Edge Structure Preserving 2-D and 3-D Image Denoising by Jump Surface Estimation

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Partha Sarathi Mukherjee

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Peihua Qiu, Adviser

August 2011

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude, and appreciation to my adviser Prof. Peihua Qiu, for his guidance, encouragement, and support to complete my PhD research. During a course on smoothing techniques and image processing, that I took in my first year of PhD program, he introduced me to some of the research areas on the application of statistics in image processing. Since then, he spent an enormous amount of time in training me, so that I can do well in research. I also want to express my appreciation to the other members of my committee, Prof. Douglas M. Hawkins, Prof. Hui Zou, and Prof. Sudipto Banerjee for their thought-provoking comments, encouragement, and support.

Also, I would like to thank the School of Statistics for providing me an opportunity to carry out and complete my PhD research.

ABSTRACT

Image denoising is often used for pre-processing images so that subsequent image analyses are more reliable. Many existing methods can not preserve complicated edge-structures well, but those structures contain useful information about the image objects. So, besides noise removal, a good denoising method should preserve important edge-structures. The major goal of this dissertation is to develop image denoising techniques so that complicated edge-structures are preserved efficiently. The developed methods are based on nonparametric estimation of discontinuous surfaces, because a monochrome image can be regarded as a surface of the image intensity function and its discontinuities are usually at the outlines of the objects. The first part of this dissertation introduces some existing methods and related literature. Next, an edge-structure preserving 2-D image denoising technique is proposed, and it is shown that it performs well in many applications. The next part considers 3-D images. Because of emerging popularity of 3-D MRI images, 3-D image denoising becomes an important research area. The edge-surfaces in 3-D images can have much more complicated structures, compared to the edge-curves in 2-D images. So, direct generalizations of 2-D methods would not be sufficient. This part handles the challenging task of mathematically describing different possible structures of the edge-surfaces in 3-D images. The proposed procedures are shown to outperform many popular methods. The next part deals with the well-known bias issue in denoising MRI images that is corrupted with rician noise, and provides an efficient method to remove that bias. The final part of this dissertation discusses the future research directions along the line of previous parts. One of them is image denoising by appropriate multilevel local smoothing techniques so that the fine details of the images are well preserved.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Images are used in almost every disciplines of science and engineering. Many of them contain noise that makes the observers' job difficult to study the image objects. There are many reasons behind the presence of noise in most of the images, including the artifacts of the image acquisition devices, e.g., cameras. The noise in the images should be removed to improve human interpretation. For example, noise in the Magnetic Resonance Images (MRI) of human brain makes the doctors' job difficult to detect tumor or any other abnormal growth. So, image denoising is an important research area in our society.

A gray-scale image can be expressed by a bivariate function $f(x, y)$, where (x, y) denotes the spatial location in the image and the function value $f(x, y)$ is proportional to the brightness of the image at (x, y). $f(x, y)$ is often called *image intensity function* in the literature.

To be handled by computer softwares, an image needs to be digitized, both in spatial location and brightness measure. In many image acquisition devices, e.g., cameras, scanners, a *digitizer* is included which converts the acquired images into their digital forms. In this dissertation, if not mentioned otherwise, all images refer to gray-scale digital images.

A conventional 2-D digital image can be expressed by a matrix $\{f(i, j), i =$

$1, \ldots, n_1, j = 1, \ldots, n_2\}$, where i, and j are the indices of the rows and columns, as shown below.

$$\begin{pmatrix} f(1,1) & f(1,2) & \cdots & f(1,n_2) \\ f(2,1) & f(2,2) & \cdots & f(2,n_2) \\ \vdots & \vdots & \ddots & \vdots \\ f(n_1,1) & f(n_1,2) & \cdots & f(n_1,n_2) \end{pmatrix}$$

Each element of the matrix is called a pixel of the image. The image resolution is related to n_1 and n_2. The resolution is high if they are large, and low if they are small. For the (i,j)-th pixel, the digitized value of $f(i,j)$ is conventionally assumed to be an integer in the range $[0, L-1]$, with 0 denoting black and $L-1$ denoting white. The magnitude of $f(i,j)$ quantifies the shade of the image at that position. The value of $f(i,j)$ is often called *gray level* of the image at the (i,j)-th pixel. For a more detailed description, see Gonzalez and Woods (1992), and Qiu (2005).

1.1 Image denoising and jump regression analysis

Assume that the observed image intensities $\{\xi_{ij}, i,j = 1, \ldots, n\}$ in a noisy 2-D image follow the following regression model:

$$\xi_{ij} = f(x_i, y_j) + \varepsilon_{ij}, \text{ for } i,j = 1, \ldots, n \tag{1.1}$$

where $\{(x_i, y_j) = (i/n, j/n), i,j = 1, \ldots, n\}$ are equally spaced pixel locations, f is the unknown image intensity function, and $\{\varepsilon_{ij}, i,j = 1, \ldots, n\}$ are independent and identically distributed (i.i.d.) random errors with mean 0 and unknown variance σ^2. While $n_1 = n_2 = n$ is used for simplicity, the regression model under the general case $n_1 \neq n_2$ can be easily written.

The image intensity function f has discontinuities/jumps at various places, called *edges* in the literature. For example, the image intensity function often has jumps at the boundaries of the image objects. So, edge-preserving image denoising is essentially the same problem as the jump-preserving surface estimation in regression analysis (Qiu, 2007). In jump regression analysis, the design points are not necessarily equally spaced, while in images they are often so. Clearly, edge-preserving image denoising can be achieved efficiently by developing and applying certain techniques of jump regression analysis.

1.2 Edge and edge-structure preserving image denoising

In image processing, noise removal is often used as a pre-processing tool. If certain image features e.g., edges, edge-structures etc. are blurred or lost during denoising, they can not be recovered later on. Because they often denote major characteristics of the image objects, it is very important to preserve them while removing noise from images.

In the literature, there are some existing procedures to denoise images while preserving edges and edge-structures. One group of methods are based on Bayesian estimation, using Markov Random Field (MRF) modeling and maximum *a posteriori* (MAP) algorithms, e.g., Besag (1974), Besag (1986), Fessler et al. (2000), Geman and Geman (1984), Godtliebsen and Sebastiani (1994), Marroquin et al. (2001), and Moussouris (1974). Some closely related methods use the regularization approach, by minimizing certain objective function that enforces a roughness penalty in addition to a term measuring fidelity of an estimator to the data (Li, 1995; Rivera and Marroquin, 2002; Rudin et al., 1992). Image denoising by median filtering, and other robust estimation procedures are widely used as pre-smoothing techniques in image processing, because they have certain abilities to preserve edges while denoising (Brownrigg, 1984; Gallagher and Wise, 1981; Hillebrand and Müller, 2007; Sun et al., 1994). Other image denoising techniques include adaptive smoothing filters (Polzehl and Spokoiny, 2000; Saint-Marc et al., 1991), bilateral filtering (Chu et al., 1998; Tomasi and Manduchi, 1998), diffusion filtering (Barash, 2002; Perona and Malik, 1990), wavelet transformation procedures (Chan et al., 2000; Figueiredo and Nowak, 2001; Nason and Silverman, 1994; Portilla et al., 2003), jump-preserving surface estimation procedures (Gijbels et al., 2006; Qiu, 1998, 2004, 2009; Sinha and Schunck, 1992; Yi and Chelberg, 1995), etc. See Qiu (2005, 2007) for more information about the existing literature on image denoising.

Most image denoising procedures mentioned above have abilities to preserve edges at places where the edge curvature is not large. At the places where the edges have angles or where their curvatures are large, the above methods usually blur or round those edge-structures. One major reason why this happens is that the edge-structures are hidden in image intensities and they are not easy to describe or measure (Chabat et al., 1999; Yang et al., 1996), and they are even more difficult to accommodate in the image denoising procedures (Gijbels et al., 2006). To better discuss this issue, the discrete version of a 2-D bilateral filtering procedure can be written as follows (cf., Section 7.4.2 of Qiu (2005)):

$$S^{(t+1)}(x,y) = \frac{1}{N^{(t+1)}(x,y)} \sum_{i=-k}^{k} \sum_{j=-k}^{k} S^{(t)}(x+i,y+j)w^{(t)}(x+i,y+j),$$

where $k > 0$ is the window size of the filter, $S^{(t+1)}(x,y)$ denotes the denoised image intensity at the pixel (x,y) in the $(t+1)$th iteration,

$$w^{(t)}(x+i,y+j) = \exp\left(-\frac{i^2+j^2}{2\sigma_d^2}\right) \exp\left(-\frac{(S^{(t)}(x+i,y+j) - S^{(t)}(x,y))^2}{2\sigma_r^2}\right),$$

and $N^{(t+1)}(x,y)$ is the summation of all weights $\{w^{(t)}(x+i,y+j)\}$. From the above expressions, it can be seen that, if a pixel $(x+i,y+j)$ and the given pixel (x,y)

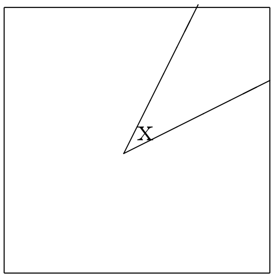

Figure 1.1: A toy example

are on two different sides of the edge curve in the neighborhood, then the weight $w^{(t)}(x+i, y+j)$ will be small because $S^{(t)}(x+i, y+j) - S^{(t)}(x, y)$ would be relatively large in such cases. However, $w^{(t)}(x+i, y+j)$ is still a positive value, although it is relatively small. In other words, observed image intensities in both sides of the edge curve are actually used in defining $S^{(t+1)}(x, y)$. Now, let us consider a toy example shown in Figure 1.1, in which there is an angular edge curve in a neighborhood of a given pixel located at the center (marked by "x"). In such cases, most pixels in the neighborhood are on the different side of the edge curve, compared to the given pixel. Therefore, the angle of the edge curve would be blurred by the bilateral filter.

Over the last few years, thanks to the advancement of image acquisition technologies, 3-D images become popular. For example, in medical science, certain image acquisition techniques, e.g., Magnetic Resonance Imaging (MRI), functional Magnetic Resonance Imaging (fMRI) can take 3-D images of various human/animal organs. Due to the artifacts of the image acquisition devices, and many other reasons, 3-D images often contain heavy noise. So, 3-D image denoising is very important in many disciplines including medical science. Most existing image denoising methods mentioned above are for 2-D images. One possibility of using them in denoising a 3-D image is to denoise each 2-D slice of the 3-D image. Clearly, this procedure can not work well, because the spatial information around each *voxel* is not used efficiently. (In the literature on 3-D images, the term *voxel* is used in place of *pixel*.) Another possibility is generalizing the 2-D denoising methods to those of 3-D. This can actually be achieved for most of the 2-D denoising methods. However, direct generalizations would also not perform well either in preserving edge-structures of 3-D images. One major reason is that, the edge locations in 2-D images form curves, whereas those in 3-D images form surfaces. Edge/jump surfaces in 3-D images have much more complicated structures than edge/jump curves in 2-D images. So, edge-structure preserving 3-D image denoising is emerging as a new research area. More discussions on this topic are provided in Sections 2.1 and 3.1.

1.3 Various noise models in medical images

In the image processing literature, most of the existing procedures assume the noise in the images to be additive, independent and identically distributed (i.i.d.) *Gaussian* noise. In real applications, this simple noise model often do not hold to be true. For example, the noise structure in magnitude MRI images usually follows a *rician* model (Bernstein et al., 1989; Sijbers et al., 1998; Nowak, 1999). Some images from radiography contain noise that are *Poisson* distributed (Webb, 1988). *Chi-square* distributed noise is also common in certain types of images.

When a conventional image denoising procedure is applied to an image that is corrupted with any of such noise, the image intensity values of the denoised image would not be close to the actual intensity values. In statistical terminology, the denoised image is said to be a *biased* estimate of the true image. This happens because such noise does not have the zero-mean property. Moreover, the noise in some cases is dependent on the image intensity values, which makes the estimation bias to be intensity dependent as well. So, it is even more difficult to efficiently denoise an image that is corrupted with intensity dependent non-zero mean noise (e.g., the *rician noise*). In the literature, there are broadly two approaches to solve this issue.

One of them is to accommodate the target noise model in the denoising procedure, to make sure that the denoised image is an unbiased estimate of the true image (Zhu et al., 2009). Another approach is to reduce/eliminate the bias after applying a conventional denoising procedure (Gudbjartsson and Patz, 1995; Wiest-Daessle et al., 2008).

Some of the existing methods to solve the 'bias problem' are computationally expensive. Many other methods estimate the bias by some naive approaches. However, those estimates of the bias are not good enough to solve the problem either. So, denoising an image that is corrupted with intensity dependent non-zero mean noise is also an important research area in medical imaging.

1.4 Organization of this dissertation

The remaining part of this dissertation is organized as follows. In Chapter 2, a 2-D image denoising procedure is proposed and its numerical performance in comparison with some existing methods is studied. A 3-D image denoising procedure is proposed in the first few Sections of Chapter 3. Some of its statistical properties are also provided. Numerical studies are presented in comparison with some state-of-the-art methods. Chapter 3 also presents a generalized version of our proposed 3-D image denoising procedure, along with some numerical examples. In Chapter 4, a bias correction procedure is proposed to denoise images that are corrupted with rician noise. Numerical studies in comparison with some existing methods are also provided. Some limitations of all our proposed methods are discussed in Chapter 5. In that chapter, some future research problems are also discussed. Proofs and derivations of some theoretical results are provided in the appendix.

Chapter 2

Edge-structure Preserving 2-D Image Denoising

In this Chapter, we first introduce the research problem of edge-structure preserving 2-D image denoising in the context of existing literature on that. Then, in Section 2.2, we propose a novel 2-D image denoising procedure which can preserve edges and major edge features, e.g., angles of the edges. Our method is based on nonparametric estimation of discontinuous surfaces from noisy data, in the framework of jump regression analysis, because a monochrome image can be regarded as a surface of the image intensity function and such a surface usually has discontinuities at the outlines of objects. In Section 2.3, we present some numerical results which show that our method works well in various applications, compared to some existing 2-D image denoising procedures.

2.1 Introduction

As discussed in Chapter 1, image denoising is often used for pre-processing images so that subsequent image analysis is more reliable (Gonzalez and Woods, 1992). Besides noise removal ability, another important requirement for image denoising procedures is that true image structures, such as edges, should be preserved in the denoising process. In this Chapter, we handle the 2-D image denoising problem in the framework of jump regression analysis (JRA), which is a research area handling regression models involving jumps and discontinuities (Qiu, 2005). In this framework, image denoising can be accomplished by estimating a discontinuous surface from noisy data, because a monochrome image can be regarded as a surface of the image intensity function and such a surface usually has discontinuities at the outlines of objects. A novel procedure is suggested in this chapter for estimating discontinuous surfaces from noisy data, which can preserve edges and major edge features, e.g., angles of the edges.

In the literature, there are some existing procedures for 2-D image denoising (Section 1.2, Chapter 1). Most existing image denoising and jump surface estimation procedures have some ability in preserving edges at places where the edge curvature is small. As discussed in Chapter 1, at places where the edges have angles or where their curvature is large, however, such edges would often be blurred or rounded by these existing methods. One major reason why this would happen is that the edge structures (e.g., angles) are hidden in observed image intensities, they are not easy to describe and measure (Chabat et al., 1999; Yang et al., 1996), and they are even more difficult to accommodate in the image denoising process (Gijbels et al., 2006).

In our opinion, edge structures are an important part of images, because they often denote major characteristics of image objects, and are easier to capture our visual attention than the parts of the edges with relatively small curvature. Therefore, they should be preserved during image denoising. In other words, a good image denoising procedure should preserve not only the parts of the edges with small curvature but also certain major edge structures, such as angles, corners, and other places on the edges with large curvature, although the latter goal is much more challenging than the former.

In this chapter, a 2-D image denoising procedure is suggested, which can preserve edges and major edge structures well. Our method consists of three major steps, outlined below. First, edge pixels are detected in the whole design space by an edge detector. Second, in a neighborhood of a given pixel, a piecewise linear curve is estimated from the detected edge pixels by a simple but efficient algorithm, to approximate the underlying edge segment in that neighborhood. Finally, observed image intensities on the same side of the estimated edge segment, as the given pixel, are averaged by the local linear kernel smoothing procedure (Fan and Gijbels, 1996), for estimating the true image intensity at the given pixel. The proposed 2-D image denoising procedure is described in detail in Section 2.2. Some numerical examples are presented in Section 2.3, for evaluating its numerical performance, in comparison with several existing denoising procedures.

2.2 Proposed methodology

We present our proposed methodology in three parts. In Section 2.2.1, 2-D local quadratic kernel (LQK) smoothing and a corresponding edge detection procedure is introduced. Local approximation to edge segments and edge-structure-preserving local denoising are described in Section 2.2.2. Data driven parameter selection is discussed in Section 2.2.3.

2.2.1 Edge detection by LQK smoothing

As discussed in Section 2.1, the first step of the proposed image denoising procedure is to detect edge pixels using an edge detector. Theoretically speaking, any reasonable edge detector can be used here. In the literature, most existing edge detectors are based on estimation of the first-order derivatives (Canny, 1986; Fleck, 1992; Qiu, 2002; Qiu and Bhandarkar, 1996) the second-order derivatives (Clark, 1988; Torre and Poggio, 1986) of the image intensity function. Recently, Sun and Qiu (2007) propose an edge detector that combines the major strengths of the two types of edge detectors, by using both the first-order and the second-order derivatives of the image intensity function. This edge detector will be used in all numerical examples of this Chapter, and it is briefly described below.

Assume that observed image intensities $\{Z_{ij}, i, j = 1, 2, \ldots, n\}$ follow the following 2-D regression model:

$$Z_{ij} = f(x_i, y_j) + \varepsilon_{ij}, \text{ for } i, j = 1, 2, \ldots, n, \tag{2.1}$$

where $\{(x_i, y_j), i, j = 1, 2, \ldots, n\}$ are equally spaced pixel locations, f is the unknown image intensity function, and $\{\varepsilon_{ij}, i, j = 1, 2, \ldots, n\}$ are independent and identically distributed (i.i.d.) random errors with mean 0 and unknown variance σ^2. At a given pixel (x, y), let us consider a circular neighborhood $O^*(x, y) = \{(u, v) : \sqrt{(u-x)^2 + (v-y)^2} \leq h_n^*\}$, where $h_n^* > 0$ is a bandwidth parameter. Then, LQK smoothing is accomplished by

$$\begin{aligned}\min_{a,b,c,d,e,f} \sum_{(x_i,y_j)\in O^*(x,y)} & \Big\{ Z_{ij} - \big[a + b(x_i - x) + c(y_j - y) + d(x_i - x)(y_j - y) + \\ & e(x_i - x)^2 + f(y_j - y)^2 \big] \Big\}^2 K\left(\frac{x_i - x}{h_n^*}, \frac{y_j - y}{h_n^*} \right),\end{aligned} \tag{2.2}$$

where K is a radially symmetric, bivariate density kernel function with support $\{(x, y) : x^2 + y^2 \leq 1\}$. The solution to a of the minimization problem (2.2) can be used as an estimator of the intensity $f(x, y)$, the solution to $(b, c)'$ as an estimator of the gradient vector $G(x, y) = (f_x(x, y), f_y(x, y))'$, and the solution to $(e, f)'$ as an estimator of $(f_{xx}(x, y), f_{yy}(x, y))'$. These estimators, denoted as $\widehat{f}(x, y)$, $\widehat{f}_x(x, y)$, $\widehat{f}_y(x, y)$, $\widehat{f_{xx}}(x, y)$, and $\widehat{f_{yy}}(x, y)$, are called LQK estimators in the literature (Fan and Gijbels, 1996). Along the estimated gradient direction $\widehat{G}(x, y) = (\widehat{f}_x(x, y), \widehat{f}_y(x, y))'$, if (x, y) is on an edge segment, then $\widehat{G}(x, y)$ would have large Euclidean length and $(\widehat{f_{xx}}(x, y), \widehat{f_{yy}}(x, y))'$ would have the *zero-crossing* properties that they are zero at

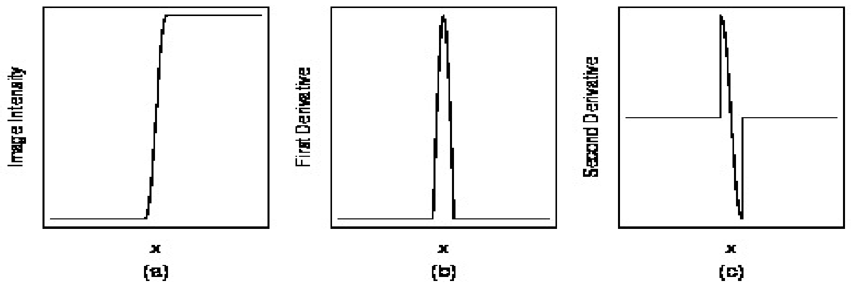

Figure 2.1: (a): 1-D profile of the image intensity surface around an edge segment. (b): First-order derivative of the 1-D profile. (c): Second-order derivative of the 1-D profile.

(x, y) and change signs on two different sides of the edge segment. See Figure 2.1 for a demonstration in one-dimensional cases. Then, a point (x, y) is flagged as a detected edge pixel if $\sqrt{\widehat{f_x}^2(x,y) + \widehat{f_y}^2(x,y)}$ is larger than a threshold value u_n and $\widehat{f_{xx}} + \widehat{f_{yy}}$ demonstrates the zero-crossing properties in $O^*(x, y)$. A formula for the threshold value is derived in Sun and Qiu (2007), which depends on a significance level α_n. In all our numerical examples presented in Section 2.3, α_n is fixed at 0.01.

2.2.2 Edge-structure preserving image denoising

Detected edge pixels are identified after the edge detection step discussed in the previous Section. In this Section, we describe the remaining two steps of the proposed 2-D image denoising procedure. At a given pixel (x, y), we consider its circular neighborhood

$$O(x, y) = \{(u, v) : \sqrt{(u - x)^2 + (v - y)^2} \leq h_n\},$$

where $h_n > 0$ is a bandwidth parameter which could be different from h_n^* used in (2.2). Detected edge pixels in $O(x, y)$ are denoted by $\{(w_k, v_k), k = 1, 2, \ldots, m\}$. Our major goal here is to estimate $f(x, y)$ from observations in $O(x, y)$ with possible edges preserved.

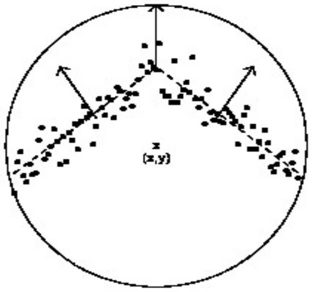

Figure 2.2: The small dots denote detected edge pixels in $O(x, y)$, and the dashed line denotes the true edge curve which contains an angle. The vertical arrow in the middle denotes $\overline{G}(x, y)$, and the two arrows on its two sides denote two typical estimated gradients of f along the two rays of the angle.

From (2.2), the estimated gradient at a detected edge pixel (w_k, v_k) is $\widehat{G}(w_k, v_k)$, for $k = 1, 2, \ldots, m$. Intuitively, if the underlying edge curve in $O(x, y)$ contains an angle, then gradients of f along its two rays would point to two different directions. Therefore, in such cases, we would expect that the estimated gradients $\{\widehat{G}(w_k, v_k), k = 1, 2, \ldots, m\}$ can be divided into two groups, each group corresponds to a ray of the angle, and the estimated gradients in each group would vary around the perpendicular direction of the corresponding ray. See Figure 2.2 for a demonstration. To estimate the edge curve in $O(x, y)$, we propose an algorithm with the following steps.

1. Compute the simple average of $\{\widehat{G}(w_k, v_k), k = 1, 2, \ldots, m\}$, denoted as $\overline{G}(x, y)$.

2. Divide $\{(w_k, v_k), k = 1, 2, \ldots, m\}$ into two groups $E = \{(w_k, v_k) : \theta(\widehat{G}(w_k, v_k), \overline{G}(x, y)) \leq 0\}$ and $E^c = \{(w_k, v_k) : \theta(\widehat{G}(w_k, v_k), \overline{G}(x, y)) > 0\}$, where $\theta(\vec{u}, \vec{v}) \in [-\pi/2, \pi/2]$ denotes the angle from vector $\vec{u}$ to vector $\vec{v}$.

3. Compute the line L that goes through the center of E in the perpendicular direction of $\overline{G}_E(x, y)$, where $\overline{G}_E(x, y)$ is the average of the estimated gradients in E.

4. Compute the line L^c that goes through the center of E^c in the perpendicular direction of $\overline{G}_{E^c}(x,y)$, where $\overline{G}_{E^c}(x,y)$ is the average of the estimated gradients in E^c.

5. Assume that L and L^c cross at point A. If A is located outside of $O(x,y)$, then the two line segments of L and L^c inside $O(x,y)$ are used for estimating the edge segments in $O(x,y)$. In the case when A is located inside of $O(x,y)$, L and L^c is each divided into two half lines by A, and $O(x,y)$ is divided into four parts by the half lines. The two half lines, one on each of L and L^c, that contain the centers of E and E^c, respectively, are selected for estimating the edge curve in $O(x,y)$. Obviously, the two selected half lines form an angle with vertex A.

