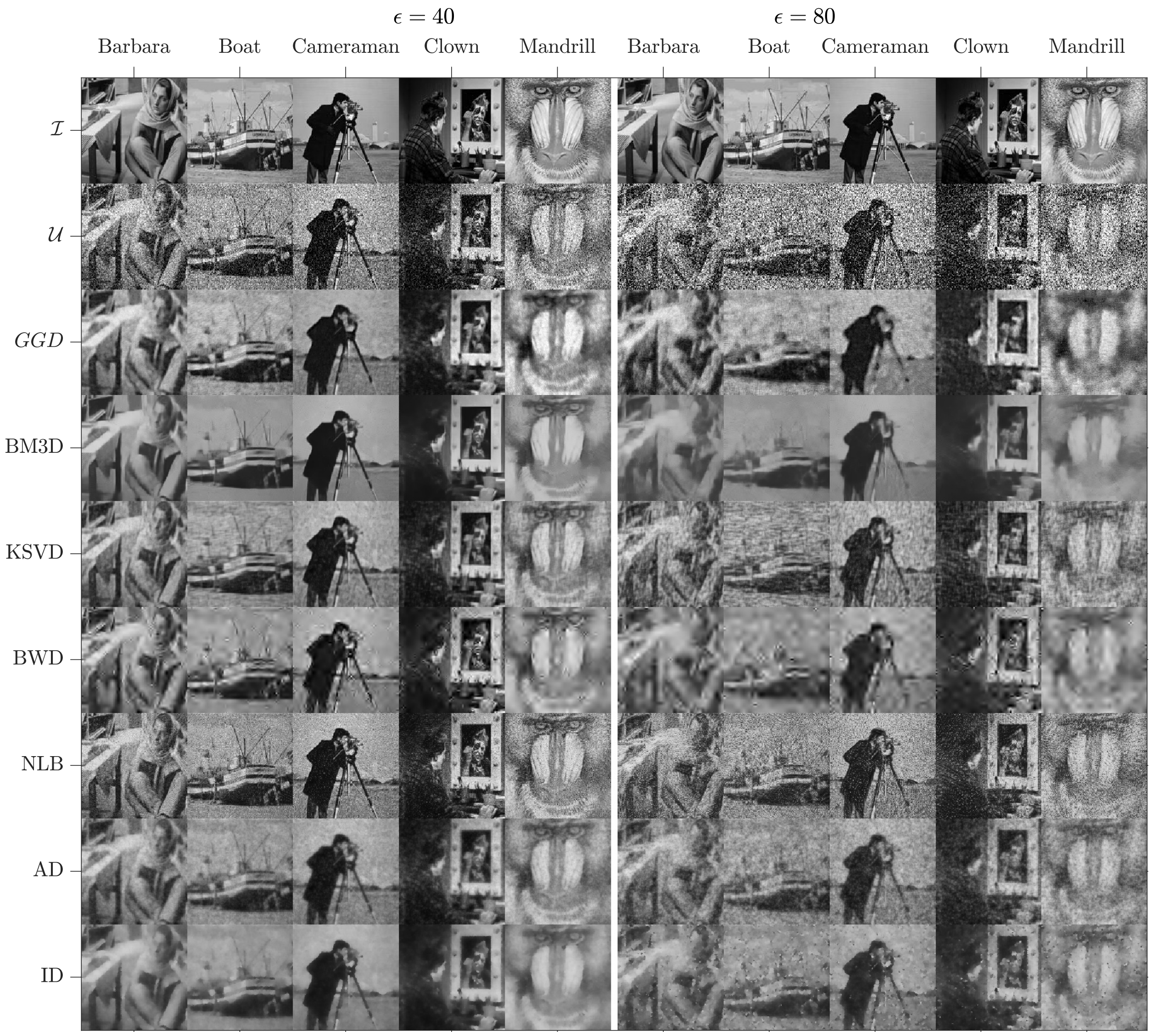

Figure 5. Visual comparison of the quality of the images denoised by GGD with that of the six other benchmark denoising methods, namely, sparse 3-D transform-domain collaborative filtering (BM3D), sparse and redundant representations over learned dictionaries (KSVD), Bayes wavelet denoising (BWD), nonlocal Bayesian image denoising (NLB), anisotropic diffusion (AD), and isotropic diffusion (ID). Each of the five test images, namely, cameraman, mandrill, Barbara, boat, and clown, of size 150×150, is imposed with two Gaussian noise distributions with a mean of 0 and standard deviations (ϵ's) of 40 and 80. Here, the original noise-free images are denoted by $\mathcal{I}$'s and their noisy versions are denoted by $\mathcal{U}$'s. While GGD is executed with the parameter values $\delta = 20$, $\rho = 7$, and $L = 50$, the other methods are executed with their recommended or default parameter values. Here, we observe that GGD retains texture and cartoon in the denoised images most of the time than that of the other six methods.

Fig. 5 visualizes the quality of the denoising of the 10 test images where we observe that GGD performs better most of the time than all of the other six methods in terms of preserving both the texture and cartoon of the original images.

4. Conclusion

In this paper, we introduced a novel image denoising method that utilizes eigenvectors of the Gramian matrix of the graph geodesics evaluated on the nosy image's patch-space. Specifically, we partitioned a given noisy image into overlapping square-shaped patches with a known length, say ρ, where each patch is a point in a high-dimensional space of ρ^2-dimensions. A low-dimensional manifold underlies this high-dimensional data cloud of the patch-set characterizes features of the noisy image. This manifold was formulated using the eigenvectors corresponding to the biggest eigenvalues of the Gramian matrix of the graph geodesic distances measured on the nosy image's patch-space. Specifically, we produced a graph structure by treating the high-dimensional patches as vertices and by joining nearest neighbor vertices to each vertex with edges of Euclidean distances between them. The geodesic distance between two patches is estimated as the graph shortest path between them. Then, we transformed the geodesic distance matrix into its Gramian matrix. The prominent eigenvectors of the Gramian matrix were used to produce the noise-free patches and these noise-free patches were merged to generate the denoised image. The reason for the adoption of geodesic distance over Euclidean distance as a proximity for the manifold distance is that the geodesic distance is nonlinear whereas the Euclidean distance is linear. Nonlinear proximity of a manifold mimics the manifold closely so that it helps capturing the true geometry of the manifold underlying the image patch-space; thus, it ensures better quality of the denoised images that retains essential image features.

We observed that when the noise contamination increases the patch size should be increased to obtain better denoising performance. This is because, if the image possesses a high noise, a big patch size helps smooth the image more; however, too big patch size might smooth the image too much. The neighborhood size doesn't have a significant influence on the denoising performance except for the case where the patch size is small. The denoising performance decreases slightly when the neighborhood size increases, especially for significantly small patch sizes. The reason for that is big neighborhood sizes add more edges into the graph structure and that causes short-circuiting of the network when the geodesics are approximated. These underestimated geodesic distances on the manifold lead to less denoising performance.

Patch size, neighborhood size, and eigenvector threshold are the only three user input parameters in GGD whereas most of the similar patch-based non-local denoising methods, such as BM3D, KSVD, and NLB, are well known to have many user input parameters. We have seen that the parameter neighborhood size of GGD is less influential for the performance that the user may set to a common value for all the experiments. For an image with high noise, our recommendation is to use a slightly bigger neighborhood size (e.g. 20) and a slightly bigger patch size (e.g. 9), and a slightly smaller eigenvector threshold (e.g. 10) in contrast to that for an image with low noise. Methods with a variety of highly influential parameters are inconvenient to use since either the user has to set a trial and error procedure to search on the entire parameter domain to find the best values for the parameters or has to make careful guesses for their values. While making a personal parameter guess is highly subjective and depends on personal experience, a trial and error procedure to search the entire parameter domain consumes a significant time and requires more computational power.