The estimated edge segment(s) by the above algorithm divides $O(x,y)$ into two or three parts, depending on whether or not the point A defined in step 5 is located inside $O(x,y)$. The part containing the given pixel (x,y) is denoted as $B(x,y)$. Then, the estimator of $f(x,y)$, denoted as $\widehat{f}(x,y)$, can be defined by the solution to a of the following minimization problem:

$$\min_{a,b,c\in R} \sum_{(x_i,y_j)\in B(x,y)} \{Z_{ij} - [a + b(x_i - x) + c(y_j - y)]\}^2 K\left(\frac{x_i - x}{h_n}, \frac{y_j - y}{h_n}\right). \quad (2.3)$$

Obviously, $\widehat{f}(x,y)$ is the local linear kernel (LLK) estimator of $f(x,y)$, constructed from observations in the one-sided neighborhood $B(x,y)$.

Next, we propose a modification to improve the performance of the proposed denoising procedure (2.3) and to simplify its computation as well. In regions where the true image intensity function f is continuous, it is desirable to use a larger bandwidth to construct an estimator of f, compared to the bandwidth used around true edges, so that the estimator is better in removing noise. To this end, at a given pixel (x,y), we first consider a neighborhood $\tilde{O}(x,y) = \{(u,v) : \sqrt{(u-x)^2 + (v-y)^2} \leq \tilde{h}_n\}$ with a larger bandwidth $\tilde{h}_n$ ($> h_n$). In $\tilde{O}(x, y)$, if the number of detected edge pixels is so small that it is unlikely to have a true edge segment in it, then we do not implement the last two steps of the proposed denoising procedure described above. In such cases, we can simply define $\widehat{f}(x, y)$ to be the conventional LLK estimator in $\tilde{O}(x, y)$, which is the solution to a of the minimization problem (2.3), after $B(x, y)$ is replaced by $\tilde{O}(x, y)$. To do so, there are at least two major benefits. One is that $\widehat{f}(x, y)$ is defined using all pixels in $\tilde{O}(x, y)$ in such cases; its denoising ability is thus greatly improved, compared to the estimator constructed in $B(x, y)$. The second benefit is that computation involved is greatly simplified. For a typical observed image, there are many pixels at which no edge segments exist in their neighborhoods. Therefore, the above two benefits are substantial. In all numerical examples presented in next section, if the number of detected edge pixels is smaller than or equal to $[n\tilde{h}_n]$, where $[x]$ denotes the integer part of x, then we define $\widehat{f}(x, y)$ to be the conventional LLK estimator in $\tilde{O}(x, y)$.

When the number of detected edge pixels is larger than $[n\tilde{h}_n]$ in $\tilde{O}(x, y)$, the chance is high that there is a true edge segment in the neighborhood. In such cases, we consider using a smaller bandwidth h_n ($< \tilde{h}_n$) to deal with the potential edges. In neighborhood $O(x, y)$ with bandwidth h_n, if the number of detected edge pixels is smaller than or equal to $[nh_n]$, then $\widehat{f}(x, y)$ is defined by the conventional LLK estimator in $O(x, y)$. Otherwise, $\widehat{f}(x, y)$ is defined by procedure (2.3). The entire proposed image denoising procedure can now be summarized as follows.

Proposed Image Denoising Procedure

- Detect edge pixels using an edge detector such as the one described in Section 2.2.1.

- For a given pixel (x, y), count the number of detected edge pixels in neighborhood $\tilde{O}(x, y)$. If this number is smaller than or equal to $[n\tilde{h}_n]$, then define

$\widehat{f}(x,y)$ to be the conventional LLK estimator in $\tilde{O}(x,y)$, and continue the denoising procedure for the next pixel. If this number is larger than $[n\tilde{h}_n]$, then consider a smaller neighborhood $O(x,y)$ and count the number of detected edge pixels in that neighborhood. If the number is smaller than or equal to $[nh_n]$, then $\widehat{f}(x,y)$ is defined to be the conventional LLK estimator in $O(x,y)$. Otherwise, estimate the edge segment in $O(x,y)$ using the algorithm described three paragraphs above, and proceed to the next step.

- Compute $\widehat{f}(x,y)$ using (2.3), and continue the denoising procedure for the next pixel.

2.2.3 Selection of procedure parameters

In the proposed image denoising procedure, there are four parameters u_n, h_n^*, $\tilde{h}_n$ and h_n (cf., expressions (2.2), (2.3), and the related discussion). They should be chosen properly because performance of the proposed procedure depends on their values. For instance, if u_n is chosen too large, then some real edge pixels would be missed by the edge detector discussed in Section 2.2.1. Consequently, some jumps in f would be blurred in the denoising process. Similarly, the bandwidths h_n^*, $\tilde{h}_n$ and h_n also play an important role in image denoising. Theoretically speaking, we can choose these parameters by minimizing the Mean Integrated Squared Error (MISE) of the surface estimator, defined by:

$$MISE(\widehat{f},f) = E\left[\int_0^1 \int_0^1 \left(\widehat{f}(x,y) - f(x,y)\right)^2 \, dxdy\right], \tag{2.4}$$

where E denotes the expectation with respect to the probability distribution of $\widehat{f}(x,y)$. In practice, because f is unknown, this method can not be actually used. In this paper, we suggest using a modified version of the conventional cross-validation (CV) procedure. Remember that, in the proposed denoising procedure, neighborhoods of two different sizes (i.e., $\tilde{h}_n$ and h_n) are used. For pixels that are quite far away from true edges, their neighborhoods have width $\tilde{h}_n$. For the other pixels, their neighborhoods have width h_n. Let I be the set of pixels whose neighborhoods have width h_n, $|I|$ be the number of pixels in I, and

$$\begin{aligned} &CV(u_n, h_n^*, \tilde{h}_n, h_n) = \\ &\frac{\lambda}{|I|} \sum_{(x_i,y_j)\in I} \Big[Z_{ij} - \widehat{f}_{-i,-j}(x_i,y_j)\Big]^2 + \frac{1-\lambda}{n^2-|I|} \sum_{(x_i,y_j)\in I^c} \Big[Z_{ij} - \widehat{f}_{-i,-j}(x_i,y_j)\Big]^2, \quad (2.5) \end{aligned}$$

where I^c denotes the complementary set of I, $\widehat{f}_{-i,-j}(x_i, y_j)$ is the "leave-one-out" estimator of $f(x_i, y_j)$ obtained by (3) when the (i, j)th observation is not used (cf., Section 2.4.4 in Qiu (2005)), and λ is a weighting parameter. It should be pointed out that both I and I^c depend on u_n, h_n^*, $\tilde{h}_n$ and h_n, although it is not explicit in notation. From (2.5), we can see that λ controls the trade-off between edge-preservation and noise removal when choosing the parameters. In the extreme case when $\lambda = 1$, the second term on the right hand side of (2.5) would disappear. In such cases, the parameters are chosen to best preserve the edges, and the performance of the denoised image in continuity regions of f are actually not taken into account. In the other extreme case when $\lambda = 0$, the first term on the right hand side of (2.5) would disappear and the parameters are actually chosen to best remove noise in the continuity regions of f. In practice, one natural choice for λ is $|I|/n^2$, in which case procedure ((2.5)) becomes the conventional CV procedure. By ((2.5)), selected parameter values are those minimizing $CV(u_n, h_n^*, \tilde{h}_n, h_n)$ under the constraint that $\tilde{h}_n > h_n$. It should be pointed out that, in all our numerical studies presented in next section, we actually did not put the above constraint when searching for the parameter values by CV. From the results (cf., Tables 2.1–2.3 in Section 2.3), we can see that searched values of $\tilde{h}_n$ and h_n satisfy the condition that $\tilde{h}_n > h_n$ in all cases, which implies that this intuitively reasonable constraint is indeed reasonable in practice.

2.3 Numerical examples

In this section, we present some numerical results regarding the performance of the proposed image denoising procedure (denoted as NEW), in comparison with several existing image denoising methods. Four existing methods are considered here, including a recent denoising procedure based on gradient estimation and one-sided surface estimation (denoted as GE, Gijbels et al. (2006)), a denoising method based on MRF modeling (denoted as MRF, Godtliebsen and Sebastiani (1994)), a wavelet transformation method (denoted as WT, Portilla et al. (2003)), and the conventional local median filter (denoted as MED).

First, we present some simulation results when the true image intensity function is one of the following three functions:

$$\begin{aligned}
f_1(x,y) &= -2(x-0.5)^2-2(y-0.5)^2+\phi(x\geq 0.4)\phi(y\geq 0.3)\phi(2x+y\leq 1.6),\\
f_2(x,y) &= -2(x-0.5)^2-2(y-0.5)^2+\\
&\quad \phi\Big(\phi(y\geq 0.3)\phi(y-\sqrt{3}x\leq 0.8-0.5\sqrt{3})\phi(y+\sqrt{3}x\leq 0.8+0.5\sqrt{3})+\\
&\quad \phi(y\leq 0.7)\phi(y-\sqrt{3}x\geq 0.2-0.5\sqrt{3})\phi(y+\sqrt{3}x\geq 0.2+0.5\sqrt{3})>0\Big),\\
f_3(x,y) &= 0.5(1-x)y+(1-0.5(1-x)y)\phi(y\leq 3(0.25-(x-0.5)^2))\phi((x-0.5)^2\\
&\quad +y^2\geq 0.3)+(1-0.5(1-x)y)\phi(0.48\leq x\leq 0.52)\phi(0.25\leq y\leq 0.5),
\end{aligned}$$

where $\phi(a)$ is the indicator function which equals 1 when $a =$"True" and 0 otherwise. One realization of these three functions from model (1) when $n = 256$ and $\sigma = 0.5$ is presented in the first column of Figure 2.3. From the plots, we can see that edges of f_1 have three angles of different sizes, edges of f_2 have twelve angles, and edges of f_3 have different curvature at different places. In model (1), random errors are generated from distribution $N(0, \sigma^2)$. We consider three σ values 0.25, 0.5, and 0.75, representing low, medium, and high noise levels, and two n values 128 or 256, denoting two different image resolutions.

In method NEW, to save some computation, parameter α_n used for determining the threshold value u_n for edge detection is fixed at 0.01. In such cases, detected edges look reasonably well. Based on our numerical experience, as long as detected edges are reasonably good, their effect on denoised image is minimal. Besides u_n, procedure NEW has another three parameters, h_n^*, $\tilde{h}_n$ and h_n, to choose. The method GE has three parameters: a bandwidth parameter, and two threshold parameters used in surface estimation and corner preservation, respectively. The MRF method combines the ideas of using a discontinuity labeling process (Geman and Geman 1984) and the iterated conditional modes algorithm (Besag 1986). This procedure assumes that a binary line component exists between any two vertically or horizontally neighboring pixels, with 1 denoting an existing edge between the two pixels and 0 denoting no edge. In a 3×3 neighborhood of a given pixel, there are 12 line components and 2^{12} possible configurations of these components. To use this procedure, probabilities of the 2^{12} possible line configurations need to be specified. In this section, these probabilities are estimated from the true image intensity function values at the design points, which is in favor of this procedure. Besides the line configurations, it has three positive procedure parameters α, β and λ to determine. In the method WT, the default family of wavelets (which is Daubechies' "extremal phase" wavelet), the "full steerable pyramid" image decomposition procedure, the "Bayes least square (BLS)" solution, and the "symmetric" boundary handling condition are used. Other parameters are chosen to be the ones suggested by Portilla et al. (2003). The method MED defines the surface estimator by the sample median of the observed image intensities in a circular neighborhood of a given pixel. So, it has one parameter (i.e., the bandwidth) to choose. Because it is simple to use and has certain ability in preserving edges while removing noise, it is widely used as a pre-smoothing procedure (cf., Gonzalez

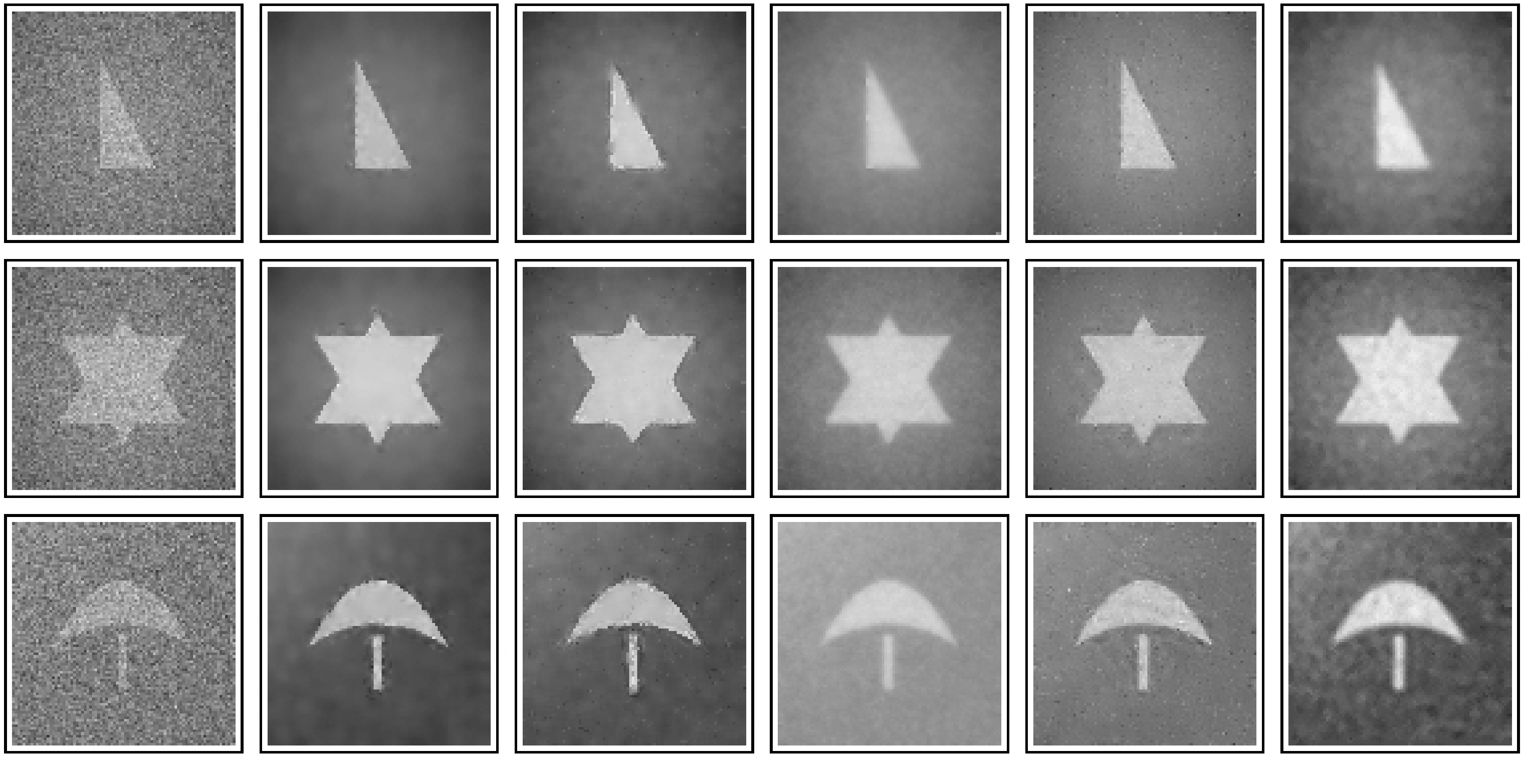

Figure 2.3: The first column denotes three noisy images when f equals f_1, f_2, or f_3, $n = 256$, and $\sigma = 0.5$. The remaining five columns denote denoised images by the methods NEW, GE, MRF, WT, and MED, respectively, when their procedure parameters are chosen to be the corresponding ones listed in Tables 2.1–2.3.

and Woods (1992), Chapter 4).

For each denoising method considered, 100 replications are performed in each combination of f, σ, and n. Their parameter values are searched so that the MISE value (cf., expression (2.4)), estimated by the sample mean of

$$ISE(\widehat{f}, f) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} [\widehat{f}(x_i, y_j) - f(x_i, y_j)]^2$$

over 100 replications, reaches the minimum. The estimated MISE values and the corresponding standard errors of ISE of various methods are presented in Tables 2.1–2.3, along with searched parameter values, for cases when f equals f_1, f_2, and f_3, respectively. For investigating ability of various methods in preserving edge structures, their estimated local MISE values, computed in circular neighborhoods of radius 0.1 when $n = 128$ and radius 0.05 when $n = 256$ of edge angles, along with the corresponding standard errors of local ISE are also presented in the tables. From the tables, it can be seen that the proposed method NEW is uniformly better in quite large margins than the remaining methods, in terms of both estimated MISE and estimated local MISE, when noise level is medium to high (i.e., $\sigma = 0.5$ or 0.75). When the noise level is low (i.e., $\sigma = 0.25$), it seems that procedure MRF always performs the best, which is consistent to the findings in Gijbels et al. (2006). For the observed images presented in the first column of Figure 2.3 when $n = 256$ and $\sigma = 0.5$, the denoised images by various methods when their parameters are chosen to be the ones corresponding to the results presented in Tables 2.1–2.3 (cf., columns corresponding to cases when $n = 256$ and $\sigma = 0.5$) are presented in columns 2–6 of Figure 2.3. From the plots, we can see that certain methods (e.g., GE, MRF, WT) either do not preserve angular edges well, or do not remove noise sufficiently. Method MED blurs all edges to a certain degree when removing noise. As a comparison, the denoised images by the proposed method (cf., the 2nd column in Figure 2.3) preserve all edges reasonably well when most noise has been removed. As a side note, for the observed triangle image shown

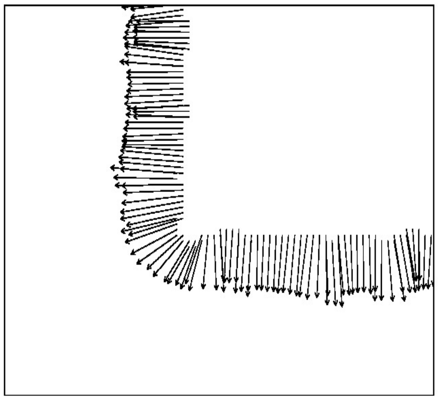

Figure 2.4: The estimated gradients at the detected edge pixels around the lower-left angle of the triangle image shown in the (1,1)-th panel of Figure 2.3.

in the (1,1)-th panel of Figure 2.3, the estimated gradients at the detected edge pixels around the lower-left angle by the proposed method are shown in Figure 2.4. From the figure, it can be seen that the detected edge pixels around that angle can indeed be divided into two groups using the estimated gradients for approximating the true edge curve, as demonstrated in Figure 2.2.

Next, we consider a real test image including a maple leaf in the middle. The image intensities are in the range $[0, 255]$, and the image has 160×160 pixels. A noisy version of this image with i.i.d. noise from $N(0, 100^2)$ is presented in the (1,1)-th plot of Figure 2.5, from which we can see that the boundary of the maple leaf has a number of quite sharp angles. We then apply the methods NEW, GE, MRF, WT, and MED to this example. Their parameters are searched by minimizing the estimated MISE values obtained from 100 replications, as in Tables 2.1–2.3. The estimated MISE values, their standard errors, and the search parameters values are presented in the first column of Table 2.4, from which we can see that the method NEW has the smallest MISE value. The denoised images by these methods with the parameters chosen to be the ones in Table 2.4 are presented in Figure 2.5. We can see

Table 2.1: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the estimated local MISE value and the corresponding standard error of local ISE computed in circular neighborhoods of true edge angles with width 0.1 when $n = 128$ and width 0.05 when $n = 256$, and the third line presents the searched procedure parameter values. This table considers the case when $f = f_1$.

	n=128			n=256		
Method	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$
NEW	.0029 (.0003) .0118 (.0013) .023,.070,.023	.0043 (.0005) .0172 (.0032) .023,.086,.031	.0061 (.0006) .0246 (.0028) .031,.102,.047	.0016 (.0001) .0128 (.0018) .012,.047,.012	.0025 (.0003) .0202 (.0036) .012,.051,.020	.0034 (.0003) .0303 (.0035) .016,.055,.027
GE	.0022 (.0002) .0108 (.0011) .031, .04, .4	.0080 (.0007) .0311 (.0033) .047, .06, .4	.0134 (.0006) .0475 (.0028) .055, .14, .8	.0011 (.0001) .0201 (.0014) .020, .04, .6	.0043 (.0003) .0355 (.0032) .027, .06, .6	.0081 (.0002) .0494 (.0030) .027, .16, .6
MRF	.0013 (.0002) .0035 (.0015) 1, 40, 1	.0100 (.0004) .0329 (.0013) 15, 5, 15	.0137 (.0008) .0431 (.0025) 15, 4, 20	.0009 (.0001) .0040 (.0019) .5, 35, 1	.0067 (.0003) .0417 (.0039) .5, 10, .5	.0091 (.0004) .0482 (.0034) 10, 5, 10
WT	.0030 (.0001) .0084 (.0007) –	.0098 (.0005) .0213 (.0018) –	.0200 (.0013) .0356 (.0037) –	.0020 (.0001) .0089 (.0001) –	.0069 (.0002) .0214 (.0016) –	.0149 (.0006) .0354 (.0032) –
MED	.0050 (.0002) .0169 (.0010) .023	.0112 (.0005) .0338 (.0019) .039	.0160 (.0008) .0495 (.0025) .055	.0028 (.0001) .0255 (.0010) .020	.0067 (.0002) .0464 (.0019) .027	.0098 (.0003) .0566 (.0028) .031

Table 2.2: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the estimated local MISE value and the corresponding standard error of local ISE computed in circular neighborhoods of true edge angles with width 0.1 when $n = 128$ and width 0.05 when $n = 256$, and the third line presents the searched procedure parameter values. This table considers the case when $f = f_2$.

	n=128			n=256		
Method	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$
NEW	.0052 (.0004) .0131 (.0011) .023,.063,.023	.0085 (.0006) .0209 (.0015) .023,.086,.039	.0122 (.0008) .0290 (.0021) .031,.094,.055	.0027 (.0001) .0137 (.0009) .012,.043,.012	.0043 (.0003) .0227 (.0013) .012,.109,.023	.0062 (.0003) .0310 (.0016) .016,.055,.027
GE	.0050 (.0002) .0104 (.0006) .023, .04, .4	.0128 (.0007) .0280 (.0018) .047, .06, .4	.0197 (.0007) .0413 (.0014) .047, .14, .8	.0020 (.0001) .0115 (.0006) .020, .04, .6	.0064 (.0003) .0257 (.0016) .027, .06, .6	.0118 (.0003) .0404 (.0013) .027, .16, .6
MRF	.0021 (.0003) .0042 (.0008) .5, 45, 1	.0152 (.0004) .0349 (.0010) .5, 5, 15	.0221 (.0009) .0509 (.0019) 1, 5, 20	.0012 (.0001) .0043 (.0008) .5, 35, 1	.0096 (.0002) .0358 (.0011) 2, 5, .5	.0131 (.0004) .0519 (.0015) 10, 5, 5
WT	.0048 (.0002) .0101 (.0004) –	.0132 (.0006) .0235 (.0011) –	.0247 (.0014) .0382 (.0023) –	.0029 (.0001) .0105 (.0004) –	.0089 (.0002) .0239 (.0010) –	.0178 (.0006) .0387 (.0019) –
MED	.0071 (.0002) .0156 (.0005) .023	.0161 (.0005) .0316 (.0012) .031	.0229 (.0009) .0462 (.0020) .047	.0042 (.0001) .0189 (.0005) .016	.0097 (.0002) .0397 (.0010) .023	.0140 (.0004) .0530 (.0015) .027

Table 2.3: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the estimated local MISE value and the corresponding standard error of local ISE computed in circular neighborhoods of true edge angles with width 0.1 when $n = 128$ and width 0.05 when $n = 256$, and the third line presents the searched procedure parameter values. This table considers the case when $f = f_3$.

	n=128			n=256		
Method	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$	$\sigma = .25$	$\sigma = .5$	$\sigma = .75$
NEW	.0034 (.0002) .0115 (.0007) .023,.070,.023	.0055 (.0004) .0190 (.0016) .023,.102,.039	.0085 (.0007) .0272 (.0028) .031,.109,.047	.0018 (.0006) .0110 (.0006) .012,.047,.012	.0030 (.0002) .0183 (.0015) .012,.051,.020	.0048 (.0003) .0296 (.0022) .016,.055,.031
GE	.0046 (.0002) .0106 (.0010) .023, .04, .4	.0118 (.0006) .0283 (.0020) .039, .08, .4	.0165 (.0007) .0381 (.0020) .039, .22, .6	.0018 (.0001) .0126 (.0007) .023, .02, .4	.0057 (.0002) .0217 (.0019) .031, .04, .4	.0100 (.0003) .0284 (.0022) .035, .09, .4
MRF	.0020 (.0003) .0050 (.0011) 1, 45, .5	.0122 (.0004) .0325 (.0013) .5, 4, 5	.0165 (.0008) .0421 (.0025) .5, 3, 1	.0014 (.0001) .0048 (.0012) .5, 30, .5	.0077 (.0002) .0339 (.0014) .1, 6, 10	.0105 (.0004) .0425 (.0025) .1, 4, 15
WT	.0042 (.0001) .0098 (.0005) –	.0116 (.0006) .0224 (.0016) –	.0224 (.0013) .0363 (.0034) –	.0026 (.0001) .0094 (.0006) –	.0080 (.0003) .0214 (.0015) –	.0163 (.0006) .0346 (.0029) –
MED	.0062 (.0002) .0185 (.0009) .023	.0135 (.0005) .0347 (.0019) .031	.0190 (.0008) .0518 (.0023) .047	.0037 (.0001) .0192 (.0007) .016	.0080 (.0002) .0367 (.0017) .023	.0113 (.0003) .0470 (.0027) .027

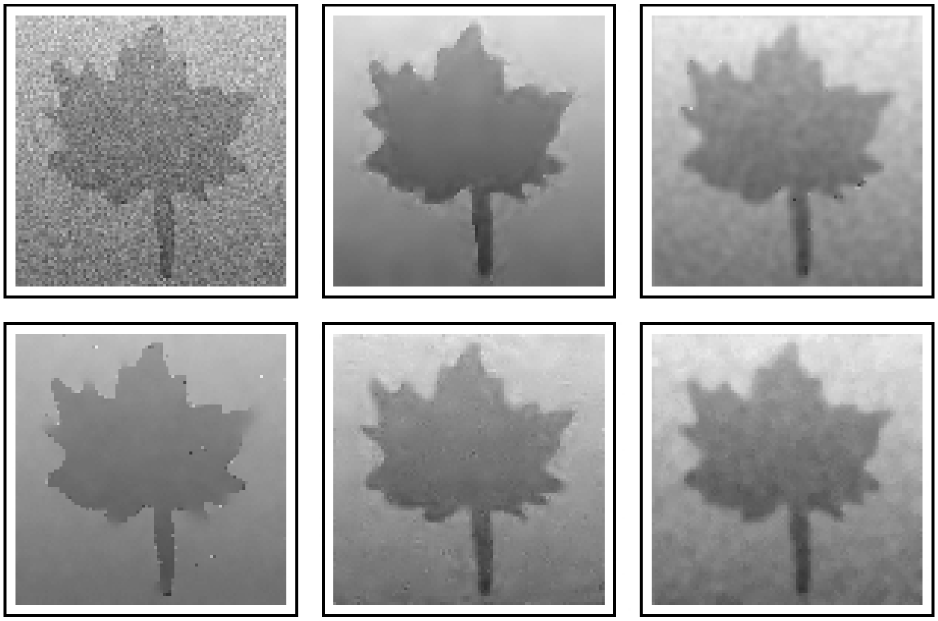

Figure 2.5: The noisy maple leaf image and its denoised images by methods NEW, GE, MRF, WT, and MED. The noise is from $N(0, 100^2)$.

that method NEW preserves the edge structure and removes the noise well, compared to its peers.

We next consider a magnetic resonance image (MRI) of a knee part of human body with 128×128 pixels. The image intensity levels range from 0 to 255 as usual. A noisy version with i.i.d. noise from $N(0, 100^2)$ is presented in the (1,1)-th plot of Figure 2.6. As in the previous example, the searched parameters and the estimated MISE values of the methods NEW, GE, MRF, WT, and MED, based on 100 replications, are presented in the second column of Table 2.4. From the table, it can be seen that method WT performs the best, and method NEW is better than the remaining three methods. The denoised images by these methods from the one shown in the (1,1)-th plot are presented in the next five plots of Figure 2.6, from which we can see that both methods NEW and WT preserve edges well, compared to the remaining three methods. It seems that method NEW preserves edge structures a little better than

Table 2.4: The first line in each entry presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), and the second line presents the searched procedure parameter values. The five columns are for cases of the leaf image, the knee image with Gaussian noise (GN), the knee image with double exponential noise (DEN), the knee image with variable noise (VN) level, and the knee image with 30% salt-and-pepper noise (SPN), respectively.

Method	Leaf	Knee (GN)	Knee (DEN)	Knee (VN)	Knee (SPN)
NEW	431.52 (13.65) .019,.063,.025	855.90 (21.89) .032,.070,.032	762.40 (19.44) .032,.070,.032	846.07 (29.80) .032,.070,.032	1314.68 (40.38) .032,.078,.032
GE	713.54 (13.47) .019,20000,0.8	941.48 (17.71) .023,30000,1	872.88 (14.04) .023,10000,1	869.67 (16.64) .023,30000,1	3041.21 (80.12) .023,10000,1
MRF	576.51 (38.84) 1,.0002,.16	920.75 (18.23) 1,.0001,20	1469.46 (81.40) .25,.0001,20	1213.01 (111.00) .5,.0001,20	1813.47 (76.29) 1,0.0001,20
WT	586.58 (18.66) –	840.14 (28.88) –	825.54 (39.05) –	1233.66 (104.63) –	1375.81 (52.12) –
MED	783.92 (19.32) .025	1151.27 (21.52) .023	763.72 (14.23) .016	989.26 (19.57) .023	556.82 (19.74) .016

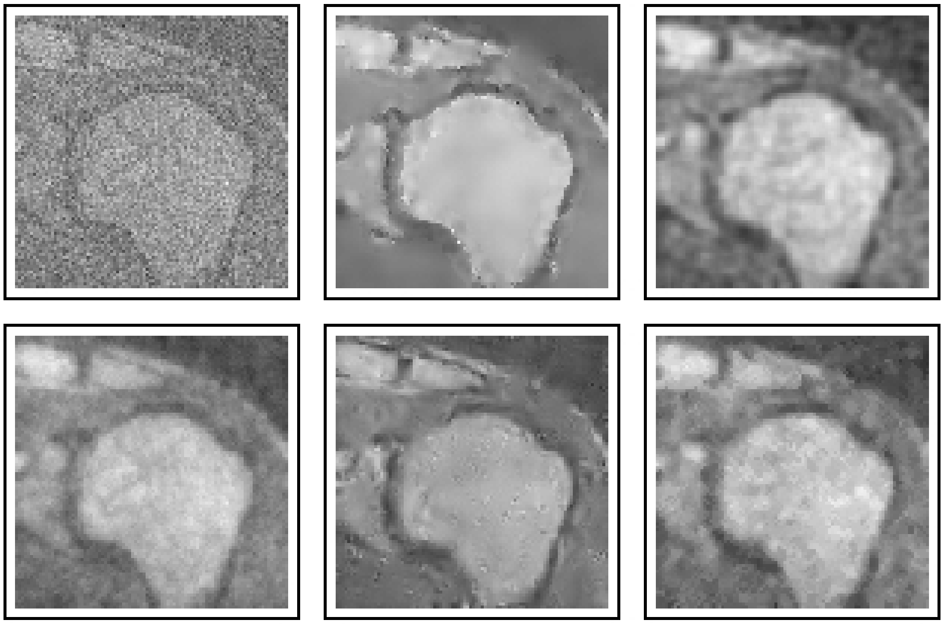

Figure 2.6: The noisy knee magnetic resonance image and its denoised images by methods NEW, GE, MRF, WT, and MED. The noise is from $N(0, 100^2)$.

method WT (cf., the edge segment surrounding the central white part in the image).

The WT method considered here is designed for handling Gaussian noise. So, it is not surprising that it would perform well in the example of Figure 2.6 where noise is from a Gaussian distribution. Next, we consider the same MRI image, but add noise from a Double Exponential distribution with location parameter 0 and scale parameter 60. The corresponding results are shown in the third column of Table 2.4 and in Figure 2.7. We can see that WT does not perform well in this case in terms of the estimated MISE.

In the above examples, noise level is homogeneous in entire observed images, for simplicity. It should be pointed out that the proposed procedure can also handle cases when the noise level depends on location. As an example, suppose that the noise level has the expression

$$\sigma(x, y) = 40 \exp\left(3.5(x - 0.5)^2 + 3.5(y - 0.5)^2\right).$$

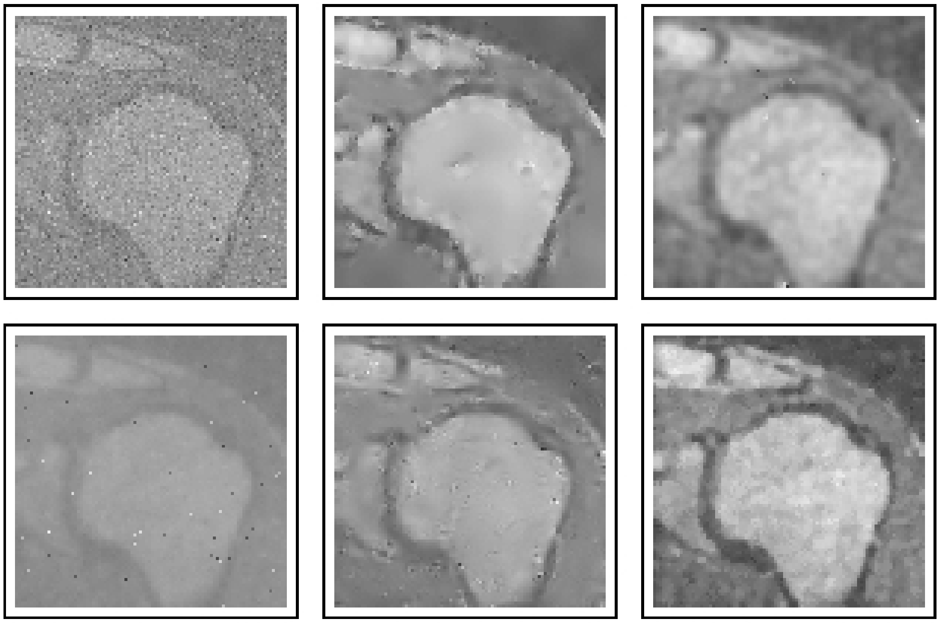

Figure 2.7: The noisy knee magnetic resonance image and its denoised images by methods NEW, GE, MRF, WT, and MED. The noise is from a Double Exponential distribution with location parameter 0 and scale parameter 60.

Obviously, $\sigma(x, y)$ is small in the central region of the image and large in the border region. An observed noisy knee image with i.i.d. noise from distribution $N(0, \sigma^2(x, y))$ is shown in the (1,1)-th plot of Figure 2.8. The estimated MISE values and denoised images of various methods are presented in the fourth column of Table 2.4 and in Figure 2.8, respectively. It can be seen that method NEW performs the best in this case.

Finally, we consider adding salt-and-pepper noise to 30% randomly selected pixels of the knee image. The added noise is binary. It equals either the maximum intensity level of the true knee image or the minimum intensity level, by random. The noisy image is shown in the (1,1)-th plot of Figure 2.9. The estimated MISE values and the denoised images of various methods are presented in the last column of Table 2.4 and in Figure 2.9, respectively. We can see that the median method MED performs the best in such a case, as expected, because median methods are robust to a small

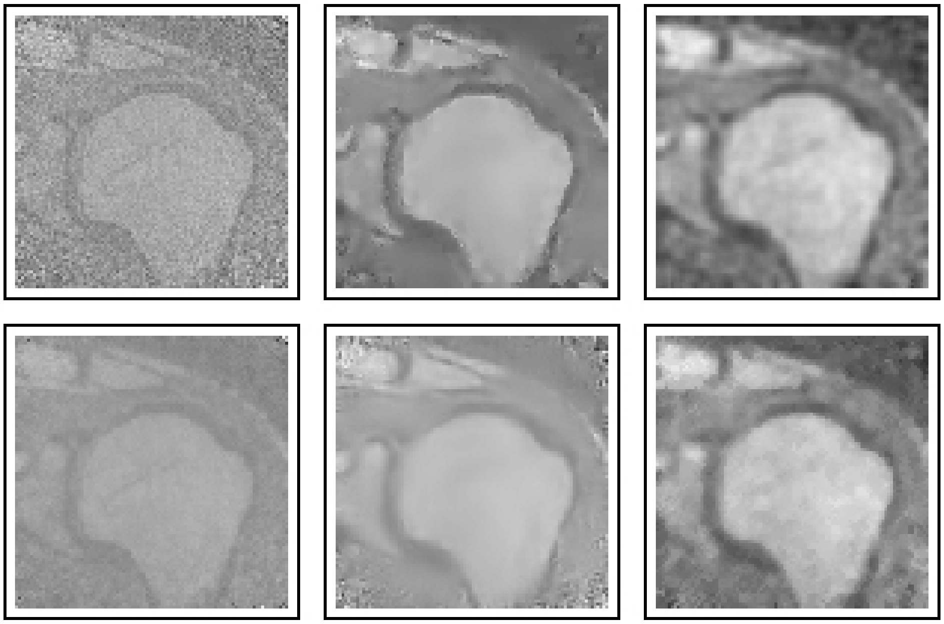

Figure 2.8: The noisy knee magnetic resonance image and its denoised images by methods NEW, GE, MRF, WT, and MED. The noise level changes with location by $\sigma(x, y) = 40 \exp\left(3.5(x - 0.5)^2 + 3.5(y - 0.5)^2\right)$.

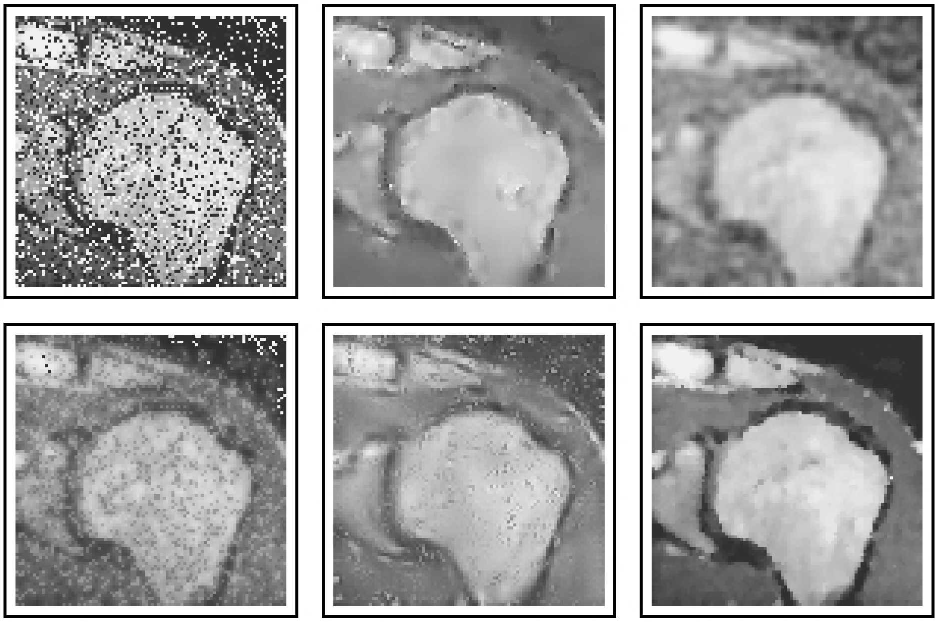

Figure 2.9: The noisy knee magnetic resonance image and its denoised images by methods NEW, GE, MRF, WT, and MED. Only 30% randomly selected pixels are added salt-and-pepper noise.

amount of large or small intensity values such as those with salt-and-pepper noise added. When the percentage of pixels with salt-and-pepper noise added increases, its performance becomes worse and worse, which has been numerically confirmed by us, although the results are skipped here. Method NEW performs better than the remaining three methods in this case.

Chapter 3

Edge-structure Preserving 3-D Image Denoising

In this Chapter, we first introduce the research problem of edge-structure preserving 3-D image denoising. It has a lot of applications in magnetic resonance imaging (MRI), functional MRI (fMRI), etc. Then, in Section 3.2, we propose a novel 3-D image denoising procedure, based on nonparametric estimation of a 3-D jump surface from noisy data. One important feature of this method is its ability to preserve edges and major edge-structures, such as intersections of two edge surfaces, pyramids, pointed corners, and so forth. Some of its theoretical results are discussed in Section 3.3. In Section 3.4, we study its numerical performance, and our numerical studies show that it works well in various applications. In Section 3.5, a generalization of our proposed 3-D image denoising method is discussed. Proofs of the theoretical results are provided in the appendix.

3.1 Introduction

As discussed in Chapter 1, three-dimensional (3-D) images become increasingly popular in applications. For instance, in magnetic resonance imaging (MRI) and functional MRI (fMRI), to study the biological mechanism of a 3-D object (e.g., a patient's head), people traditionally acquire a set of two-dimensional (2-D) images from slices of the 3-D object. Then, the 3-D object is reconstructed from the 2-D images, which is a research area called 3-D image reconstruction in the literature (e.g., Sonka et al. (2008), Chapters 11 and 12). 3-D image reconstruction is technically challenging, and the reconstructed 3-D image often contains substantial error in estimating the true 3-D image. Thanks to the rapid progress in image acquisition techniques, we can now acquire 3-D images directly in certain applications, including MRI and fMRI. However, observed 3-D images often contain noise due to hardware imperfections and other reasons. Noise removal is important for the reliability of subsequent image analyses, which is the focus of this chapter.

As discussed in Section 1.2, most image denoising methods are for analyzing 2-D images. For a detailed discussion, see Chapter 1 and related references. Some 2-D image denoising procedures have been generalized for analyzing 3-D images. For instance, 3-D image denoising based on minimization of Total Variation (TV) is popular in the computer sciences literature, e.g., Keeling (2003), Wang and Zhou (2006). The TV approach is first suggested by Rudin et al. (1992) for denoising 2-D images, and one particular algorithm to accomplish that method is due to Chambolle (2004). Several MATLAB programs based on Chambolle's algorithm have been developed recently for denoising 3-D images, e.g., Getreuer (2007). MATLAB programs for 3-D image denoising using anisotropic diffusion are also available, e.g., Lopes (2007). Other 3-D image denoising procedures include the ones based on 3-D wavelet transformations (Weickert et al., 1998; Hostalkova et al., 2007; Coupe et al., 2008a; Woiselle et al., 2008), non-local means (Coupe et al., 2008b), distance-weighted Wiener filtering (Lu et al., 2001), and so forth.

As discussed before, besides noise removal, another important requirement for image denoising procedures is that they should preserve important image structures, such as edges and major edge features. We notice that the structure of a typical

Figure 3.1: Different structures of edge surfaces.

3-D image is much more complicated than that of a 2-D image. For instance, edge locations are surfaces in 3-D cases and they usually have much more complicated structure than the edge curves in 2-D cases. Besides planar parts, an edge surface can have complicated structures, such as intersections of two or more edge surfaces, pyramids, pointed corners, and so forth. See Figure 3.1 for a demonstration. Most existing image denoising procedures mentioned above have certain ability in preserving edges at places where the curvature of the edge surfaces is small (i.e., the planar parts). At places where the curvature of the edge surfaces is large (e.g., intersections of two or more edge surfaces, pyramids, and pointed corners), however, the edge features would be blurred or rounded by them. This phenomenon is similar to its 2-D counterpart. In 3-D images also, major edge features are an important part of the image under study, because they often represent major characteristics of the image objects and are easier to capture our visual attention, compared to those relatively planar parts of the edge surfaces. Therefore, they should be well preserved during image denoising. So, in case of 3-D images also, a good image denoising procedure should preserve not only the planar parts of the edge surfaces but also the major edge features, although the latter goal is much more challenging than the former.

In this Chapter, we propose a 3-D image denoising procedure which can preserve edges and major edge features well. Our procedure consists of three major steps, briefly outlined below. First, edge pixels are detected in the whole design space by an edge detector. Second, in a neighborhood of a given pixel, the underlying edge surface is estimated from the detected edge pixels by an algorithm that takes into account three possible scenarios of the edge surface. Finally, observed image intensities located on the same side of the estimated edge surface, as the given pixel, are averaged by the local linear kernel smoothing procedure for estimating the true image intensity at the given pixel.

The remaining part of the article is organized as follows. Our proposed 3-D image denoising procedure is described in detail in Section 3.2. Some of its statistical properties are discussed in Section 3.3. In Section 3.4, we present some numerical examples to evaluate its numerical performance. Proofs of the theoretical results are provided in the Appendix.

3.2 Proposed methodology

We present our proposed methodology in three parts. Section 3.2.1 describes a 3-D edge detection procedure based on local linear kernel (LLK) smoothing. Local approximation of edge surfaces and local image denoising are described in Section 3.2.2. Data driven parameter selection is discussed in Section 3.2.3.

3.2.1 3-D edge detection by LLK smoothing

As discussed in Section 3.1, the first step of the proposed 3-D image denoising procedure is to detect edge pixels of a 3-D image using an edge detector. In the literature, there are many edge detectors for analyzing 2-D images (Canny, 1986; Qiu and Yandell, 1997; Sun and Qiu, 2007). These edge detectors can generally be extended for analyzing 3-D images. Theoretically speaking, any reasonable 3-D edge detector can be used in the first step of our 3-D image denoising procedure. Next, we introduce a 3-D edge detector based on LLK smoothing, which can be regarded as a modification of the 2-D edge detector in Qiu and Yandell (1997).

Assume that a 3-D image follows the regression model

$$\xi_{ijk} = f(x_i, y_j, z_k) + \varepsilon_{ijk}, \quad \text{for } i, j, k = 1, 2, \ldots, n, \tag{3.1}$$

where $\{(x_i, y_j, z_k) = (i/n, j/n, k/n),\ i, j, k = 1, 2, \ldots, n\}$ are the equally spaced design points (or voxels) in the design space $\Omega = [0,1] \times [0,1] \times [0,1]$, $\{\varepsilon_{ijk}\}$ are i.i.d. random errors with mean 0 and unknown variance σ^2, $f(x, y, z)$ is an unknown regression function denoting the image intensity function, and $N = n^3$ is the sample size. We further assume that there exists a partition $\{\Lambda_l,\ l = 1, 2, \ldots, s\}$ of the design space Ω such that: (i) each Λ_l is a connected region in Ω; (ii) $\bigcup_{l=1}^{s} \Lambda_l = \Omega$; (iii) $f(x, y, z)$ is continuous in $\Lambda_l \backslash \partial\Lambda_l$, for $l = 1, 2, \ldots, s$, where $\partial\Lambda_l$ is the boundary point set of Λ_l, and (iv) there exist at most finite number of line segments $\{\ell_l,\ l = 1, 2, \ldots, s^*\}$ in $[\bigcup_{i=1}^{s} \partial\Lambda_i] \bigcap \Omega$ such that for each line segment ℓ_l there are Λ_{l_1},$\Lambda_{l_2} \in \{\Lambda_l,\ l = 1, 2, \ldots, s\}$ satisfying $\ell_l \subseteq \partial\Lambda_{l_1} \bigcap \partial\Lambda_{l_2}$ and for any $(x^*, y^*, z^*) \in \ell_l$

$$\lim_{(x,y,z)\to(x^*,y^*,z^*),(x,y,z)\in\Lambda_{l_1}} f(x, y, z) = \lim_{(x,y,z)\to(x^*,y^*,z^*),(x,y,z)\in\Lambda_{l_2}} f(x, y, z).$$

Then, we call $D := [\bigcup_{l=1}^{s} \partial\Lambda_l] \bigcap \Omega$ the *jump location surfaces (JLSs)* of $f(x, y, z)$. Obviously, JLSs describe the places where f has jumps. So, they also describe edge surfaces.

To detect edge pixels at a given point $(x, y, z) \in \Omega$, let us consider its spherical neighborhood

$$O^*(x, y, z) = \{(u, v, w) : (u, v, w) \in \Omega, \sqrt{(u - x)^2 + (v - y)^2 + (w - z)^2} \leq h_n^*\},$$

where h_n^* is a bandwidth parameter. In $O^*(x, y, z)$, a 3-D plane is fitted using the local linear kernel (LLK) smoothing as follows

$$\min_{a,b,c,d} \sum_{i,j,k=1}^{n} \{\xi_{i,j,k} - [a + b(x_i - x) + c(y_j - y) + d(z_k - z)]\}^2 \times K\left(\frac{x_i - x}{h_n^*}, \frac{y_j - y}{h_n^*}, \frac{z_k - z}{h_n^*}\right), \qquad (3.2)$$

where K is a 3-D density kernel function defined in a unit ball. The solution to (a, b, c, d) of the minimization problem (3.2) is denoted as $(\widehat{a}(x, y, z), \widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))$. Then $\widehat{a}(x, y, z)$ is the LLK estimator of $f(x, y, z)$, and $(\widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))$ are LLK estimators of $(f'_x(x, y, z), f'_y(x, y, z), f'_z(x, y, z))$. It is not difficult to check that

$$\begin{pmatrix} \widehat{a}(x,y,z) \\ \widehat{b}(x,y,z) \\ \widehat{c}(x,y,z) \\ \widehat{d}(x,y,z) \end{pmatrix} = \begin{pmatrix} w_{000} & w_{100} & w_{010} & w_{001} \\ w_{100} & w_{200} & w_{110} & w_{101} \\ w_{010} & w_{110} & w_{020} & w_{011} \\ w_{001} & w_{101} & w_{011} & w_{002} \end{pmatrix}^{-1} \begin{pmatrix} \sum \xi_{ijk} K_{ijk} \\ \sum \xi_{ijk}(x_i - x)K_{ijk} \\ \sum \xi_{ijk}(y_j - y)K_{ijk} \\ \sum \xi_{ijk}(z_k - z)K_{ijk} \end{pmatrix}, \qquad (3.3)$$

where $K_{ijk} = K(\frac{x_i - x}{h_n^*}, \frac{y_j - y}{h_n^*}, \frac{z_k - z}{h_n^*})$, $w_{i_1 i_2 i_3} = \sum (x_i - x)^{i_1} (y_j - y)^{i_2} (z_k - z)^{i_3} K_{ijk}$, for $i_1, i_2, i_3 = 0, 1, 2$, and $\sum$ is the sum over all design points.