We ran the same experiment for several realizations with different random seeds in the probability distribution that we sample the noise if the experiment is to test the influence of noise contamination on denoising performance. The errorbars in the figures represent the standard deviation of the reconstruction performance across the realizations. For each performance metric, we observed that the errorbars of the plots corresponding to the denoised images associated with small eigenvector thresholds are smaller than that of the initial noisy images. Smaller errorbars infer the stability of the denoising method across realizations. Due to the fact that small eigenvector thresholds remove more noise from the denoising, that will also reduce the influence of the choice of the random seed on the performance. This independent aspect across different noise samples is essential for denoising methods since the noise in images is natural so the user doesn't know how the noise is sampled from the underlying probability distribution.

We validated the performance of GGD against six benchmark denoising algorithms, namely, sparse 3-D transform-domain collaborative filtering (BM3D), sparse and redundant representations over learned dictionaries (KSVD), Bayes wavelet denoising (BWD), nonlocal Bayesian image denoising, anisotropic diffusion, Bayesian estimation denoising, and isotropic diffusion. GGD preserves both the texture, containing edges and corners, and the cartoon, containing piece-wise smooth parts, of the original images to high accuracy than that of the other six methods. Specifically, GGD performs better than the renowned methods such as BM3D and KSVD, and than the commercially implemented method BWD that is available in MATLAB. For all the methods, we observe that the denoising performance associated with the test image boat is the lowest than that of the other four test images due to the fact that this image possesses more texture than that of the other four test images.

Our comparative study in this paper excludes deep learning denoising methods and only based on algebraic methods since GGD is purely an algebraic method made to attain unique advantages over deep learning denoising methods. In contrast to the fact that deep learning based image denoising methods have received surged attention recently, such frameworks suffer from major drawbacks including difficulties in training, especially, when the noise contamination is high; vanishing gradient when the network is considerably deep; and high computational cost due to repeated training [22]. Training a deep learning denoising framework requires extensive samples of heterogeneous data. Generating such data requires extended time, trained labour, and money. Moreover, training a deep learning denoising model requires special computational resources including powerful GPU and CPU. Our GGD framework encounters minimal of the aforementioned issues as it is being an algebraic framework.

GGD generates $\rho \times \rho$ overlapping patches at each pixel so that it produces n^2 patches for an image of size $n \times n$. These patches represent ρ^2-dimensional space in which a low-dimensional smooth manifold underlies. Revealing this smooth manifold is the baseline for denoising. The smoothness of this manifold is guaranteed by the overlapping patches as from any patch to its adjacent patch shares many common pixels so that the two patches are not spatially far apart in the patch space. Conversely, the boundary is replicated $0 - \rho$ times to ensure that a patch of size $\rho \times \rho$ is sampled at each pixel near the boundary. This replication creates blurred boundaries of the denoised images in which big ρ intensifies the issue. As big ρ offers smoothness of the denoising, it is recommended to use an appropriate value for ρ that guarantees both denoising smoothness and less blurring of the boundary.

GGD uses eigenvalue decomposition to generate eigenvectors of the Gramian matrix where this matrix is $n^2 \times n^2$ for an image of $n \times n$ pixels. Since the computational complexity of the eigenvalue decomposition of a matrix of size $m \times m$ is $\mathcal{O}(m^3)$ [29], the computational complexity of the eigenvectors of this Gramian matrix is significantly expensive as $\mathcal{O}(n^6)$. Thus, to overcome this issue, computation of only the required fewer number of eigenvectors of this Gramian matrix is worthwhile. GGD always performs better denoising with small eigenvector thresholds independent of the nature of the input image or the other parameters in-use. Since we currently compute the entire eigenvector spectrum and use only the prominent ones of them in GGD, in the future, we will improve GGD by incorporating eigenvector estimation strategies that are available in the literature of *Bigdata*. For that, we are planning to replace the regular eigenvalue decomposition routing with multiple eigenvalue approximation strategies such as, 1) inverse-free preconditioned *Krylov subspace method* called *Augmented Lanczos Bidiagonalization* [32]; 2) *random sampling* method that trains a neural network of rows of the matrix [27]; 3) *Monte Carlo* low-rank approach that iteratively makes approximations [14]; and 4) randomized singular value decomposition [36]. This future work will reduce the computational complexity of GGD significantly so that the method can be brought into a stage where it can be integrated into real-time image denoising applications.