The estimated gradient vector $\widehat{\boldsymbol{\beta}}(x, y, z) = (\widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))^T$ provides an estimate of the direction that the underlying regression function f increases the fastest. So, if its magnitude is larger, f would be steeper around (x, y, z), and it is more likely that (x, y, z) is an edge pixel. However, when f is steep but continuous in $O^*(x, y, z)$, $\widehat{\boldsymbol{\beta}}(x, y, z)$ can also have a relatively large magnitude. To remove this slope effect, we consider two neighboring design points (x_{N1}, y_{N1}, z_{N1}) and (x_{N2}, y_{N2}, z_{N2}) along the direction of $\widehat{\boldsymbol{\beta}}(x, y, z)$. Their spherical neighborhoods of size h_n^* do not overlap with $O^*(x, y, z)$, but are adjacent to $O^*(x, y, z)$ on either side. Intuitively, if (x, y, z) is on a JLS, then (x_{N1}, y_{N1}, z_{N1}) and (x_{N2}, y_{N2}, z_{N2}) would be on two different sides of the JLS. Based on this intuition, we define the following jump detection criterion:

$$\delta(x,y,z) = \min\{\|\widehat{\boldsymbol{\beta}}(x,y,z) - \widehat{\boldsymbol{\beta}}_{N_1}(x,y,z)\|, \|\widehat{\boldsymbol{\beta}}(x,y,z) - \widehat{\boldsymbol{\beta}}_{N_2}(x,y,z)\|\},$$

where $\widehat{\boldsymbol{\beta}}_{N_1}(x,y,z)$ and $\widehat{\boldsymbol{\beta}}_{N_2}(x,y,z)$ are the estimated gradients in the neighborhoods of (x_{N1}, y_{N1}, z_{N1}) and (x_{N2}, y_{N2}, z_{N2}), respectively, and $\|\cdot\|$ is the Euclidean norm. If there is no jump in the three neighborhoods, then $\widehat{\boldsymbol{\beta}}(x,y,z)$, $\widehat{\boldsymbol{\beta}}_{N_1}(x,y,z)$ and $\widehat{\boldsymbol{\beta}}_{N_2}(x,y,z)$ should be close to each other. Hence, $\delta(x,y,z)$ is small. On the other hand, if (x,y,z) is on a JLS, then $\delta(x,y,z)$ would be relatively large, due to the jump. Therefore, $\delta(x,y,z)$ can be used for detecting jumps. The point (x,y,z) is then detected as a jump point if

$$\delta(x,y,z) > u_n, \tag{3.4}$$

where u_n is a threshold parameter.

3.2.2 Local approximation to the underlying jump location surfaces

As discussed in Section 3.1, JLSs of a 3-D image could have a number of different structures (cf., Figure 3.1). To preserve the major edge structures, we consider estimation of the underlying JLS in the following neighborhood of a given point $(x,y,z) \in \Omega$:

$$O(x,y,z) = \{(u,v,w) : (u,v,w) \in \Omega, \sqrt{(u-x)^2 + (v-y)^2 + (w-z)^2} \leq h_n\},$$

where h_n is a bandwidth that could be different from h_n^* used in edge detection (3.2). To this end, the following three cases are considered. (i) The underlying JLS in $O(x,y,z)$ is planar, and can be well approximated by a local plane. (ii) The JLS in $O(x,y,z)$ contains a ridge or valley, and can be well approximated by two

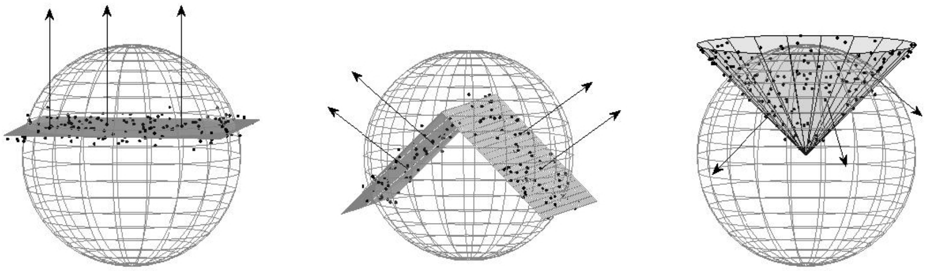

Figure 3.2: Three basic edge structures used for approximating the underlying JLS in a neighborhood of a given design point. In each plot, the dots denote the detected edge pixels, shaded surface denotes approximation of surface and the arrows point to its normal direction.

crossing half-planes. (iii) The JLS contains a pointed part in $O(x, y, z)$, and can be approximated reasonably well by a cone. See Figure 3.2 for a demonstration of these three basic edge structures. In reality, the JLSs may have more complicated structures than the ones considered here. For instance, they may contain a pyramid, saddle points, or an elliptically pointed part. However, with the three basic structures, the major edge features can be preserved well, which is supported by the numerical examples presented in Section 3.4. In the next four parts, we describe our proposals to estimate the three basic edge structures from the detected edge pixels, and to choose one of them for estimating the underlying JLS.

Approximation to the JLS in $O(x, y, z)$ by a local plane

In cases when the underlying JLS is planar in $O(x, y, z)$ (cf., the left panel of Figure 3.1), we suggest approximating it by a local plane constructed as follows.

(i) The plane passes the center (c_x, c_y, c_z) of the detected edge pixels in $O(x, y, z)$.

(ii) Its normal direction is determined by the eigenvector of the largest eigenvalue of

$$G = \frac{1}{m}\Psi\Psi^T, \tag{3.5}$$

where $\Psi = (\widehat{\boldsymbol{\beta}}_1^*, \widehat{\boldsymbol{\beta}}_2^*, \ldots, \widehat{\boldsymbol{\beta}}_m^*)$, and $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$ denote estimated gradient directions (with unit lengths) at all m detected edge pixels in $O(x, y, z)$.

See the left panel of Figure 3.2 for a demonstration, where the dots denote the detected edge pixels, the arrows denote the direction determined by the eigenvector of the largest eigenvalue of G, and the shaded plane denotes the plane for approximation. Lemma 3 in the Appendix shows that the approximation plane converges almost surely to the true tangent plane of f at (x, y, z) under some regularity conditions.

Approximation to the JLS in $O(x, y, z)$ by two crossing half-planes

In cases when the underlying JLS has a ridge or valley in $O(x, y, z)$ (cf., the middle panel of Figure 3.1), we suggest approximating it by two crossing half-planes as follows.

(i) Calculate the eigenvector corresponding to the smallest eigenvalue of G, which should be a good estimate of the direction of the ridge/valley contained in the underlying JLS, because the JLS has the smallest variation along that direction.

(ii) Determine the plane P that passes (c_x, c_y, c_z) along the estimated ridge/valley direction obtained in step (i) and along $\overline{\boldsymbol{\beta}}^*$ as well, where $\overline{\boldsymbol{\beta}}^*$ is the average of $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$.

(iii) Divide all detected edge pixels in $O(x, y, z)$ into two groups as follows. We first determine the direction that is orthogonal to the plane P, and then determine the two groups of the detected edge pixels based on the signs of their inner products with the orthogonal direction. Those detected edge pixels with non-negative inner products belong to one group, and those with negative inner products belong to the other group.

(iv) For each group of the detected edge pixels, determine a plane that passes the center of that group of pixels in the direction orthogonal to the averaged gradient direction within that group of pixels.

(v) The two crossing half-planes are then obtained from the two planes computed in step (iv). The subspace of $O(x, y, z)$ formed by them should contain (c_x, c_y, c_z).

Approximation to the JLS in $O(x, y, z)$ by a cone

In cases when the underlying JLS contains a pointed part in $O(x, y, z)$ (cf., the right panel of Figure 3.1), we suggest approximating it by a cone, which can be uniquely determined by specifying its central axis, its vertex, and the half cone angle (i.e., the angle between the central axis and a straight line on the lateral surface), as follows.

(i) The direction of the central axis of the cone is estimated by the direction $(1, d_2, d_3)^T$ that minimizes the sample variance of its inner products with $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$. Since the angle between the central axis of the cone and the normal direction at any point on the lateral surface of the cone is a constant, this estimate should be reasonable to use. Simple calculations show that $d_2 = (\Psi_{23}\Psi_{13} - \Psi_{33}\Psi_{12})/(\Psi_{22}\Psi_{33} - \Psi_{23}^2)$ and $d_3 = (\Psi_{12}\Psi_{23} - \Psi_{22}\Psi_{13})/(\Psi_{22}\Psi_{33} - \Psi_{23}^2)$, where $\Psi_{j_1 j_2}$ is the sample covariance of the (j_1, j_2)th components of $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$, for $j_1, j_2 = 1, 2, 3$.

(ii) The half cone angle is estimated by the complement of the averaged angle between the direction of the central axis specified in step (i) and $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$, which is denoted as $\widehat{\theta}$.

(iii) To specify the location of the central axis, let us consider a sphere $\widetilde{O}(x, y, z)$ that is centered at (x, y, z) and has radius $\widetilde{h}_n > h_n$. The plane $\widetilde{P}$ passing (c_x, c_y, c_z) and having the normal direction $(1, d_2, d_3)^T$ would divide $\widetilde{O}(x, y, z)$ into two parts. Centers of the detected edge pixels in these parts are then calculated, and the one closer to $\widetilde{P}$ is denoted as (c_x^*, c_y^*, c_z^*). Then, the line passing (c_x^*, c_y^*, c_z^*) along the direction $(1, d_2, d_3)^T$ is used as the central axis of the cone. In all numerical examples presented in Section 3.4, we choose $\widetilde{h}_n = 3h_n$.

(iv) The vertex location (v_x, v_y, v_z) of the cone is estimated by minimizing the orthogonal distance between the cone and the detected edge pixels in $O(x, y, z)$. After some algebraic manipulations, the estimator can be calculated by

$$(\widehat{v}_x, \widehat{v}_y, \widehat{v}_z) = (c_x^* + \overline{\beta}_1^* t, c_y^* + \overline{\beta}_2^* t, c_z^* + \overline{\beta}_3^* t),$$

where $\overline{\beta}_1^*$, $\overline{\beta}_2^*$, and $\overline{\beta}_3^*$ are three components of $\overline{\boldsymbol{\beta}}^*$, t is one of $[\sum d_l t_l / (\tan \widehat{\theta} || \overline{\boldsymbol{\beta}}^* ||) + \sum t_l^2] / \sum t_l$ and $[\sum d_l t_l / (\tan \widehat{\theta} || \overline{\boldsymbol{\beta}}^* ||) - \sum t_l^2] / (-\sum t_l)$ that minimizes $\sum (d_l \cos \widehat{\theta} - |t - t_l| || \overline{\boldsymbol{\beta}}^* || \sin \widehat{\theta})^2$,

$$\begin{aligned} t_l &= \frac{\overline{\beta}_1^*(x_l^* - c_x) + \overline{\beta}_2^*(y_l^* - c_y) + \overline{\beta}_3^*(z_l^* - c_z)}{||\overline{\boldsymbol{\beta}}^*||^2}, \\ d_l &= \sqrt{(x_l^* - c_x^* - \overline{\beta}_1^* t_l)^2 + (y_l^* - c_y^* - \overline{\beta}_2^* t_l)^2 + (z_l^* - c_z^* - \overline{\beta}_3^* t_l)^2}, \end{aligned}$$

(x_l^*, y_l^*, z_l^*) denotes the l-th detected edge pixel in $O(x, y, z)$, and $\sum$ is over all detected edge pixels in $O(x, y, z)$.

Selection of the local surface

In the previous three parts, we have discussed how to approximate the underlying JLS in $O(x, y, z)$ using one of the three basic surfaces shown in Figure 3.2. In practice, we need to choose one of the three surfaces based on observations in $O(x, y, z)$ for estimating $f(x, y, z)$. To this end, various model selection criteria, including the Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and so forth, have been considered. For instance, by the BIC, the fitted surface minimizing the following BIC criterion should be chosen:

$$BIC(x, y, z) = m \log\left(\frac{RSS(x, y, z)}{m}\right) + k \log(m), \tag{3.6}$$

where $RSS(x, y, z)$ denotes the sum of squares of the orthogonal distances from the individual detected edge pixels to the fitted surface in question, m is the number of detected edge pixels in $O(x, y, z)$, and k is the total number of parameters in the model of the surface. For the local plane, the two crossing half-planes, and the cone, their numbers of parameters are 3, 6, and 7, respectively. By the AIC, the second term on the right side of (3.6) should be replaced by $2k$.

In our numerical studies, we tried both AIC and BIC methods, and found that results with the BIC method are usually better in terms of the mean squared error (MSE) of the estimated f. This might be consistent with the findings in Nishii (1984) that the probability of choosing the true model by the BIC tends to 1 when the sample size increases, as long as the true model is included in the candidate set. In the current problem, most part of the true JLSs should be well approximated by one of the three basic surfaces considered. For this reason, we use the BIC in all numerical examples presented in Section 3.4.

3.2.3 Proposed 3-D image denoising procedure

After the underlying JLS is approximated by a local surface in the neighborhood $O(x, y, z)$, $O(x, y, z)$ can be divided into two parts by the approximation surface, denoted as $O_1(x, y, z)$ and $O_2(x, y, z)$. Without loss of generality, we assume that (x, y, z) is contained in $O_1(x, y, z)$. Then, $f(x, y, z)$ can be estimated by the solution to a, denoted as $\widehat{f}(x,y,z)$, of the following minimization problem:

$$\min_{a,b,c,d} \sum_{(x_i,y_j,z_k)\in O_1(x,y,z)} \left\{\xi_{ijk} - [a + b(x_i - x) + c(y_j - y) + d(z_k - z)]\right\}^2 \times K\left(\frac{x_i - x}{h_n}, \frac{y_j - y}{h_n}, \frac{z_k - z}{h_n}\right). \tag{3.7}$$

From (3.2) and (3.7), $\widehat{f}(x,y,z)$ is a weighted average of the observations whose design points are located on the same side of the approximation surface in $O(x,y,z)$ as the given point (x,y,z). Intuitively, as long as the the approximation surface estimates the underlying JLS well, $\widehat{f}(x,y,z)$ should preserve edges and major edge features well.

For a real image, there are regions where f is smooth. In these regions, the number of detected edge pixels should be small. Also, in such regions, a relatively larger bandwidth is preferred to increase the noise removal ability of the procedure. So, before estimating f using (3.7), we suggest counting the number of detected edge pixels in $O(x,y,z)$. If the number is large so that a potential JLS is likely in $O(x,y,z)$ (e.g., larger than or equal to $(nh_n)^2/4$), then estimate f using (3.7). Otherwise, consider a larger spherical neighborhood $\breve{O}(x,y,z)$ with bandwidth $\breve{h}_n > h_n$. In $\breve{O}(x,y,z)$, if the number of detected edge pixels is still smaller than $(n\breve{h}_n)^2/4$, then $f(x,y,z)$ can be estimated simply by the conventional LLK estimator constructed from all observations in $\breve{O}(x,y,z)$. If the number of detected edge pixels in $\breve{O}(x,y,z)$ is larger than or equal to $(n\breve{h}_n)^2/4$, $f(x,y,z)$ is estimated by the conventional LLK estimator constructed in $O(x,y,z)$. By this modification, procedure (3.7) is used only when the number of detected edge pixels in $O(x,y,z)$ is relatively large. To do so, there are at least two benefits. One is that much computation is saved, because the conventional LLK estimator is much easier to compute, compared to the one-sided estimator obtained by (3.7). The other benefit is that the estimated f would be more efficient, because it is constructed from all observations in $O(x,y,z)$ or $\breve{O}(x,y,z)$, instead of from part of observations in $O(x, y, z)$, in cases when the number of detected edge pixels in $O(x, y, z)$ is small. Considering the fact that most points in the design space are continuity points of f, these benefits are substantial. Regarding $\breve{h}_n$, based on our numerical experience, we can simply choose $\breve{h}_n = 1.75h_n$. In such cases, the numerical results are satisfactory in all examples considered in Section 3.4. Now we can summarize the proposed 3-D image denoising procedure as follows.

Proposed 3-D image denoising procedure

- Detect edge pixels by the procedure described in Section 3.2.1.

- For a given pixel (x, y, z), count the number of detected edge pixels in $O(x, y, z)$. If this number is smaller than $(nh_n)^2/4$, then consider a larger spherical neighborhood $\breve{O}(x, y, z)$ with bandwidth $\breve{h}_n$. If the number of detected edge pixels in $\breve{O}(x, y, z)$ is still smaller than $(n\breve{h}_n)^2/4$, then define $\widehat{f}(x, y, z)$ to be the conventional LLK estimate in $\breve{O}(x, y, z)$. Otherwise, define $\widehat{f}(x, y, z)$ to be the conventional LLK estimate in $O(x, y, z)$. Continue the denoising procedure for the next pixel.

- If the number of detected edge pixels in $O(x, y, z)$ is larger than or equal to $(nh_n)^2/4$, then define $\widehat{f}(x, y, z)$ by (3.7), after the underlying JLS in $O(x, y, z)$ is approximated by one of the three basic local surfaces, as discussed in Section 3.2. Continue the denoising procedure for the next pixel.

3.2.4 Selection of procedure parameters

In the proposed 3-D image denoising procedure, there are three parameters h_n^*, u_n, and h_n (cf., expressions (3.2), (3.4), (3.7)). They should be chosen properly because performance of the proposed image denoising procedure would depend on their values.

To choose u_n, we notice that

$$\begin{aligned}
& P(\delta(x,y,z) > u_n) \\
\leq\ & P(\|\widehat{\boldsymbol{\beta}}(x,y,z) - \widehat{\boldsymbol{\beta}}_{N1}(x,y,z)\| > u_n) \\
=\ & P\Big((\widehat{b}(x,y,z) - \widehat{b}_{N1}(x,y,z))^2 + (\widehat{c}(x,y,z) - \widehat{c}_{N1}(x,y,z))^2 + \\
& (\widehat{d}(x,y,z) - \widehat{d}_{N1}(x,y,z))^2 > u_n^2\Big) \\
=\ & E\Big\{P\Big((\widehat{b}(x,y,z) - \widehat{b}_{N1}(x,y,z))^2 + (\widehat{c}(x,y,z) - \widehat{c}_{N1}(x,y,z))^2 + \\
& (\widehat{d}(x,y,z) - \widehat{d}_{N1}(x,y,z))^2 > u_n^2 \,\Big|\, \widehat{b}(x,y,z), \widehat{c}(x,y,z), \widehat{d}(x,y,z)\Big)\Big\}.
\end{aligned}$$

For fixed $\widehat{b}(x,y,z)$, $\widehat{c}(x,y,z)$, and $\widehat{d}(x,y,z)$, $((\widehat{b}(x,y,z) - \widehat{b}_{N1}(x,y,z))^2 + (\widehat{c}(x,y,z) - \widehat{c}_{N1}(x,y,z))^2 + (\widehat{d}(x,y,z) - \widehat{d}_{N1}(x,y,z))^2)/\sigma_{N1}^2$ approximately follows the χ_3^2 distribution, under the assumption that there are no jumps in $O^*(x,y,z) \bigcup O^*(x_{N1}, y_{N1}, z_{N1}) \bigcup O^*(x_{N2}, y_{N2}, z_{N2})$, where $\sigma_{N1}^2 = \mathrm{Var}(\widehat{b}_{N1}(x,y,z))$. From expressions in (3.3), we have

$$\sigma_{N1}^2 = \sigma^2 \frac{\sum (x_i - x_{N1})^2 K_{N1}^2}{\{(\sum (x_i - x_{N1}) K_{N1}\}^2},$$

where $K_{N1} = K\left(\frac{x_i - x_{N_1}}{h_n^*}, \frac{y_j - y_{N_1}}{h_n^*}, \frac{z_k - z_{N_1}}{h_n^*}\right)$. Therefore, a natural choice for u_n is

$$u_n = \widehat{\sigma} \sqrt{\frac{\chi_{3,\alpha_n}^2 \sum (x_i - x_{N1})^2 K_{N1}^2}{(\sum (x_i - x_{N1})^2 K_{N1})^2}}, \tag{3.8}$$

where χ_{3,α_n}^2 is the $1-\alpha_n$ quantile of the χ_3^2 distribution and $\widehat{\sigma}$ is a consistent estimator of σ. For simplicity, we can define $\widehat{\sigma}$ to be the residual mean squares of the conventional LLK estimator of f defined in $O^*(x,y,z)$, and α_n can be specified beforehand to be a small number, say, $\alpha_n = 0.001$.

To choose h_n^* and h_n, we suggest using the cross-validation (CV) procedure with the following CV score:

$$CV(h_n^*, h_n) = \frac{1}{n^3} \sum_{i,j,k=1}^{n} \left(\xi_{ijk} - \widehat{f}_{-i,-j,-k}(x_i, y_j, z_k) \right)^2, \tag{3.9}$$

where $\widehat{f}_{-i,-j,-k}(x_i, y_j, z_k)$ is the estimate of $f(x_i, y_j, z_k)$ when we do not include the (i, j, k)-th pixel (x_i, y_j, z_k) in all subsequent steps of the proposed image denoising procedure after edge detection. Then, h_n^* and h_n are chosen to be the minimizer of $CV(h_n^*, h_n)$. Note that, when computing $\widehat{f}_{-i,-j,-k}(x_i, y_j, z_k)$, the (i, j, k)-th pixel is still used in edge detection, which is for simplifying computation. By (3.9), we only need to detect edges once in the whole design space. Based on our numerical experience, it would not change the value of $\widehat{f}_{-i,-j,-k}(x_i, y_j, z_k)$ much to include the (i, j, k)-th pixel or not in edge detection. In the statistical literature, there are some alternative methods for choosing smoothing parameters like h_n^* and h_n, including the Mallow's C_p, plug-in algorithms, bootstrap, and so forth (Marron, 1988; Loader, 1999; Hall and Robinson, 2009). The CV method is used here mainly because of its simplicity, which is especially relevant for 3-D image analysis where computation is generally extensive.

3.3 Some statistical properties

In this section, we discuss some statistical properties of the estimated image intensity function $\widehat{f}$. In our description, a point (x, y, z) on the JLSs is called a *singular point* if one of the following conditions is satisfied. (i) There exists some constant $\nu > 0$ such that, for any $0 < \tilde{\nu} < \nu$, the spherical neighborhood of (x, y, z) with diameter $\tilde{\nu}$ is divided into more than two connected regions by the JLSs. (ii) There exists a direction along which the JLSs do not have the one-sided directional tangent line at

(x, y, z). (iii) The jump magnitude of f at (x, y, z) is 0. (iv) f does not have a tangent plane at (x, y, z), and (x, y, z) is neither on a ridge/valley of the JLSs nor a vertex of a circular cone (cf., the middle and right panels of Figure 3.2). All other points on the JLSs are called nonsingular points. Also, we define

$$\begin{aligned}
\Omega_\epsilon &= [\epsilon, 1-\epsilon] \times [\epsilon, 1-\epsilon] \times [\epsilon, 1-\epsilon], \\
J_\epsilon &= \{(x, y, z) : (x, y, z) \in \Omega, d_E((x, y, z), (x^*, y^*, z^*)) \leq \epsilon, \text{ for some } (x^*, y^*, z^*) \\
&\quad \in D\}, \\
S_\epsilon &= \{(x, y, z) : (x, y, z) \in \Omega, d_E((x, y, z), (x^*, y^*, z^*)) \leq \epsilon, \text{ for a singular point} \\
&\quad (x^*, y^*, z^*) \in D\}, \\
\Omega_{\bar{J},\epsilon} &= \Omega_\epsilon \backslash J_\epsilon, \\
\Omega_{\bar{S},\epsilon} &= \Omega_\epsilon \backslash S_\epsilon,
\end{aligned}$$

where ϵ is a small positive constant, d_E denotes the Euclidean distance, and D denotes the set of points on the JLSs. Let $\widehat{D}_n = \{(x_i, y_j, z_k) : \delta(x_i, y_j, z_k) > u_n\}$ be the set of all detected edge pixels. Then, we have

Theorem 3.3.1: Assume that f has continuous first order partial derivatives over $(0,1) \times (0,1) \times (0,1)$ except on the JLSs, and its first order partial derivatives have one-sided limits at nonsingular points of the JLSs on both sides of a given JLS; $h_n^* = o(1)$, $1/(nh_n^*) = o(1)$, and $\log^2(n)/n^3(h_n^*)^6 = O(1)$; $E|\varepsilon_{111}^4| < \infty$; the kernel function K is a Lipschitz-1 continuous, isotropic, trivariate density function; and α_n is chosen such that (i) $\alpha_n = o(1)$, (ii) $(nh_n^*)^{7/2}/\{n^2\sqrt{-\log \alpha_n}\} = o(1)$ and (iii) $\sqrt{-\log \alpha_n/(nh_n^*)^3} = o(1)$. Then, for any small constant $\epsilon > 0$, $d_H(D \bigcap \Omega_{\bar{S},\epsilon}, \widehat{D}_n \bigcap \Omega_{\bar{S},\epsilon}) = O(h_n^*)$ a.s., where $d_H(A, B)$ denotes the Hausdorff distance between two point sets A and B, defined by

$$d_H(A,B) = \max\left\{ \sup_{(x,y,z)\in A} \inf_{(x',y',z')\in B} d_E((x,y,z),(x',y',z')), \right.$$
$$\left. \sup_{(x,y,z)\in B} \inf_{(x',y',z')\in A} d_E((x,y,z),(x',y',z')) \right\} \blacksquare$$

Theorem 3.3.2: Besides the conditions in Theorem 3.3.1, if we further assume that the bandwidth h_n satisfies the conditions that $h_n = o(1)$, $1/(nh_n) = o(1)$ and $h_n^*/h_n^3 = o(1)$, then $\|\widehat{f} - f\|_{\Omega_{\bar{J},h_n}} = \max_{(x,y,z)\in\Omega_{\bar{J},h_n}} |\widehat{f}(x,y,z) - f(x,y,z)| = O(h_n^2)$ a.s. For any small constant $\epsilon > 0$, when $(x,y,z) \in J_{h_n}\backslash S_\epsilon$, $\widetilde{h}_n = o(1)$ and $\frac{h_n}{\widetilde{h}_n} = o(1)$, we have $\widehat{f}(x,y,z) = f(x,y,z) + O(h_n)$ a.s. $\blacksquare$

Theorem 3.3.1 establishes the almost sure consistency of the detected edge pixels. If we choose $h_n^* = O(n^{-1/4})$, $\alpha_n = O(e^{-n^2})$, and all other conditions in the theorem are satisfied, then we have $d_H(D \bigcap D_\epsilon,\ \widehat{D}_n \bigcap D_\epsilon) = O(n^{-1/4})$ a.s. Theorem 3.3.2 says that the estimated image intensity function is uniformly strong consistent in the whole design space excluding a small region around the design border and the true JLSs. At a given point around the true JLSs, it is pointwise strong consistent, as long as the point in question is a small distance away from any singular points of the JLSs. Proofs of the above two theorems are provide in the appendix.

3.4 Numerical examples

In this section, we present some numerical results regarding the performance of the proposed 3-D image denoising procedure (denoted as NEW), in comparison with four existing methods that are commonly used in practice, including an anisotropic diffusion method (denoted as AD, cf., Lopes (2007)), a method based on total variation minimization (denoted as TV, cf., Rudin et al. (1992), Getreuer (2007)), a method based on optimized non-local means (denoted as ONLM, cf., Coupe et al. (2008b)), and the conventional local median filtering method (denoted as MED). For the TV method, the code by Getreuer (2007) interprets a 3-D image as a 2-D image with vector-valued image intensities. Here, we have modified it by minimizing a TV criterion constructed from a 3-D image directly for 3-D image denoising. In this method, there is a regularization parameter to choose. The AD method is accomplished by an iterative algorithm that contains two parameters: the diffusion parameter κ and the number of iterations of the algorithm. The ONLM method has two bandwidth parameters to choose. The median filtering method denoises the image by using the sample median of the observations in a spherical window to estimate the true image intensity at the central pixel of the window. It has one parameter to choose, which is the bandwidth of the spherical window. Our proposed denoising method NEW has three parameters h_n^*, u_n and h_n to choose.

First, we consider the following two true image intensity functions

$$\begin{aligned} f_1(x,y,z) &= -(x-0.5)^2-(y-0.5)^2-(z-0.5)^2+ \\ & I\left(\max(I(|x-0.5|\leq 0.25)I(|y-0.5|\leq 0.25)I(|z-0.5|\leq 0.25),\right. \\ & \left. I((x-0.5)^2+(y-0.5)^2\leq 0.15^2)I(|z-0.5|\leq 0.35))=1\right), \end{aligned}$$

$$\begin{aligned} f_2(x,y,z) &= \frac{1}{4}\sin(2\pi(x+y+z)+1)+\frac{1}{4}+I\left(\max(I((x-0.5)^2+(y-0.5)^2\right. \\ & \leq \frac{1}{4}(z-0.5)^2)I(z\leq 0.5)I(z\geq 0.2), \\ & I((x-0.5)^2+(y-0.5)^2+(z-0.5)^2\leq 0.4^2) \\ & \left. I((x-0.5)^2+(y-0.5)^2+(z-0.5)^2>0.2^2)I(z<0.45))=1\right). \end{aligned}$$

Their edge surfaces are shown in the two plots of Figure 3.3. Basically, f_1 contains a cube with a solid cylinder in the middle, and its background and foreground are

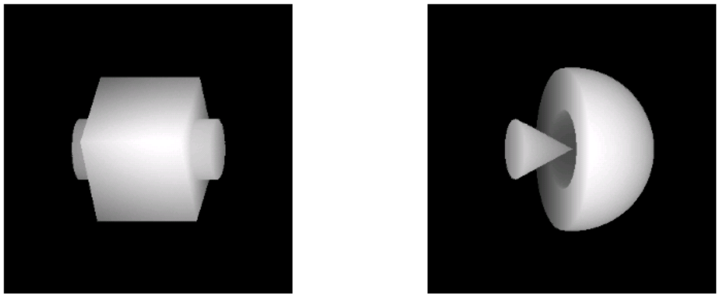

Figure 3.3: The left and right panels show the jump location surfaces of f_1 and f_2, respectively.

variable but smooth. From the left panel of Figure 3.3, we can see that the JLSs of f_1 contain planes, intersections of two planes, intersections of three planes, smooth non-linear surfaces, and intersections of a plane and a smooth non-linear surface. From the right panel of Figure 3, we can see that f_2 contains a half hollow hemisphere and a cone. The two objects are separated, although they are close to each other. The foreground of f_2 has a fixed intensity level, and its background has a variable intensity level. Therefore, jump sizes are different at different places of the JLSs.

We then apply the five image denoising procedures to the 3-D noisy images generated from model (3.1) when f equals f_1 or f_2, $n = 64$ or 128 (denoting two different image resolutions), $\varepsilon_{111} \sim N(0, \sigma^2)$ with $\sigma = 0.1$, 0.2 or 0.3 (representing low, medium and high noise levels). Because procedures ONLM, TV, AD and MED do not have corresponding data-driven parameter selection algorithms yet, to make a fair comparison, we search their procedure parameters by minimizing the MISE value estimated by the sample mean of

$$ISE(\widehat{f}, f) = \frac{1}{n^3} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \left(\widehat{f}(x_i, y_j, z_k) - f(x_i, y_j, z_k) \right)^2$$

over 100 replications, for each combination of f, σ and n, where $\widehat{f}$ denotes the denoised image of the related denoising method.

The MISE criterion provides us a measure of the overall performance of a denoising procedure. But, it can not tell us how well the JLSs are preserved by the denoising procedure. Hall and Qiu (2007) defined a measure of the jump size (JS) of a 2-D image. A discretized version of its 3-D generalization can be written as follows. For the true image intensity function f, its JS can be measured by

$$JS(f) = \frac{1}{(n-2)^3} \sum_{i=2}^{n-1} \sum_{j=2}^{n-1} \sum_{k=2}^{n-1} |f(x_i', y_j', z_k') - f(x_i'', y_j'', z_k'')|,$$

where (x_i', y_j', z_k') and (x_i'', y_j'', z_k'') are two pixels on two different sides of the pixel (x_i, y_j, z_k) along the $x-$, $y-$, $z-$direction, or a diagonal direction that is closest to the gradient direction of f at (x_i, y_j, z_k). Obviously, if (x_i, y_j, z_k) is an edge pixel, then $|f(x_i', y_j', z_k') - f(x_i'', y_j'', z_k'')|$ is close to the jump magnitude of f at (x_i, y_j, z_k). Otherwise, $|f(x_i', y_j', z_k') - f(x_i'', y_j'', z_k'')|$ is close to 0. So, $JS(f)$ is a reasonable measure of the accumulative jump magnitude of f along the JLSs. For a denoised image $\widehat{f}$, we can compute $JS(\widehat{f})$ similarly, by using the estimated gradient directions of f. Then, $EP(\widehat{f}) = (JS(f) - JS(\widehat{f}))/JS(f)$ would be a reasonable measure of edge-preservation (EP) for the denoising method in question. In the numerical examples, after we denoise images by the NEW method and all other competing methods, we calculate this measure using the neighborhood size 0.0234 when $n = 64$ and 0.0117 when $n = 128$ to calculate the gradient directions. For MRI images we used it to be 0.0117. The choices of this neighborhood size seem reasonable if we compare them with the choice of h_n^* in different images listed in Tables 3.1–3.3 below.

The numerical results for f_1 and f_2 are presented in Tables 3.1 and 3.2, respectively. From Table 3.1, it can be seen that the proposed method NEW is uniformly better than the competing methods TV, AD, and MED in both MISE and EP, although the MISE values of the methods NEW and TV are almost the same in cases when $\sigma = 0.1$. Compared to the method ONLM, the method NEW performs better in terms of MISE in cases when $\sigma = 0.2$ and 0.3, and performs similarly in terms of

MISE in cases when $\sigma = 0.1$. In terms of EP, it seems that ONLM is better when σ is small (i.e., $\sigma = 0.1$ when $n = 64$, and $\sigma = 0.1$ or 0.2 when $n = 128$). Similar conclusions can be made from the results presented in Table 3.2, except that the methods NEW, TV and ONLM perform similarly in this example in terms of MISE, but the method NEW is much better than the other two methods in terms of EP.

One realization of f_1 is generated from model (3.1) with $n = 128$, $\varepsilon_{111} \sim N(0, \sigma^2)$, and $\sigma = 0.2$. Three cross sections of this realization at $z = 0.5$, $z = 0.1875$, and $x = 0.4688$ are shown in the first column of Figure 3.4. The corresponding cross sections of the denoised images by TV, AD, ONLM, MED and NEW are shown in the 2–6 columns. In the denoising methods, their parameters are chosen to be the ones shown in the part with $n = 128$ and $\sigma = 0.2$ of Table 3.1. Similarly, three cross sections of a realization of f_2 at $x = 0.5$, $x = 0.8125$, and $x = 0.6562$, from model (3.1) with $n = 128$, $\varepsilon_{111} \sim N(0, \sigma^2)$, and $\sigma = 0.2$, along with the cross sections of the denoised images of the five methods, are shown in Figure 3.5. From the two figures, it can be seen that TV, ONLM and MED tend to blur the edges at some places, the denoised image by AD seems to contain some scattered noise, and the denoised image by NEW preserves the edges well and removes noise well. To better see the results, for the three slices of the denoised images shown in Figure 3.5, we present the differences between them and the corresponding slices of the true image in Figure 3.6. From the images in this figure, we can see that the method NEW is indeed better in preserving edges than the four competing methods, which is consistent with the results of EP shown in Table 3.2.

Next, we consider a magnetic resonance image (MRI) of a human brain with $128 \times 128 \times 52$ pixels. Its image intensity levels range from 0 to 809. I.i.d. noise from the distribution $N(0, \sigma^2)$ is added to the image, where σ is chosen to be 40, 70, or 100. The parameters of the four denoising procedures are chosen in the same way as those in the examples of Tables 3.1 and 3.2. The results are shown in left-side

Table 3.1: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the measure of edge-preservation EP and its standard error, and the third line presents the searched procedure parameter values. This table considers the case when $f = f_1$.

	n=64			n=128		
Method	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
TV	.0003 (.0000) .1011 (.0023) 11.0	.0009 (.0000) .1909 (.0037) 6.0	.0018 (.0000) .2480 (.0050) 4.5	.0001 (.0000) .1053 (.0013) 11.0	.0004 (.0000) .1930 (.0019) 6.0	.0008 (.0000) .2737 (.0024) 4.0
AD	.0006 (.0000) -.0310 (.0022) .18 , 6	.0035 (.0000) -.0457 (.0050) .4 , 4	.0082 (.0001) .1109 (.0089) .6 , 4	.0003 (.0000) -.0246 (.0009) .18 , 8	.0020 (.0000) -.0632 (.0021) .38 , 5	.0048 (.0000) .1719 (.0040) .6 , 5
ONLM	.0003 (.0000) -.0082 (.0025) 10 , 2	.0012 (.0000) .0408 (.0054) 10 , 2	.0032 (.0001) .1191 (.0075) 10 , 2	.0001 (.0000) -.0070 (.0013) 10 , 2	.0006 (.0000) .0140 (.0026) 10 , 2	.0015 (.0000) .0421 (.0035) 10 , 2
MED	.0022 (.0000) .1615 (.0023) .0313	.0033 (.0000) .2757 (.0039) .0359	.0055 (.0001) .3214 (.0051) .0359	.0008 (.0000) .1579 (.0010) .0189	.0019 (.0000) .3789 (.0014) .0250	.0033 (.0000) .4667 (.0016) .0250
NEW	.0003 (.0000) .0224 (.0025) .023 , 13 , .037	.0008 (.0000) .0369 (.0049) .023 , 22 , .050	.0012 (.0001) .0464 (.0075) .028 , 22 , .053	.0001 (.0000) .0199 (.0015) .012 , 24 , .025	.0003 (.0000) .0251 (.0027) .012 , 44 , .028	.0005 (.0000) .0277 (.0038) .014 , 44 , .028

Table 3.2: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the measure of edge-preservation EP and its standard error, and the third line presents the searched procedure parameter values. This table considers the case when $f = f_2$.

	n=64			n=128		
Method	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
TV	.0006 (.0000) .1648 (.0025) 13.0	.0015 (.0000) .2375 (.0046) 7.5	.0023 (.0000) .3563 (.0058) 4.5	.0003 (.0000) .1795 (.0013) 12.0	.0006 (.0000) .3077 (.0022) 6.0	.0010 (.0000) .3504 (.0029) 4.5
AD	.0007 (.0000) .0153 (.0031) .22 , 4	.0038 (.0000) .1111 (.0067) .4 , 4	.0087 (.0001) .1418 (.0066) .75 , 3	.0004 (.0000) .0085 (.0017) .18 , 7	.0022 (.0000) .1709 (.0041) .4 , 5	.0052 (.0000) .1880 (.0033) .72 , 4
ONLM	.0006 (.0000) .1264 (.0029) 3 , 2	.0014 (.0000) .2146 (.0048) 4 , 2	.0024 (.0000) .2529 (.0067) 5 , 2	.0003 (.0000) .1795 (.0017) 5 , 3	.0006 (.0000) .2308 (.0031) 5 , 3	.0009 (.0000) .2564 (.0045) 6 , 3
MED	.0011 (.0000) .1918 (.0028) .0313	.0026 (.0000) .3793 (.0042) .0406	.0038 (.0000) .5019 (.0049) .0500	.0006 (.0000) .2735 (.0015) .0203	.0014 (.0000) .4530 (.0021) .0250	.0022 (.0000) .5726 (.0020) .0297
NEW	.0006 (.0000) .0625 (.0035) .023 , 9 , .031	.0013 (.0000) .1016 (.0066) .023 , 16 , .038	.0019 (.0000) .2007 (.0099) .028 , 18 , .053	.0002 (.0000) .0477 (.0022) .012 , 20 , .019	.0006 (.0000) .0873 (.0045) .014 , 22 , .028	.0008 (.0000) .1568 (.0065) .014 , 34 , .033

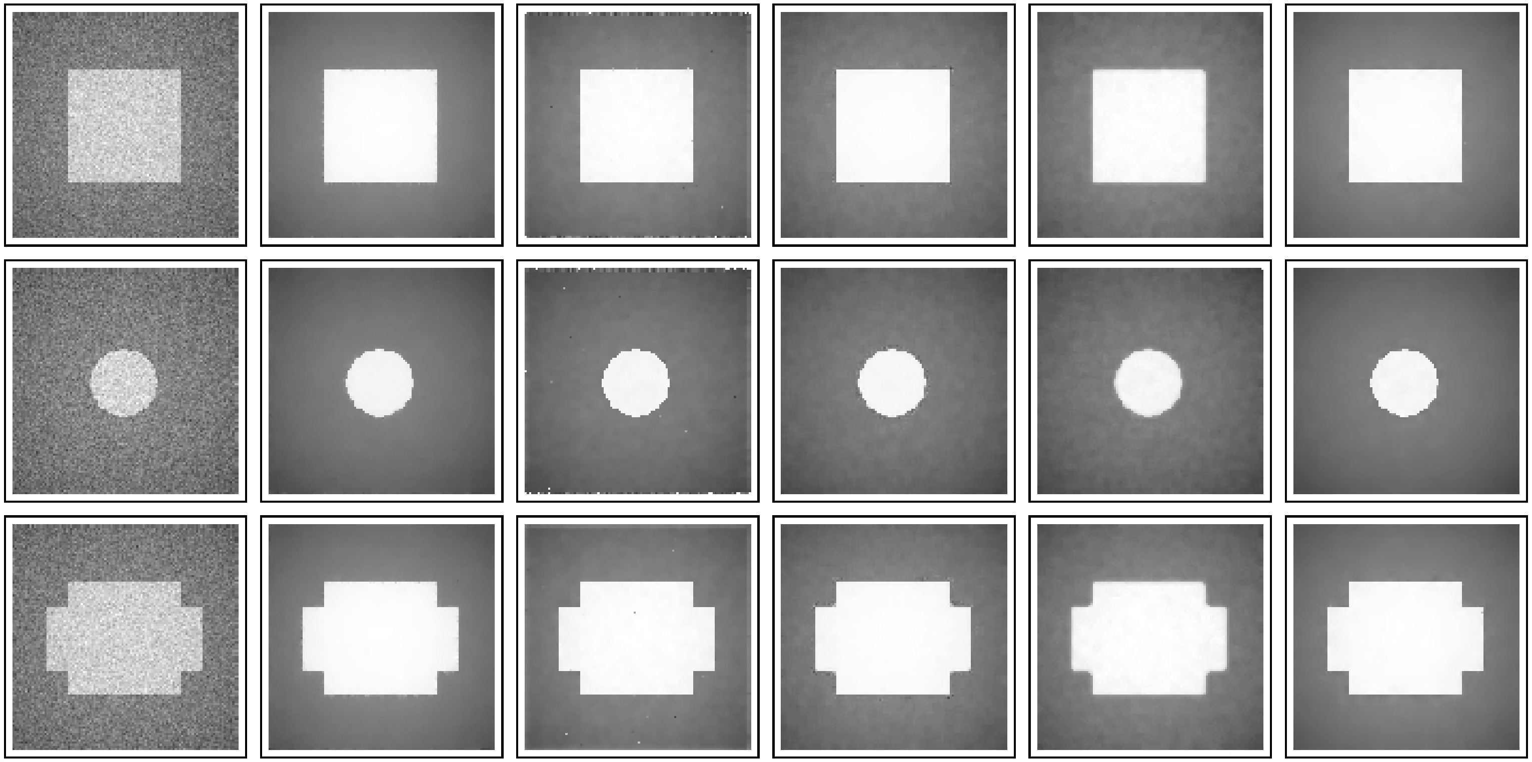

Figure 3.4: The first column presents three cross sections of an image generated from model (3.1) with $f = f_1$, $n = 128$, $\varepsilon_{111} \sim N(0, \sigma^2)$, and $\sigma = 0.2$. Columns 2–6 show denoised images by TV, AD, ONLM, MED and NEW, respectively, when their procedure parameters are chosen to be the corresponding ones listed in Table 3.1.

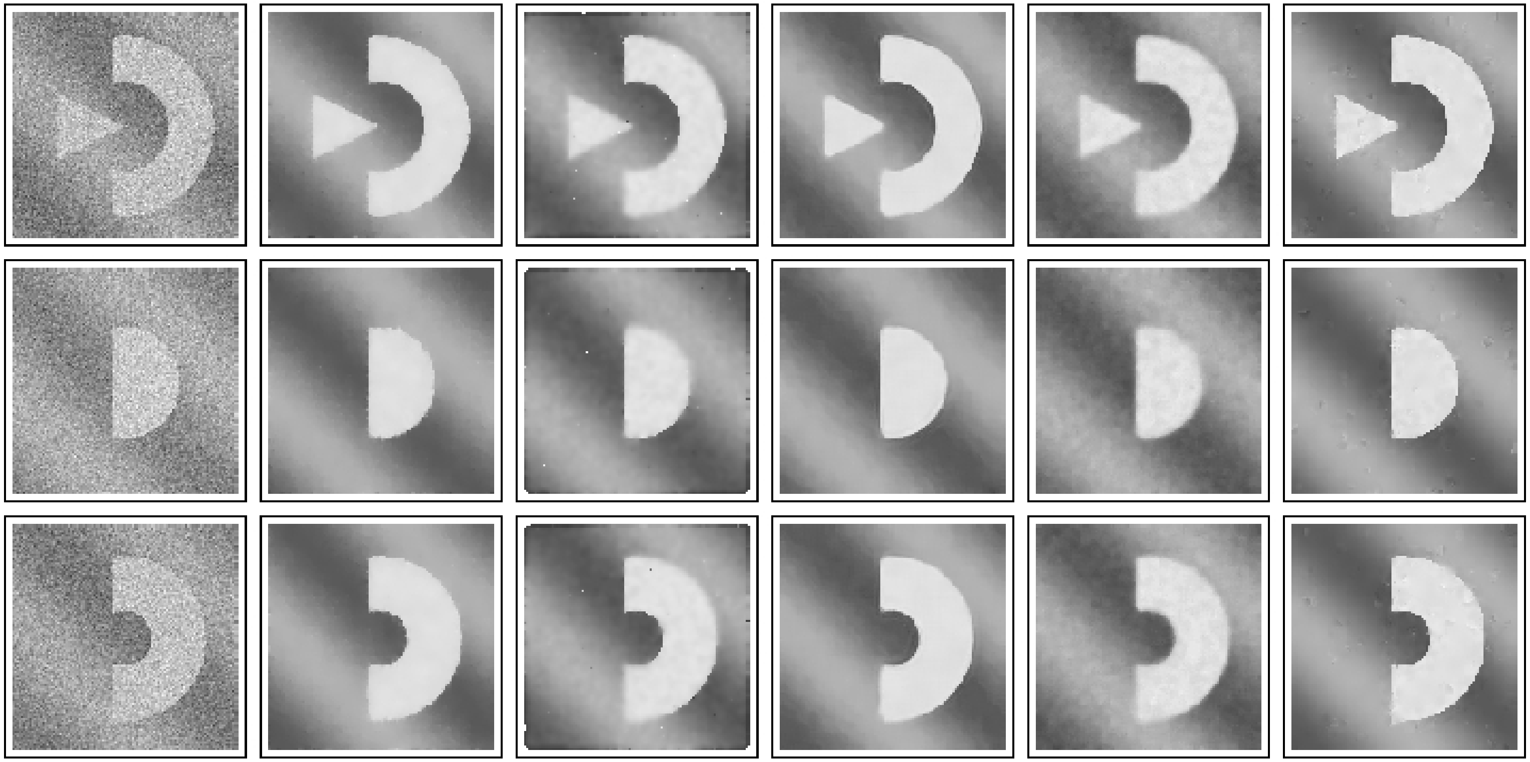

Figure 3.5: The first column presents three cross sections of an image generated from model (3.1) with $f = f_2$, $n = 128$, $\varepsilon_{111} \sim N(0, \sigma^2)$, and $\sigma = 0.2$. Columns 2–6 show denoised images by TV, AD, ONLM, MED and NEW, respectively, when their procedure parameters are chosen to be the corresponding ones listed in Table 3.2.