REFERENCES

1. Matlab/C/Python/Shell programming and image/video processing/compression.
2. P. K. Agarwal and J. Erickson. Geometric range searching and its relatives. *Advances in Discrete and Computational Geometry*, 223:1–56, 1997.
3. M. H. Alkinani and M. R. El-Sakka. Patch-based models and algorithms for image denoising: a comparative review between patch-based images denoising methods for additive noise reduction, dec 2017.
4. N. Bahadur, R. Paffenroth, and K. Gajamannage. Dimension Estimation of Equity Markets. In *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pages 5491–5498. Institute of Electrical and Electronics Engineers Inc., dec 2019.
5. R. Bernardes, C. Maduro, P. Serranho, A. Araújo, S. Barbeiro, and J. Cunha-Vaz. Improved adaptive complex diffusion despeckling filter. *Optics Express*, 18(23):24048, nov 2010.
6. P. Chatterjee and P. Milanfar. Patch-based near-optimal image denoising. *IEEE Transactions on Image Processing*, 21(4):1635–1649, apr 2012.
7. K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, aug 2007.

8. S. K. Devalla, P. K. Renukanand, B. K. Sreedhar, G. Subramanian, L. Zhang, S. Perera, J.-M. Mari, K. S. Chin, T. A. Tun, N. G. Strouthidis, T. Aung, A. H. Thiéry, and M. J. A. Girard. DRUNET: a dilated-residual U-Net deep learning network to segment optic nerve head tissues in optical coherence tomography images. *Biomedical Optics Express*, 9(7):3244, jul 2018.
9. E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.
10. M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, dec 2006.
11. L. Fan, F. Zhang, H. Fan, and C. Zhang. Brief review of image denoising techniques. *Visual Computing for Industry, Biomedicine, and Art*, 2(1):1–12, dec 2019.
12. L. Fan, F. Zhang, H. Fan, and C. Zhang. Brief review of image denoising techniques, dec 2019.
13. R. W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.
14. S. Friedland, A. Niknejad, M. Kaveh, and H. Zare. Fast Monte-Carlo low rank approximations for matrices. In *Proceedings 2006 IEEE/SMC International Conference on System of Systems Engineering*, volume 2006, pages 218–223, 2006.
15. K. Gajamannage. Geodesic Gramian denoising. https://github.com/kelumdi/Geodesic_Gramian_Denoising, 2024.
16. K. Gajamannage and E. M. Bollt. Detecting phase transitions in collective behavior using manifold's curvature. *Mathematical Biosciences and Engineering*, 14(2):437 – 453, 2016.
17. K. Gajamannage, S. Butail, M. Porfiri, and E. M. Bollt. Dimensionality reduction of collective motion by principal manifolds. *Physica D: Nonlinear Phenomena*, 291:62–73, jan 2015.
18. K. Gajamannage, S. Butail, M. Porfiri, and E. M. Bollt. Identifying manifolds underlying group motion in Vicsek agents. *European Physical Journal: Special Topics*, 224(17-18):3245–3256, 2015.
19. K. Gajamannage and R. Paffenroth. Bounded manifold completion. *Pattern Recognition*, 111:107661, dec 2021.
20. K. Gajamannage, R. Paffenroth, and E. M. Bollt. A nonlinear dimensionality reduction framework using smooth geodesics. *Pattern Recognition*, 87:226–236, mar 2019.
21. K. Gajamannage, Y. Park, M. Muddamallappa, and S. Mathur. Efficient noise filtration of images by low-rank singular vector approximations of geodesics' gramian matrix. *arXiv preprint arXiv:2209.13094*, 2022.
22. K. Gajamannage, Y. Park, R. Paffenroth, and A. P. Jayasumana. Reconstruction of fragmented trajectories of collective motion using Hadamard deep autoencoders. *Pattern Recognition*, 131:108891, nov 2022.