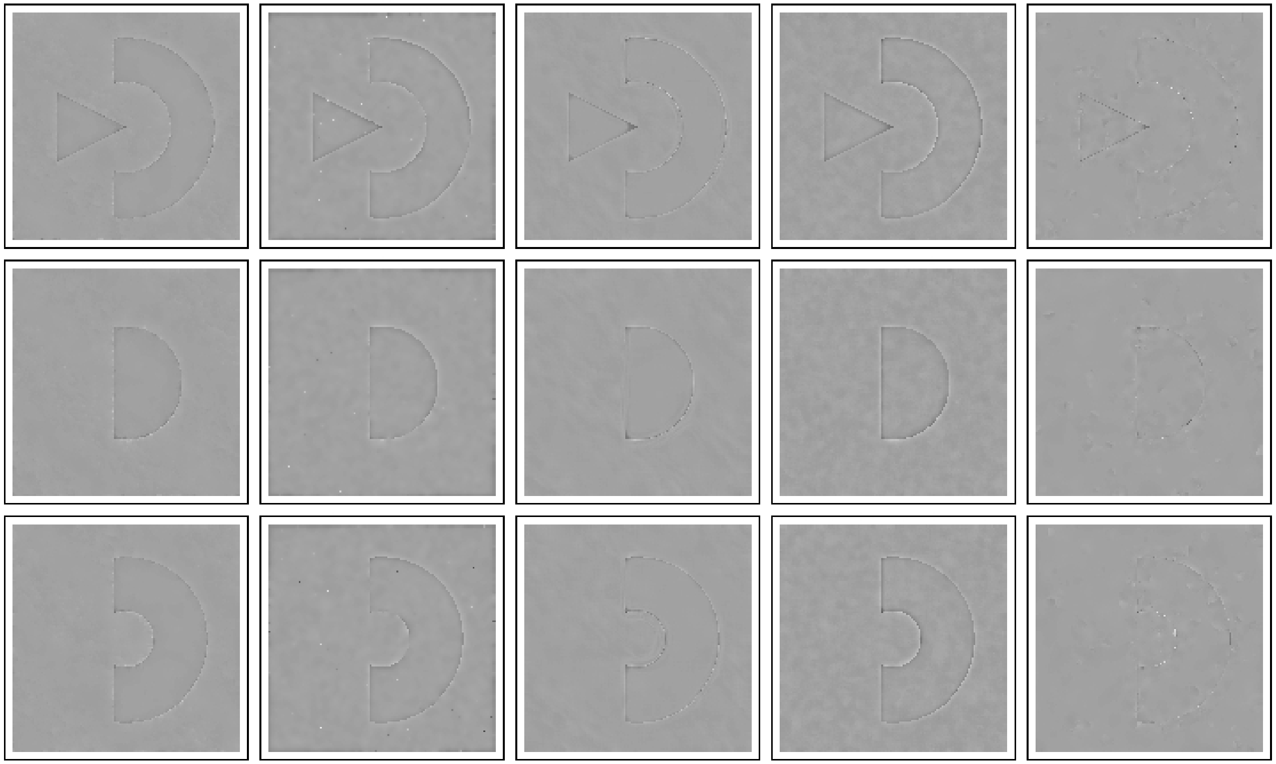

Figure 3.6: The five columns show the differences between the three slices of the denoised images shown in Figure 3.5 and the corresponding slices of the true image, for methods TV, AD, ONLM, MED and NEW, respectively.

portion of Table 3.3 and in Figure 3.7. From Table 3.3, we can see that the proposed procedure NEW outperforms all three competing methods in all cases in terms of both the MISE and the edge preservation measure EP. In Figure 3.7, the first two rows present two cross-sections of a noisy version of the 3-D MRI image when $\sigma = 40$ (1st column), and their denoised versions by procedures TV, AD, ONLM, MED, and NEW (columns 2–6). The third and fourth rows and the fifth and sixth rows present the corresponding results when $\sigma = 70$ and 100, respectively. In order to see the detail better, in the seventh row, we zoom out the upper-middle portion of the images in the fourth row. From the images in this figure, it seems that procedure NEW preserves edges better than the competing methods, which is consistent with results in Table 3.3.

We also consider the case when the noise level is different at different places of the image. In practice, the noise level is often higher in the foreground (i.e., the central part of the image with the image objects) and lower in the background. To mimic this situation, the following variable noise level is considered:

$$\sigma(x, y, z) = 25 + 50 \exp\left[-\frac{(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2}{4}\right].$$

The denoising methods are executed in the same way as before, and the results are presented in the right-side portion of Table 3.3 and in Figure 3.8. From the table and the figure, we can see that the proposed procedure NEW performs relatively well in this case too.

In Section 3.2.4, a cross-validation (CV) procedure is proposed for choosing the parameters of the proposed procedure NEW. Next, we apply NEW to all examples discussed above in the same as described, except that u_n is chosen by (3.8) with $\alpha_n = 0.001$ and h_n^* and h_n are chosen by minimizing the CV score defined in (3.9). The results based on 100 replications are presented in Table 3.4. From this table and Tables 3.1–3.3, we can see that (i) results of NEW with its parameters selected by

Table 3.3: In each entry, the first line presents the estimated MISE value from 100 simulations and the corresponding standard error of ISE (in parenthesis), the second line presents the measure of edge-preservation EP and the corresponding standard error, and the third line presents the searched procedure parameter values.

	Gaussian Noise			Variable Noise
Method	$\sigma = 40$	$\sigma = 70$	$\sigma = 100$	variable σ
TV	374.0 (1.0)	737.5 (2.2)	1063.6 (3.4)	756.4 (2.5)
	.3021 (.0015)	.4788 (.0020)	.5350 (.0025)	.4647 (.0021)
	.05	.025	.018	.025
AD	456.2 (1.3)	971.8 (3.0)	1525.0 (4.8)	997.5 (3.1)
	.2574 (.0016)	.4042 (.0021)	.4440 (.0018)	.4223 (.0022)
	110 , 2	190 , 2	700 , 1	200 , 2
ONLM	403.8 (1.2)	764.5 (2.8)	1129.1 (4.6)	785.2 (2.8)
	.2264 (.0013)	.3158 (.0019)	.3494 (.0024)	.3289 (.0019)
	6 , 1	10 , 1	16 , 1	11 , 1
MED	642.7 (1.6)	1065.9 (2.7)	1472.1 (4.1)	1093.9 (2.8)
	.2120 (.0013)	.4241 (.0018)	.4727 (.0023)	.4200 (.0019)
	.0078	.0117	.0141	.0117
NEW	371.3 (1.1)	669.7 (2.2)	949.5 (3.9)	700.0 (2.5)
	.1899 (.0017)	.2529 (.0026)	.3128 (.0038)	.2574 (.0029)
	.012,8000,.012	.012,12000,.014	.012,16000,.017	.012,12500,.014

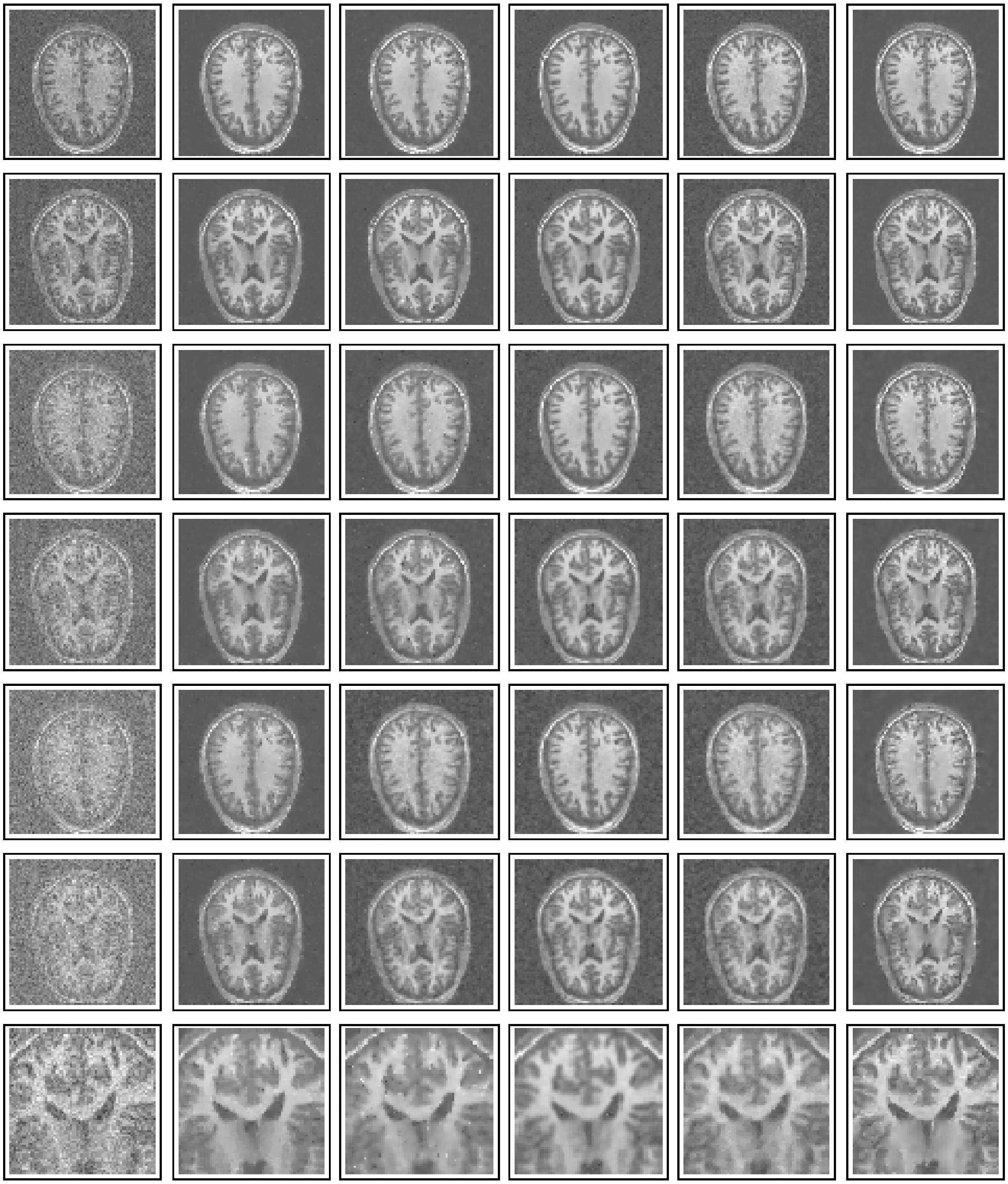

Figure 3.7: The first two rows present two cross-sections of a noisy version of the 3-D MRI image when $\sigma = 40$ (1st column), and their denoised versions by procedures TV, AD, ONLM, MED, and NEW (columns 2–6). The third and fourth rows and the fifth and sixth rows present the corresponding results when $\sigma = 70$ and 100, respectively. Images in the seventh row zoom out the upper-middle portion of the images in the fourth row. Procedure parameters are chosen to be the ones listed in Table 3.3.

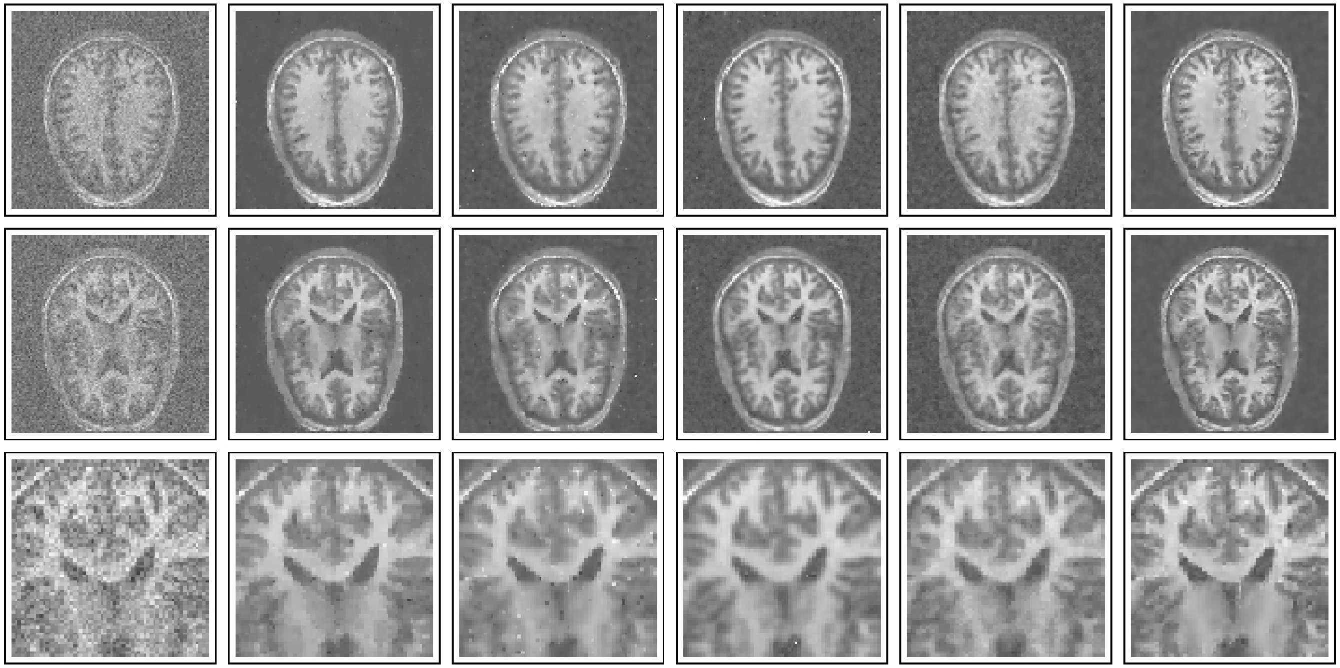

Figure 3.8: The first two rows present two cross-sections of a noisy version of the 3-D MRI image when $\sigma = 25 + 50\exp\{-[(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2]/4\}$ (1st column), and their denoised versions by procedures TV, AD, ONLM, MED, and NEW (columns 2–6). Images in the third row zoom out the upper-middle portion of the images in the second row. Procedure parameters are chosen to be the ones listed in Table 3.3.

CV are close to results of NEW with its parameters selected optimally by minimizing the estimated MISE in all cases, and (ii) when its parameters chosen by CV, NEW still outperforms TV and MED in all cases, outperforms AD when $\sigma = 0.2$ or 0.3, and is comparable with AD when $\sigma = 0.1$. The three cross sections of its denoised image in the case of Figure 3.4 are shown in the first column of Figure 3.9, the three cross sections of its denoised image in the case of Figure 3.5 are shown in the second column, and the cross sections of its denoised images corresponding to rows 2, 4 and 6 of Figure 3.7 are shown in the third column. Comparing Figure 3.9 with Figures 3.4, 3.5, and 3.7, we can see that the denoised images of NEW when its parameters are chosen by CV are indeed similar to its denoised images when its parameters are chosen optimally by minimizing the estimated MISE.

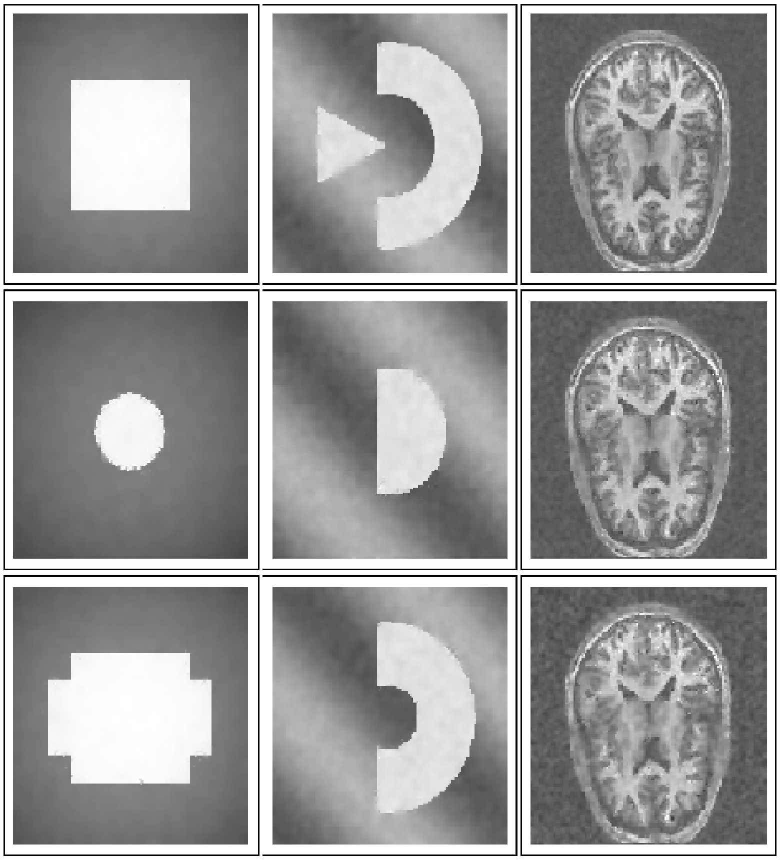

Figure 3.9: The first column shows the cross sections of the denoised image of NEW in the case of Figure 3.4, the second column shows the cross sections of the denoised image of NEW in the case of Figure 3.5, and the third column shows the cross sections of the denoised images of NEW corresponding to rows 2, 4 and 6 of Figure 3.7. In this example, procedure parameters of NEW are chosen by CV.

Table 3.4: Simulation results of the procedure NEW when its parameters are chosen by (3.8) with $\alpha_n = 0.001$ and by minimizing the CV score defined in (3.9).

	n=64			n=128		
	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
f_1	.0003 (.0000)	.0008 (.0000)	.0013 (.0001)	.0002 (.0000)	.0004 (.0000)	.0006 (.0000)
	-.0096 (.0028)	-.0136 (.0055)	.0132 (.0073)	.0129 (.0016)	.0124 (.0027)	.0161 (.0037)
	.034 , .041	.034 , .038	.034 , .056	.014 , .027	.014 , .027	.014 , .028
f_2	.0006 (.0000)	.0013 (.0000)	.0020 (.0000)	.0002 (.0000)	.0007 (.0000)	.0008 (.0000)
	.0650 (.0036)	.1332 (.0061)	.1956 (.0103)	.0479 (.0024)	.0812 (.0049)	.1723 (.0065)
	.023 , .034	.031 , .047	.031 , .053	.016 , .020	.012 , .025	.017 , .031
	$\sigma = 40$	$\sigma = 70$	$\sigma = 100$		variable σ	
MRI	435.9 (1.3)	729.0 (2.2)	1024.6 (4.4)		763.6 (2.6)	
	.1752 (.0018)	.1921 (.0031)	.2727 (.0039)		.1877 (.0033)	
	.012 , .014	.012 , .014	.012 , .017		.012 , .014	

3.5 Generalization of the proposed 3-D procedure

In this section, we modify the proposed 3-D image denoising procedure in several aspects described below. First, approximating the JLS locally by one of many surface templates, not by just three, can possibly provide a substantial improvement. If we know beforehand that some surface structures are likely to be present in a 3-D image, then it should be included in the pool of surface templates, so that a better surface approximation can be obtained. Second, a simpler edge detection method and a simpler procedure for selecting surface templates are used in the modified method. Third, for better performance, algorithms for locally approximating the JLS are also generalized to some extent. Some other modifications have also been made, which will be discussed in detail below. In the remaining part of this section, 'the proposed 3-D image denoising procedure' would refer to the modified method.

3.5.1 Proposed modified method

Our modified 3-D image denoising procedure still consists of three major steps. First, edge voxels are detected using a 3-D edge detector that is constructed under the JRA framework. Second, in a neighborhood of a given voxel, the underlying edge surfaces are approximated by a surface template chosen from a pre-specified surface template family. Third, the true image intensity at the given voxel is estimated by a weighted average of the observed image intensities in the neighborhood whose voxels are located on the same side of the surface template as the given voxel. Details of these steps are described below.

Identification of Edge Voxels

At a given voxel $(x, y, z) \in \Omega$, to know whether it is an edge voxel, let us consider its spherical neighborhood $O^*(x, y, z)$ with radius h_n^*. In $O^*(x, y, z)$, we fit a 3-D plane using the local linear kernel (LLK) smoothing procedure (3.2). The solution to (a, b, c, d) of (3.2) is denoted as $(\widehat{a}(x, y, z), \widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))$, and their expressions are given in (3.3). Then, $\widehat{a}(x, y, z)$ is the LLK estimator of $f(x, y, z)$, and $(\widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))$ are LLK estimators of $(f'_x(x, y, z), f'_y(x, y, z), f'_z(x, y, z))$. In (3.2), a 3-D plane is fitted in the neighborhood $O^*(x, y, z)$, by a weighted least square procedure, for approximating the underlying image intensity function f around the given voxel (x, y, z). The amount of data involved is determined by the radius h_n^*. Intuitively, at places where the noise level is high, h_n^* should be chosen relatively large to remove more noise. At places where the curvature of the image intensity function is small (i.e., the image intensity surface is quite straight), h_n^* can be chosen relatively large as well. In practice, however, the noise and the true image intensity function are both unobservable. All we have are the observed image intensities $\{\xi_{ijk}\}$. A data-driven procedure for choosing h_n^* is discussed in a later part of this section.

The estimated gradient vector $\widehat{\boldsymbol{\beta}}(x, y, z) = (\widehat{b}(x, y, z), \widehat{c}(x, y, z), \widehat{d}(x, y, z))^T$ provides an estimate of the direction that f increases the fastest around (x, y, z). Let us consider a plane that passes (x, y, z) and is orthogonal to $\widehat{\boldsymbol{\beta}}(x, y, z)$. Then, this plan divides $O^*(x, y, z)$ into two halves $O_1^*(x, y, z)$ and $O_2^*(x, y, z)$, as demonstrated in Figure 3.10. In $O_1^*(x, y, z)$ and $O_2^*(x, y, z)$, we compute weighted averages of the observed image intensities, respectively, with the weights determined by K. The weighted averages are denoted as $\widehat{a}_1(x, y, z)$ and $\widehat{a}_2(x, y, z)$. Then, (x, y, z) is flagged as an edge voxel if

$$|\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)| > T_n, \tag{3.10}$$

where T_n is a threshold. In the case when there are no edge voxels in $O^*(x, y, z)$, it is derived in the appendix that $\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)$ is distributed approximately as $N(0, 4\widehat{\sigma}^2 \sum K^2_{h_n^*}(x_i, y_j, z_k)/[\sum K_{h_n^*}(x_i, y_j, z_k)]^2)$, where $\sum$ is over all voxels, and

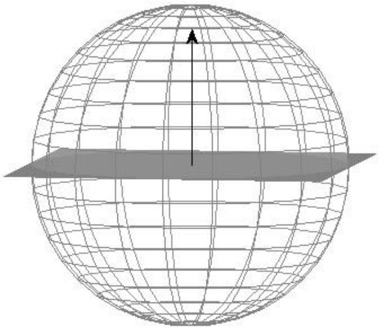

Figure 3.10: The spherical neighborhood $O^*(x,y,z)$ is divided into two halves $O_1^*(x,y,z)$ and $O_2^*(x,y,z)$ by a plane passing the center (x,y,z) and perpendicular to the estimated gradient $\widehat{\boldsymbol{\beta}}(x,y,z)$.

$\widehat{\sigma}^2 = \frac{1}{N}\sum_{i,j,k=1}^{n}(\xi_{ijk} - \widehat{a}(x_i,y_j,z_k))^2$. Therefore, a reasonable choice of T_n is

$$T_n = \frac{2Z_{1-\alpha_n/2}\widehat{\sigma}\sqrt{\sum K_{h_n^*}^2(x_i,y_j,z_k)}}{\sum K_{h_n^*}(x_i,y_j,z_k)}, \tag{3.11}$$

where $Z_{1-\alpha_n/2}$ is the $(1-\alpha_n/2)$th quantile of the standard normal distribution, and α_n is a pre-specified significance level. The value of α_n provides a measure of the chance that a non-edge voxel would be detected as an edge voxel. Its value should be chosen relatively small if the total number of voxels is large. Otherwise, there will be many detected false edge voxels. We have performed many numerical studies in which the number of voxels of each test image is $r \times 10^6$ with $r \in [0.5, 10]$ (e.g., images with $128 \times 128 \times 64$ or $256 \times 256 \times 128$ voxels), and find that results are reasonably good when we choose $\alpha_n \in [0.0001, 0.001]$. In such cases, each non-edge voxel has less than 0.1% chance to be detected as an edge voxel. In all numerical examples presented later in this Section, we use a single value $\alpha_n = 0.0002$ for simplicity, although the value of α_n can always be adjusted in different cases.

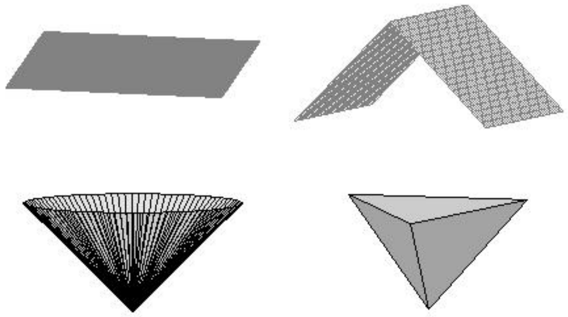

Figure 3.11: A family of four surface templates for approximating edge surfaces.