23. K. Gajamannage, Y. Park, and A. Sadovski. Geodesic gramian denoising applied to the images contaminated with noise sampled from diverse probability distributions. *IET Image Processing*, 17(1):144–156, 2023.
24. A. Horé and D. Ziou. Image quality metrics: PSNR vs. SSIM. In *Proceedings - International Conference on Pattern Recognition*, pages 2366–2369, hore2010, 2010.
25. I. M. Johnstone and B. W. Silverman. Needles and straw in haystacks: Empirical BAYES estimates of possibly sparse sequences. *Annals of Statistics*, 32(4):1594–1649, aug 2004.
26. K. E. Joyce, S. E. Belliss, S. V. Samsonov, S. J. McNeill, and P. J. Glassey. A review of the status of satellite remote sensing and image processing techniques for mapping natural hazards and disasters. *Progress in Physical Geography*, 33(2):183–207, apr 2009.
27. M. Kobayashi, G. Dupret, O. King, and H. Samukawa. Estimation of singular values of very large matrices using random sampling. *Computers and Mathematics with Applications*, 42(10-11):1331–1352, 2001.
28. M. Lebrun, A. Buades, and J. M. Morel. A nonlocal Bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, 6(3):1665–1688, sep 2013.
29. J. Lee, V. Balakrishnan, C. K. Koh, and D. Jiao. From O(k2N) to O(N): A fast complex-valued eigenvalue solver for large-scale on-chip interconnect analysis. In *IEEE MTT-S International Microwave Symposium Digest*, pages 181–184, 2009.
30. J. A. Lee, A. Lendasse, and M. Verleysen. Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. *Neurocomputing*, 57(1-4):49–76, 2004.
31. T. M. Lehmann, C. Gönner, and K. Spitzer. Survey: Interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, 18(11):1049–1075, 1999.
32. Q. Liang and Q. Ye. Computing singular values of large matrices with an inverse-free preconditioned Krylov subspace method. *Electronic Transactions on Numerical Analysis*, 42:197–221, dec 2014.
33. F. G. Meyer and X. Shen. Perturbation of the eigenvectors of the graph Laplacian: Application to image denoising, mar 2014.
34. P. Perona and J. Malik. Scale-Space and Edge Detection Using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.
35. R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.
36. Y. Saad. *Numerical methods for large eigenvalue problems*. SIAM, 2011.
37. D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference, ACM 1968*, pages 517–524, New York, New York, USA, jan 1968. ACM Press.
38. A. D. Szlam, M. Maggioni, and R. R. Coifman. Regularization on graphs with function-adapted diffusion processes. *Journal of Machine Learning Research*, 9(Aug):1711–1739, 2008.
39. K. M. Taylor and F. G. Meyer. A random walk on image patches. *SIAM Journal on Imaging Sciences*, 5(2):688–725, jun 2012.
40. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, apr 2004.
41. J. Weickert. *Anisotropic diffusion in image processing*. Teubner Stuttgart, 1998.
42. R. Yan, L. Shao, and Y. Liu. Nonlocal hierarchical dictionary learning using wavelets for image denoising. *IEEE Transactions on Image Processing*, 22(12):4689–4698, 2013.
43. K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, jul 2017.
44. K. Zhang, W. Zuo, and L. Zhang. FFDNet: Toward a fast and flexible solution for CNN-Based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, sep 2018.

45. M. Zontak and M. Irani. Internal statistics of a single natural image. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 977–984. IEEE Computer Society, 2011.