Template Surfaces and Edge Surface Approximation

As discussed in Section 3.1, there are mainly three different types of edge-structures in 3-D images (Figure 3.2). But, there are many variants of each type of edge features in real images. For instance, for pointed corners, instead of the corn-shape included in the family, we can also consider the shape of a triangular pyramid that is displayed in the lower-right panel of Figure 3.11. However, because the edge surface approximation is performed locally in a small neighborhood of a given voxel, consideration of this and other variants can only marginally improve the edge surface approximation, which is confirmed by numerical examples later in this Section. Further, consideration of extra surface templates would add a substantial amount of computation to the entire image denoising process. For these reasons, we recommend using the first three surface templates shown in Figure 3.11, and adding some extra surface templates to the family only in cases when we know that they are common edge features in a specific application.

Next, we describe our proposed algorithms for locally approximating the edge surfaces by the surface templates shown in Figure 3.11. At a given voxel (x, y, z), let us consider its spherical neighborhood $O(x, y, z)$ with radius h_n, where h_n could be different from h_n^* used in (3.2). Again, a data-driven procedure for choosing h_n will be discussed later in this Section. Let $\{\mathbf{s}_l, l = 1, 2, \ldots, m\}$ be the detected edge voxels in $O(x, y, z)$, $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$ be the corresponding estimated gradient directions (with unit lengths) at these edge voxels by (3.2), and

$$G = (w_1\widehat{\boldsymbol{\beta}}_1^*, w_2\widehat{\boldsymbol{\beta}}_2^*, \ldots, w_m\widehat{\boldsymbol{\beta}}_m^*)(w_1\widehat{\boldsymbol{\beta}}_1^*, w_2\widehat{\boldsymbol{\beta}}_2^*, \ldots, w_m\widehat{\boldsymbol{\beta}}_m^*)^T,$$

where $w_l = |\widehat{a}_1(\mathbf{s}_l) - \widehat{a}_2(\mathbf{s}_l)| - T_n$ which are all positive at detected edge voxels $\{\mathbf{s}_l, l = 1, 2, \ldots, m\}$ (3.10). Therefore, G is a weighted second moment from origin of $\{\widehat{\boldsymbol{\beta}}_l^*\}$, and the weights are determined by the significance of individual detected edge voxels. The eigenvalues of G are denoted as $\lambda_1 \leq \lambda_2 \leq \lambda_3$, and the corresponding eigenvectors with unit lengths are $\mathbf{v}_1, \mathbf{v}_2$, and $\mathbf{v}_3$. Then, if all $\widehat{\boldsymbol{\beta}}_l^*$s are the same (i.e., the underlying edge surface is a plane), G would have a rank of 1 and $\mathbf{v}_3$ would be the normal direction of the edge plane, and vice versa. Therefore, to approximate the edge surfaces in $O(x, y, z)$ by the first surface template shown in Figure 3.11, a reasonable solution is the plane that passes the weighted center $\bar{\mathbf{s}}$ of $\{\mathbf{s}_l\}$ with the weights $\{w_l\}$ and has the normal direction of $\mathbf{v}_3$.

To approximate the edge surfaces in $O(x, y, z)$ by the second surface template (i.e., the one shown in the upper-right panel of Figure 3.11), we proceed in two steps. First, $\{\mathbf{s}_l\}$ are divided into two groups by a plane that passes $\bar{\mathbf{s}}$ along the directions of $\mathbf{v}_1$ and $\overline{\boldsymbol{\beta}}^*$ where $\overline{\boldsymbol{\beta}}^*$ denotes the weighted average of $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$ with the weights $\{w_l, l = 1, 2, \ldots, m\}$. Second, with each group of the detected edge voxels, find an approximation plane passing their weighted averaged locations with its normal direction to be the weighted average of the estimated gradient directions at the detected edge voxels. Then, the two resulting half planes that cross each other and form a subspace in $O(x, y, z)$ containing $\bar{\mathbf{s}}$ are used for approximating the edge surfaces in $O(x, y, z)$.

To approximate the edge surfaces in $O(x, y, z)$ by a cone (i.e., the surface template shown in the lower-left panel of Figure 3.11), we need to specify its central axis, vertex position, and the angle between the central axis and any generatrix. Assume that the direction of the central axis is $\mathbf{d} = (1, d_2, d_3)^T$. Since the angle between this direction and the normal direction at any point on the cone is a constant, d_2 and d_3 can be estimated by minimizing the weighted sample variance of the inner products between $\mathbf{d}$ and $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$. Simple calculations show that

$$\begin{aligned} d_2 &= (\Psi_{23}\Psi_{13} - \Psi_{33}\Psi_{12}) / \left(\Psi_{22}\Psi_{33} - \Psi_{23}^2\right) \\ d_3 &= (\Psi_{12}\Psi_{23} - \Psi_{22}\Psi_{13}) / \left(\Psi_{22}\Psi_{33} - \Psi_{23}^2\right), \end{aligned}$$

where $\Psi_{j_1 j_2}$ is the (j_1, j_2)th component of the weighted sample covariance matrix of $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$, for $j_1, j_2 = 1, 2, 3$. To specify the location of the central axis, let us consider a sphere $\widetilde{O}(x, y, z)$ of radius $\widetilde{h}_n > h_n$. The plane $\widetilde{P}$ passing $\bar{\mathbf{s}}$ with the normal direction of $\mathbf{d}$ would divide $\widetilde{O}(x, y, z)$ into two parts. Weighted centers of the detected edge voxels in the two parts are then calculated, and the one closer to $\widetilde{P}$ is denoted as $\bar{\mathbf{s}}^* = (c_x^*, c_y^*, c_z^*)$. Then, the line passing $\bar{\mathbf{s}}^*$ along the direction $\mathbf{d}$ is defined to be the central axis of the cone. In this paper, we choose $\widetilde{h}_n = 3h_n$. As a matter of fact, selection of $\widetilde{h}_n$ does not have much effect on the final results. After the central axis is determined, the angle between the central axis and any generatrix can be easily estimated by the weighted average of the angles between $\mathbf{d}$ and $\{\widehat{\boldsymbol{\beta}}_l^*, l = 1, 2, \ldots, m\}$. The location of the vertex (v_x, v_y, v_z) of the cone can be estimated by minimizing the weighted orthogonal distance between the cone and the detected edge voxels in $O(x, y, z)$. After some algebraic manipulations, the estimator can be calculated by

$$(v_x, v_y, v_z) = (c_x^* + \overline{\beta}_1^*\rho, c_y^* + \overline{\beta}_2^*\rho, c_z^* + \overline{\beta}_3^*\rho),$$

where $\overline{\boldsymbol{\beta}}^* = (\overline{\beta}_1^*, \overline{\beta}_2^*, \overline{\beta}_3^*)^T$, ρ is one of

$$[\sum w_l d_l t_l / (\tan \widehat{\theta} ||\overline{\boldsymbol{\beta}}^*||) + \sum w_l t_l^2] / \sum w_l t_l$$

and

$$[\sum w_l d_l t_l / (\tan \widehat{\theta} ||\overline{\boldsymbol{\beta}}^*||) - \sum w_l t_l^2] / (- \sum w_l t_l)$$

that minimizes $\sum w_l (d_l \cos \widehat{\theta} - |t - t_l| ||\overline{\boldsymbol{\beta}}^*|| \sin \widehat{\theta})^2$, $t_l = (\overline{\boldsymbol{\beta}}^*)^T (\mathbf{s}_l - \overline{\mathbf{s}}) / \|\overline{\boldsymbol{\beta}}^*\|^2$, and $d_l = \|\mathbf{s}_l - \overline{\mathbf{s}}^* - t_l \overline{\boldsymbol{\beta}}^*\|$.

To approximate the edge surfaces in $O(x, y, z)$ by a triangular pyramid (i.e., the surface template shown in the lower-right panel of Figure 3.11), we need to estimate its vertical axis (i.e., the line passing through the vertex and the center of its triangular base), the angular orientation of the triangular base, the normal directions of the three lateral planes, and the position of the vertex. To estimate the vertical axis, let us consider the spherical neighborhood $\widetilde{O}(x, y, z)$ that is described in the previous paragraph. The weighted center of the detected edge voxels within $\widetilde{O}(x, y, z)$ is denoted as $\widetilde{\mathbf{s}} = (\widetilde{c}_x, \widetilde{c}_y, \widetilde{c}_z)$. Then, the vertical axis is estimated by the line passing $\widetilde{\mathbf{s}}$ in the direction of the weighted mean of the estimated gradient directions at all detected edge voxels in $\widetilde{O}(x, y, z)$. To determine the angular orientation of the triangular base, we consider a plane that passes $\widetilde{\mathbf{s}}$ and is orthogonal to the vertical axis. We also fix a half straight line $\widetilde{L}$ on the plane that starts from $\widetilde{\mathbf{s}}$. Another two half straight lines starting from $\widetilde{\mathbf{s}}$ on the plane can then be determined such that the three half lines are 120^0 apart and partition the plane into three equal parts. The detected edge voxels in $O(x, y, z)$ can then be divided into three groups based on the parts of the plane that their projections to the plane belong to. Intuitively, if the three groups of the detected edge voxels are located on the three lateral planes of the triangular pyramid, respectively, then the variability of the estimated gradient directions corresponding to each group of detected edge voxels should be small. Based on this intuition, the sample variances of the three groups of the estimated gradient directions are computed, respectively, and their weighted average is also computed, with the weights proportional to the numbers of detected edge voxels in the three groups. This weighted average is called the within-group variance hereafter. We then keep rotating $\widetilde{L}$ clockwise (or counterclockwise) within the plane with a step of ν^0 until the cumulative rotated degrees reaching 120^0. For each position of $\widetilde{L}$, the within-group variance is calculated. Then, the final position of $\widetilde{L}$ is chosen to be the one among all its positions considered with the smallest within-group variance. After the final position of $\widetilde{L}$ is determined, the normal directions of the three lateral sides of the pyramid can be estimated by the weighted means of the estimated gradient directions within the three groups of detected edge voxels, respectively. Finally, the vertex of the pyramid can be searched on the vertical axis to minimize the weighted sum of the squared orthogonal distances from the detected edge voxels in $O(x, y, z)$ to the respective lateral sides of the pyramid. To this end, we search for the pyramid vertex on both sides of the vertical axis within $O(x, y, z)$, starting from $\widetilde{\mathbf{s}}$ with a step of τ. In the algorithm just described, there are two parameters ν and τ. In our numerical study presented in later in this Section, we choose $\nu = 5$ and $\tau = 0.05/n$, because results with smaller values of these parameters would hardly change in all cases considered.

In practice, we need to choose one of the four estimated surface templates based on observed image intensities for approximating the edge surfaces in $O(x, y, z)$. For that purpose, one natural solution is to choose the estimated surface template with the smallest residual sum of squares (RSS), where RSS is defined to be the sum of squared orthogonal distances from $\{\mathbf{s}_l, l = 1, 2, \ldots, m\}$ to the estimated surface template. However, by this idea, the first surface template would never be selected because it is a special case of the second surface template and consequently its RSS value will never be the smallest one. To overcome this difficulty, we suggest the following two-step algorithm. Let RSS_1, RSS_2, RSS_3, and RSS_4 be the RSS values of the four estimated surface templates, respectively. Then,

(i) the third template is selected if RSS_3 is the smallest one among RSS_1, RSS_2, RSS_3, and RSS_4, and the fourth template is selected if RSS_4 is the smallest RSS value;

(ii) otherwise, the first template is selected if

$$\begin{aligned} F(x,y,z) &= \frac{(RSS_1(x,y,z) - RSS_2(x,y,z))/3}{RSS_2(x,y,z)/(m-6)} \\ &\leq \chi^2_{3,1-\tilde{\alpha}}, \end{aligned}$$

and the second template is selected if $F(x,y,z) > \chi^2_{3,1-\tilde{\alpha}}$, where $\chi^2_{3,1-\tilde{\alpha}}$ is the $(1-\tilde{\alpha})$th quantile of the χ^2_3 distribution.

In step (ii), we have used the statistical result that $F(x,y,z)$ is distributed asymptotically as χ^2_3 when m increases Seber (1977), which is true here because the first template is a special case of the second template and the second template has three more parameters than the first template. For the value of $\tilde{\alpha}$, if it is chosen smaller, then the second template would have a smaller chance to be selected. Based on our numerical experience, results are reasonably good if we choose $\tilde{\alpha} = 0.01$. For simplicity, this value of $\tilde{\alpha}$ is used in all numerical examples presented in this paper, although its value can always be adjusted in different cases. Note that, if we decide to use the first three surface templates only in our image denoising procedure, then step (i) should be changed to

(i)' the third template is selected if RSS_3 is the smallest one among RSS_1, RSS_2, and RSS_3,

and step (ii) is not changed.

3-D Image Denoising

This part is, in many ways, similar to the method discussed in Section 3.2.3. Still, for readers' convenience, a brief description is provided below.

After the edge surfaces in the neighborhood $O(x, y, z)$ are approximated by a surface template, the neighborhood is divided by the surface template into sub-neighborhoods. The one containing the voxel (x, y, z) is denoted as $U(x, y, z)$. Then, $f(x, y, z)$ can be estimated by the solution to a of the minimization problem (3.2), after $K_{h_n^*}(x_i, y_j, z_k)$ is replaced by $I((x_i, y_j, z_k) \in U(x, y, z))K_{h_n}(x_i, y_j, z_k)$, where $I(\cdot)$ is the indicator function and it equals 1 if its argument is "true" and 0 otherwise. From (3.2), and the expressions given in (3.3) for its solutions, we can see that $\widehat{f}(x, y, z)$ is a weighted average of the observed image intensities whose voxels are located on the same side of the estimated surface template in $O(x, y, z)$ as the given voxel (x, y, z). Intuitively, as long as the estimated surface template approximates the underlying edge surfaces well, $\widehat{f}(x, y, z)$ should preserve edges and major edge features well.

In the proposed denoising procedure, there are two parameters h_n^* and h_n involved. We suggest choosing them by the following cross-validation (CV) procedure (3.9). Because each $\widehat{f}_{-i,-j,-k}(x_i, y_j, z_k)$ is computed in a small neighborhood $O(x, y, z)$, computation involved in the CV procedure (3.9) is manageable.

It should be pointed out that the denoising procedure discussed above is for cases when the pointwise noise contained in the observed image has a mean of 0 (3.1). In some applications, such as MRI imaging, the noise level at a given voxel may depend on the true image intensity level, and the noise distribution can often be described by a Rician, Poisson, noncentral Chi-square, or other models Macovski (1996). In such cases, denoised images by procedures designed for removing additive zero-mean noise would be biased for estimating the true images, and many bias-correction methods have been proposed in the literature (e.g., Section 1.3 and related references). In the next section, we will present a numerical example with Rician noise involved.

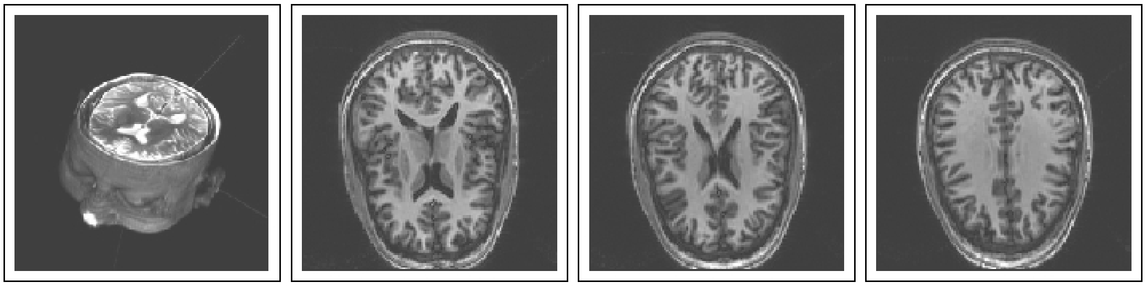

Figure 3.12: A demonstration of a 3-D image and its three 2-D slices.

For a real image, there are regions where f is smooth. In these regions, the number of detected edge voxels should be small. So, before estimating f, we suggest counting the number of detected edge voxels in $O(x, y, z)$. If the number is so small (e.g., smaller than $(nh_n)^2$) that a potential edge surface in $O(x, y, z)$ is unlikely, then $f(x, y, z)$ can be estimated simply by the conventional LLK estimator constructed from all observations in $O(x, y, z)$. The denoising procedure described in the first paragraph of this subsection is used only when the number of detected edge voxels in $O(x, y, z)$ is relatively large. To do so, there are at least two benefits. One is that much computation is saved, because the conventional LLK estimator is much easier to compute, compared to the proposed denoising procedure based on local edge surface approximation. The other benefit is that the estimated f would be more reliable in cases when the number of detected edge voxels is small, because it is constructed from all observations in $O(x, y, z)$, instead of from part of the observations. Considering the fact that most voxels in an image are not edge voxels, these benefits should be substantial.

3.5.2 Numerical studies

In this section, we present some numerical examples to evaluate the performance of the proposed 3-D image denoising procedure, in comparison with three existing procedures, including the ones based on total variation Getreuer (2007) (denoted as TV, a different version than the one used in Section 3.4), anisotropic diffusion Lopes (2007) (denoted as AD), and optimized non-local means Coupe et al. (2008b) (denoted as ONLM). In Section 3.5.1, it has been pointed out that the first three surface templates shown in Figure 3.11 could describe major edge features in 3-D images well. In order to investigate the impact of the inclusion of the fourth surface template (i.e., the triangular pyramid shown in the lower-right panel of Figure 3.11) on the performance of our proposed method, we consider its two versions. One uses the first three surface templates only (denoted as NEW), and the other uses all four surface templates (denoted as NEW-P). In both versions, the kernel function K is chosen to be the Epanechnikov kernel function $K(x, y, z) = C(1-x^2)(1-y^2)(1-z^2)$ defined in the unit ball $\{(x, y, z) : x^2+y^2+z^2 \leq 1\}$, where $C > 0$ is a normalization constant. It has been shown in the statistical literature Epanechnikov (1969) that this kernel function has certain optimality properties. The procedure TV has a regularization parameter involved, the procedure AD is an iterative algorithm and contains two parameters, i.e., the diffusion parameter and the number of iterations, and the procedure ONLM has two bandwidth parameters to choose. To evaluate the performance of a denoised image $\widehat{f}$, a standard statistical criterion is the mean integrated squared error (MISE), defined as $\text{MISE}(\widehat{f}, f) = \int_0^1 \int_0^1 \int_0^1 [\widehat{f}(x, y, z) - f(x, y, z)]^2 \, dxdydz$, which is estimated by the sample mean of

$$\text{ISE}(\widehat{f}, f) = \frac{1}{n^3} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \left[\widehat{f}(x_i, y_j, z_k) - f(x_i, y_j, z_k)\right]^2$$

based on 100 replications.

We first consider a 3-D MRI image of a human brain which has 128 × 128 × 52 voxels. Its image intensity levels range from 0 to 809. A demonstration of the 3-D image and its three slices are shown in Figure 12. This is the same image used in Section 3.4. I.i.d. random noise from the distribution $N(0, \sigma^2)$ is added to the image, and σ is chosen to be 80, 100, or 120, representing low, medium, and high noise levels. With each σ value, the parameters of all denoising procedures are chosen such that their estimated MISE values based on 100 replications reach the minimum. For the proposed denoising procedure NEW, we also consider choosing its parameters by the CV procedure (3.9), to investigate the effectiveness of the CV procedure. The corresponding method is denoted as NEW-CV.

The estimated MISE values, their standard errors, and the procedure parameter values of the related methods are presented in the first three columns of Table 3.5. From the table, we can see that the proposed procedure NEW outperforms all three competing methods TV, AD and ONLM in all cases, and the outperformance is statistically significant because the difference of the MISE values of the procedure NEW and any one of these methods is larger than 2 times of the sum of the two corresponding standard errors. Also, when its parameters are chosen by the CV procedure (3.9), its performance gets slightly worse, compared to the procedure NEW with the parameters chosen by minimizing MISE. But, NEW-CV still outperforms all three competing methods in all cases by a reasonably large margin. By comparing NEW with NEW-P, it can be seen that the performance of NEW-P is indeed only marginally better than that of NEW. The first three columns in Figure 3.13 demonstrate the three slices of the denoised images by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, when $\sigma = 100$. The corresponding slices of the original image are those shown in Figure 3.12 . From the images, it can be seen that edge structures are better preserved by procedures NEW, NEW-CV, and NEW-P (cf., several central dark regions in images shown in the first column). The residual plots of the images in the first column are shown in the fourth column, from which we can see that the residual plots of procedures TV, AD and ONLM show slight patterns around the brain boundary and such patterns are almost invisible in the residual plots of procedures NEW, NEW-CV and NEW-P, which can be confirmed by the close-up views of an upper-right portion of the residual images that are shown in column 5 of Figure 3.13. To further investigate the denoised images, the density curves of the image intensities of the denoised images when $\sigma = 100$ are shown in Figure 3.14. Because the density curves of the procedures NEW, NEW-CV, and NEW-P are almost identical, only the one of NEW-CV is visible in the plot. From the plot, it can be seen that, when image intensities are in the range from 0 to about 70, the density curves of NEW, NEW-CV, and NEW-P are closer to the density curve of the true image, shown by the thick solid curve in the plot, than the density curves of the three competing procedures. In cases when image intensities are larger than 70, the density curves of all procedures are close to each other. From the plot, it can also be seen that more than 80% image intensities of the true image or each of the denoised images are below 70. So, we can conclude that overall the image intensities of the denoised images by procedures NEW, NEW-CV, and NEW-P are closer to the image intensities of the true image, compared to the image intensities of the denoised images by the three competing procedures.

Table 3.5: In each entry, the first line presents the estimated MISE value and their standard errors (in parenthesis), the second line presents the searched procedure parameter values.

Method	$\sigma = 80$	$\sigma = 100$	$\sigma = 120$	variable σ
TV	1119.3 (2.9) .0030	1401.1 (3.9) .0025	1649.1 (4.9) .0019	1073.1 (2.8) .0032
AD	1157.8 (3.5) 475,1	1525.0 (4.8) 700,1	1916.9 (6.3) 900,1	1088.9 (3.3) 225,2
ONLM	884.8 (3.4) 12,1	1127.5 (4.6) 16,1	1364.0 (6.2) 25,1	845.2 (3.2) 11,1
NEW	837.3 (2.4) .0172,.0156	1022.7 (3.1) .0188,.0188	1189.2 (4.1) .0188,.0188	811.5 (2.2) .0172,.0156
NEW-CV	850.4 (2.4) .0117,.0156	1029.0 (2.9) .0172,.0188	1189.2 (4.1) .0188,.0188	823.8 (2.4) .0117,.0156
NEW-P	835.3 (2.5) .0172,.0156	1011.8 (3.5) .0180,.0180	1184.1 (4.7) .0180,.0180	809.7 (2.2) .0172,.0156

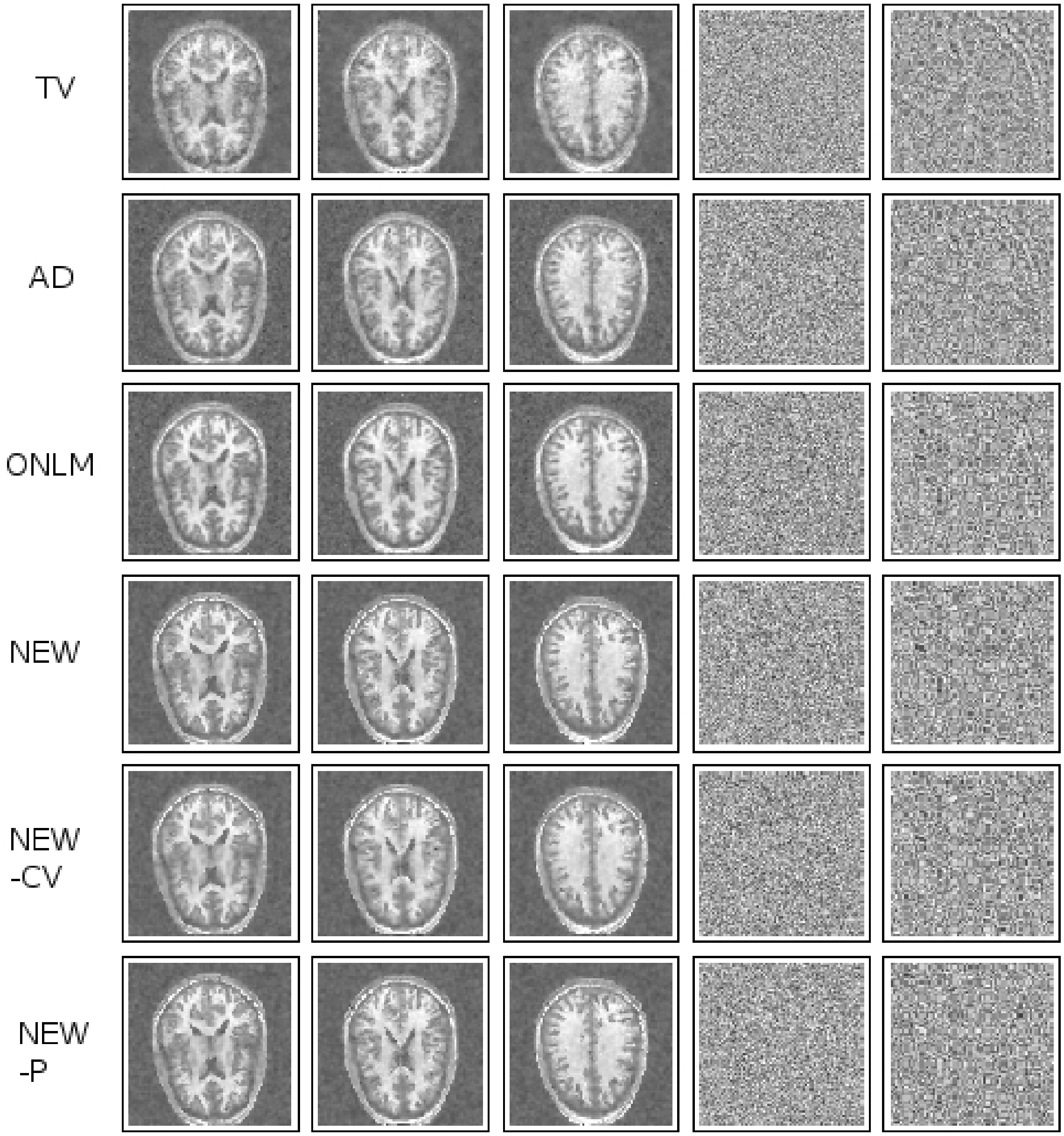

Figure 3.13: The first three columns show the three slices of the denoised images by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, in six rows when $\sigma = 100$. Images in column 4 are residual images of those shown in column 1. Images in column 5 show the close-up views of an upper-right portion of the images in column 4.

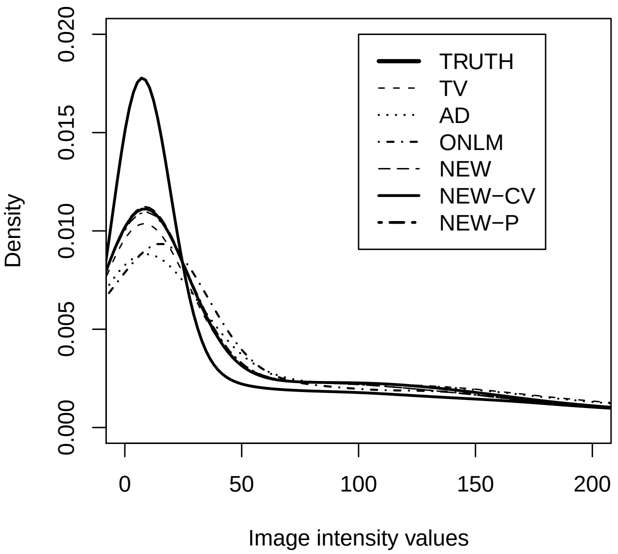

Figure 3.14: Density curves of the image intensities of the true 3-D image and the six denoised 3-D images when $\sigma = 100$.

In practice, noise level may not be homogeneous in the entire image. To simulate this scenario, we consider the following variable σ:

$$\sigma(x, y, z) = 30 + 50 \exp\left[-\frac{(x - .5)^2 + (y - .5)^2 + (z - .5)^2}{4}\right]. \tag{3.12}$$

The MISE values, their standard errors, and the searched procedure parameter values of the six methods in this case are shown in the last column of Table 3.5. Denoised images are shown in Figure 3.15 for the first slice presented in Figure 3.12. From Table 3.5 and Figure 3.15, it can be seen that procedures NEW, NEW-CV, and NEW-P also perform better than the three competing procedures in this case.

Next, we consider adding the Rician noise to the 3-D test image shown in Figure 3.12. By adding the Rician noise, the observed image intensity at the voxel (x, y, z)

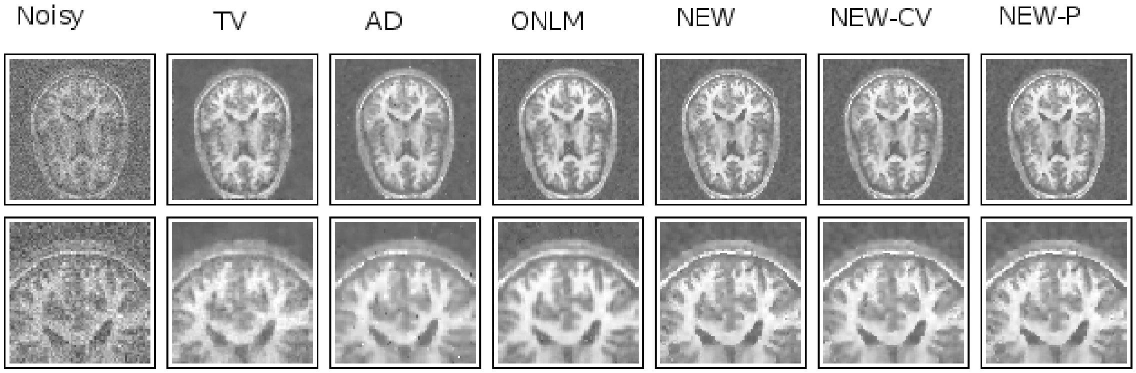

Figure 3.15: Images in the first row show the first slice (shown in Figure 3.12) of the noisy image and the denoised images by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, when the noise level is variable, as defined in refeq:03:006. Images in the second row show the close-up views of a top portion of the corresponding images in the first row.

is generated by $\sqrt{[f(x,y,z)+\epsilon_1]^2+\epsilon_2^2}$, where ϵ_1 and ϵ_2 are i.i.d. noise from the distribution $N(0,\tilde{\sigma}^2)$. As in the previous examples, we consider cases when $\tilde{\sigma} = 80, 100$, and 120. For each denoising procedure, we use the bias correction method proposed in Wiest-Daessle et al. (2008) to remove estimation bias. Let $\tilde{f}(x,y,z)$ be the intensity of the denoised image by a given denoising procedure. Then, its bias-corrected version is defined to be $\sqrt{\tilde{f}^2(x,y,z)-2\tilde{\sigma}^2}$. In practice, $\tilde{\sigma}$ is often unknown and should be estimated from the observed image. In this example, it is estimated by $\sqrt{s/2}$, where s is the sample standard deviation of the squared observed intensities at the first $16 \times 16 \times 13$ voxels. The calculated MISE values based on 100 replications of all six procedures are presented in Figure 3.16. Again, the MISE curve of the procedure NEW is not visible in the plot because it is overlapped with the one of NEW-CV. From the plot, it can be seen that procedures NEW, NEW-CV, and NEW-P outperform all three competing procedures in all cases except the case when $\tilde{\sigma} = 80$. In that case, the procedure ONLM is the best. In the case when $\tilde{\sigma} = 100$, the procedure ONLM performs similarly to the procedures NEW, NEW-CV, and NEW-P, which

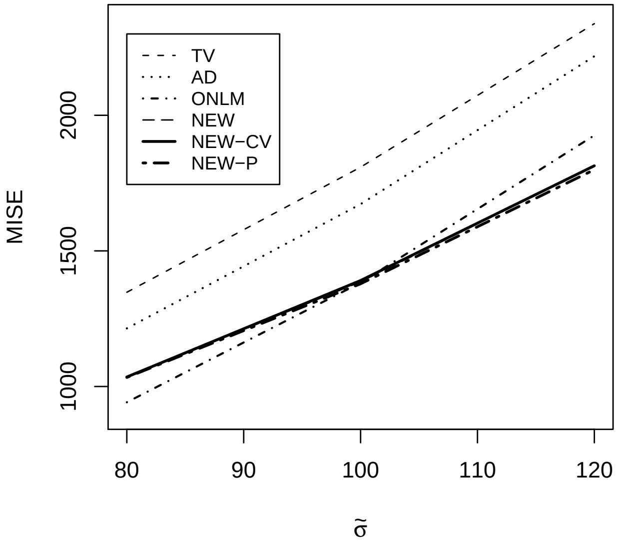

Figure 3.16: MISE values of various methods based on 100 replications when $\tilde{\sigma} = 80, 100$, and 120.

can also be seen from Figure 3.17 where the denoised images of the six procedures are shown for the second slice presented in Figure 3.12.

We next consider another 3-D MRI test image of a person's head which has $128 \times 128 \times 126$ voxels with intensity levels ranging from 0 to 255. Three slices of the image are shown in Figure 3.18. Then, i.i.d. noise from the distribution $N(0, \sigma^2)$ is added to the image. We consider three noise levels: $\sigma = 50, 75$, and 100, representing low, medium, and high noise levels. With each noise level, parameters of all the denoising procedures considered are chosen in exactly the same way as that in the example of Table 3.5. The estimated MISE values based on 100 replications, their standard errors, and the procedure parameter values are shown in the first three columns of Table 3.6. From the table, we can see that (i) procedures NEW, NEW-CV, and NEW-P are uniformly better than the three competing procedures TV, AD, and ONLM, (ii) procedure NEW-CV is slightly worse than procedure NEW, and (iii) procedure NEW-P only marginally improves procedure NEW. All these results are consistent

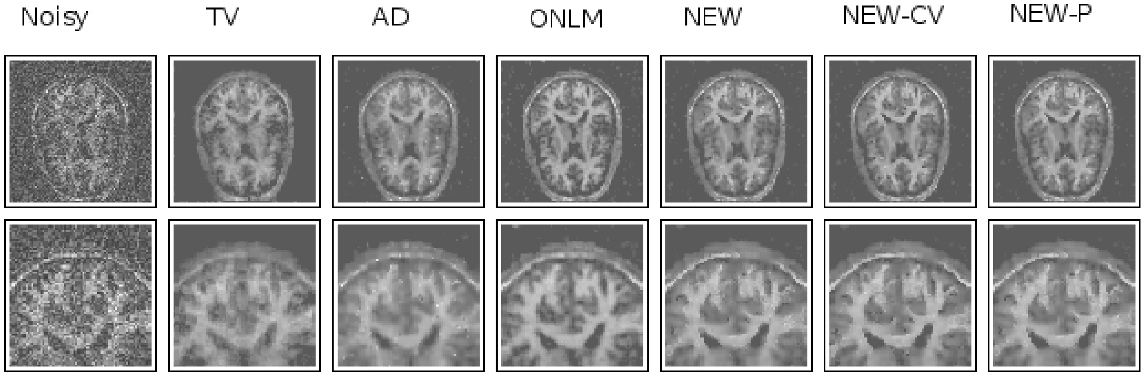

Figure 3.17: Images in the first row show a slice of the noisy image and its denoised versions by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, when Rician noise with $\tilde{\sigma} = 100$ is considered. The slice presented here corresponds to the second slice shown in Figure 3.12. Images in the second row show the close-up views of a top portion of the images in the first row.

with the ones found in the example of Table 3.5. The first three columns of Figure 3.19 demonstrate three slices of the denoised images by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, when $\sigma = 75$. The three slices correspond to the ones shown in Figure 3.18. From the images, it seems that the edge structures are better preserved by procedures NEW, NEW-CV, and NEW-P (cf., the boundary of the neck part in images shown in the first column). This can be confirmed by the deviation plots (i.e., plots of true images minus denoised images) of the images in the first column, shown in the fourth column of Figure 3.19, and their close-up views (shown in the fifth column of Figure 3.19) of the neck portion.

In MRI imaging, salt-and-pepper noise is quite common. Next, we simulate a scenario with salt-and-pepper noise using the test image of head, by changing the image intensities of 5% randomly selected voxels to 0 and changing the image intensities of another 5% randomly selected voxels to 255. The noisy image is shown in the (1,1)th plot of Figure 3.20 for the first slice presented in Figure 3.18. Then, all six denoising procedures are applied to the noisy image. Their procedure parameters are chosen in

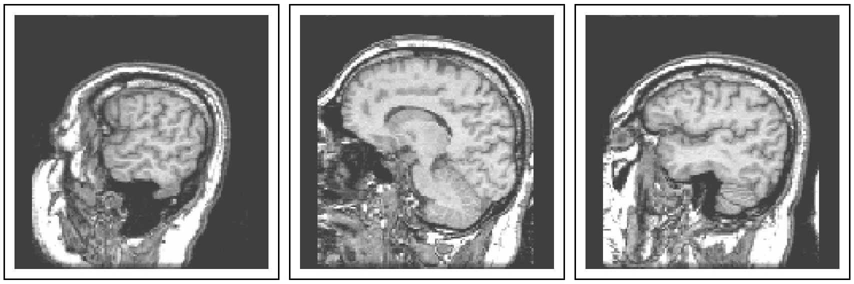

Figure 3.18: Three slices of a 3-D test image of a person's head.

Table 3.6: In each entry, the first line presents the estimated MISE value and their standard errors (in parenthesis), the second line presents the searched procedure parameter values.

Method	$\sigma = 50$	$\sigma = 75$	$\sigma = 100$	salt-and-pepper
TV	557.1 (0.7) .0030	803.5 (1.6) .0019	1020.2 (2.0) .0013	752.1 (2.1) .0033
AD	488.9 (1.0) 135,2	749.6 (1.3) 230,2	964.4 (1.7) 350,2	836.0 (1.7) 800,1
ONLM	504.5 (0.9) 10,1	707.5 (1.7) 14,1	934.2 (2.8) 16,1	629.9 (2.0) 8,1
NEW	462.1 (0.7) .0172,.0141	610.9 (1.0) .0188,.0203	731.3 (1.4) .0219,.0234	594.5 (1.7) .0219,.0156
NEW-CV	482.8 (0.7) .0172,.0188	610.9 (1.0) .0188,.0188	731.7 (1.5) .0219,.0219	617.8 (1.4) .0188,.0188
NEW-P	461.6 (0.7) .0180,.0156	608.9 (1.1) .0188,.0188	729.7 (1.7) .0219,.0203	592.9 (1.5) .0250,.0156

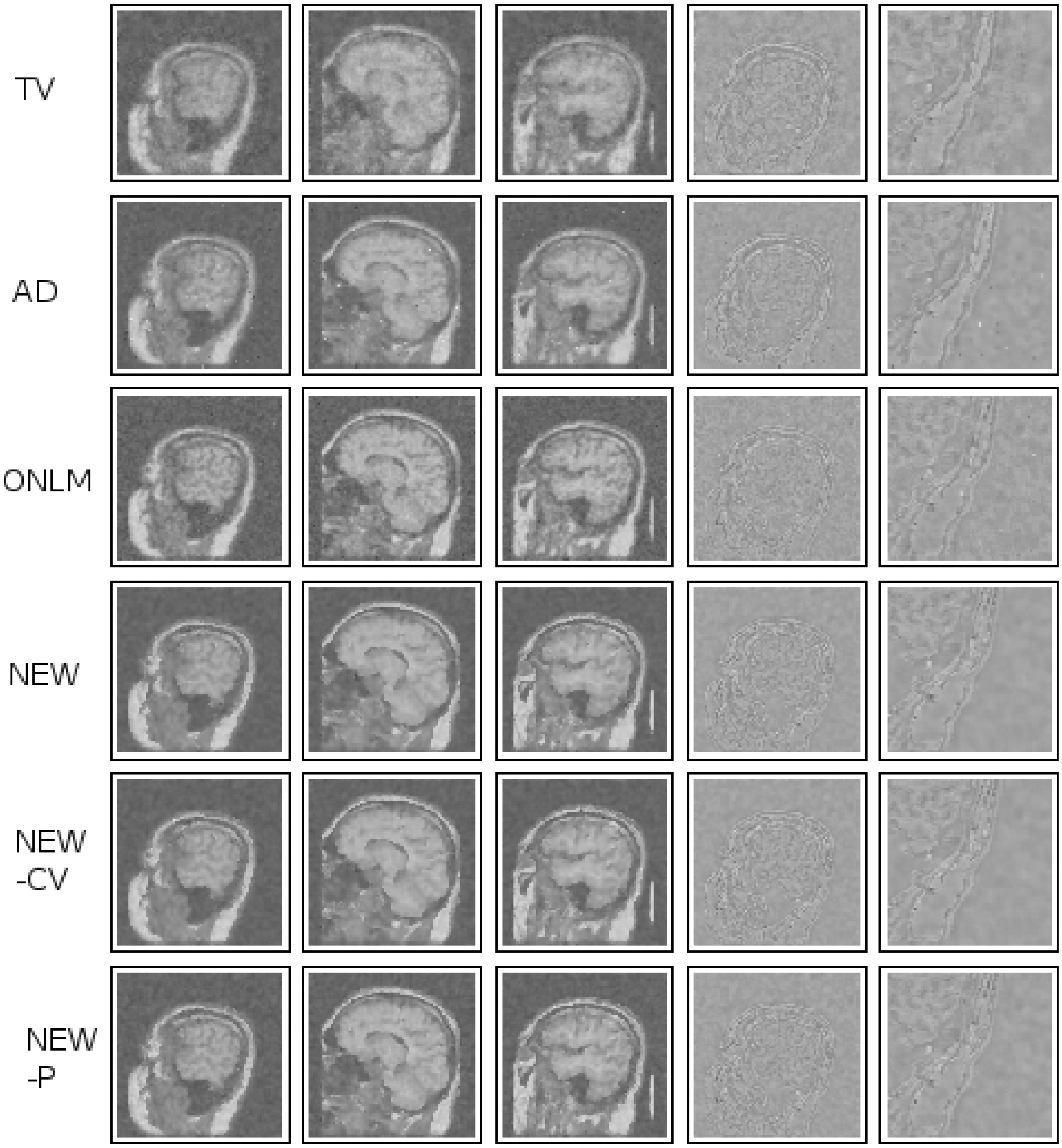

Figure 3.19: The first three columns show the three slices of the denoised images of the test image of head by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, in six rows when $\sigma = 75$. Images in column 4 are residual images of those shown in column 1. Images in column 5 show the close-up views of the neck portion of the images in column 4.

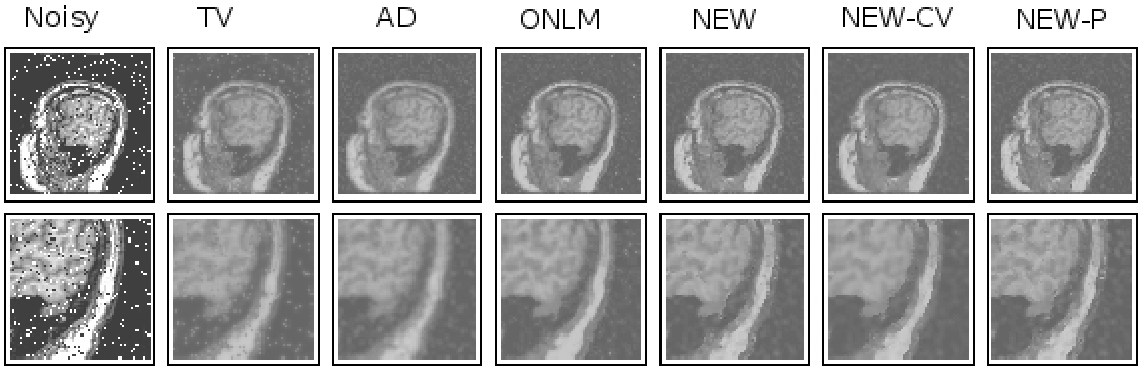

Figure 3.20: Images in the first row show the first slice (shown in Figure 3.18) of the noisy image and the denoised images by procedures TV, AD, ONLM, NEW, NEW-CV, and NEW-P, respectively, when 10% salt-and-pepper noise is added to the test image of head. Images in the second row show the close-up views of the neck portion of the corresponding images in the first row.

the same way as before. The MISE values, their standard errors, and the searched procedure parameter values of the six procedures in this case are shown in the last column of Table 3.6. Denoised images are shown in the first row of Figure 3.20 for the first slice. Images in the second row of Figure 3.20 show the close-up views of the neck portion of the images in the first row. From Table 3.6 and Figure 3.20, it can be seen that procedures NEW, NEW-CV, and NEW-P also perform better than the three competing procedures in this case.

Chapter 4

Efficient Bias Correction for MRI Image Denoising

Magnetic resonance imaging (MRI) is a popular radiology technique that is used for visualizing detailed internal structure of the body. Observed MRI images are generated by the inverse Fourier transformation from received frequency signals of a MR scanner system. Previous research has demonstrated that random noise involved in the observed MRI images can be adequately described by the so-called rician noise model. Under that model, the observed image intensity at a given pixel is a non-linear function of the true image intensity and two independent zero-mean random variables with a same normal distribution. Because of such a complicated noise structure in the observed MRI images, denoised images by conventional denoising methods are usually biased, and the bias would reduce image contrast and affect negatively on subsequent image analysis. Therefore, it is important to address the bias issue properly. To this end, several bias correction procedures have been proposed in the literature (Chapter 1.3). Most of them are proposed based on intuition. In Section 4.2, we study the rician noise and the corresponding bias-correction problem systematically, and propose a new and more accurate bias-correction formula. Numerical studies in Section 4.3 show that it works well in various applications.

4.1 Introduction

Magnetic resonance imaging (MRI) is a technique used mainly for assessing pathological or other physiological conditions in living tissues, by visualizing the inside of living organisms Vlaardingerbroek and den Boer (1999). In simple terms, its methodological basis lies in: (i) different tissues have different compositions and physical properties, such as water molecule densities, from which the tissue type at a given position can be determined, and (ii) these differences, in water molecule density say, can be depicted as various image contrasts using the MRI technique.

When a part of a body (e.g., a patient's head) is placed in a uniform magnetic field of a given direction, say, the z direction, the hydrogen nuclei of water in that part would align themselves in parallel or anti-parallel with the field, creating a net magnetization, and rotate with the Larmor frequency. The basis of MRI lies in manipulating the local magnetic field such that the local resonant frequency would differ at different locations, which is achieved by applying additional, small, linear magnetic field gradients. In a MR scanner system, three orthogonally positioned gradient coils would produce such magnetic fields that vary linearly along their respective axes (e.g., x, y, and z axes), and these small fields are added to the main magnetic field. Turning on the coils in any particular combination would produce a field gradient along any desired direction. After applying radio frequency (RF) pulses transmitted by a separate RF coil, emitted radiation is absorbed by nuclei. Consequently, the net magnetization is tipped away from the main z axis; the nuclei continue their rotation, and as the excited nuclei relax back to the initial lower-energy alignment along the main field, RF signals are re-emitted and received by a RF receiver coil. Along the z direction, suppose a particular perpendicular slice of the body part at $z = z_0$ is to be imaged. Then, a RF pulse with frequency corresponding to that slice position would excite the nuclei in that plane. Considering only the proton density and spin

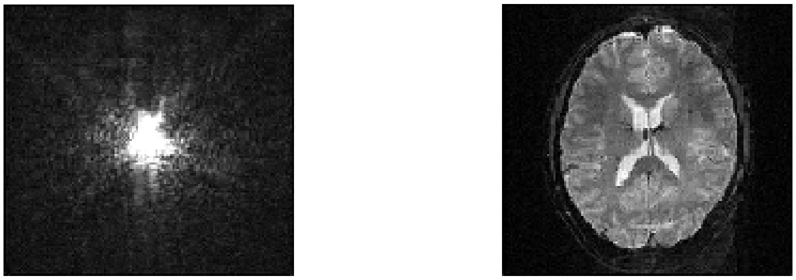

Figure 4.1: (a) Signals in frequency domain. (b) Corresponding spatial image obtained by the discrete inverse Fourier transformation of the signals shown in (a).

relaxation, the received signal can be expressed by

$$S(k_x, k_y) = \int\int_{\Omega_{z_0}} m(x, y) \exp[i2\pi(k_x x + k_y y)]dxdy, \tag{4.1}$$

where Ω_{z_0} denotes the 2-dimensional (2-D) region of the slice, $m(x, y)$ is the density of hydrogen protons at (x, y), and k_x and k_y are the frequency change rates along the x and y directions of the local magnetic fields. Note that some constant multipliers have been ignored on the right-hand-side of (1) for simplicity. It can be seen that $S(k_x, k_y)$ is a Fourier transformation of $m(x, y)$. Therefore, if we have signals $S(k_x, k_y)$ in frequency domain, for all $k_x, k_y = 1, 2, \ldots, n$, then $m(x, y)$ can be determined in spatial domain at $n \times n$ regularly spaced pixels by the discrete inverse Fourier transformation (Chapter 7 in Qiu (2005)), as demonstrated in Figure 4.1.

Equation (4.1) is only a theoretical model for describing MRI image acquisition. In practice, there will be random noise involved in the received signal $S(k_x, k_y)$ in the frequency domain, due to various reasons, including hardware imperfections, signal dropouts caused by field inhomogeneity, and so forth. There has been a considerable amount of existing research for describing the noise pattern in observed MRI images in the spatial domain e.g., Gudbjartsson and Patz (1995), Macovski (1996). Most existing research shows that the noise pattern in observed MRI images can be adequately described by the following rician noise model, e.g., Bernstein et al. (1989), Sijbers et al. (1998):

$$Z(x,y) = \sqrt{[m(x,y) + \mu_1(x,y)]^2 + \mu_2^2(x,y)}, \tag{4.2}$$

where $Z(x,y)$ denotes the observed image intensity at the pixel (x,y), $m(x,y)$ is the true image intensity, and $\mu_1(x,y)$ and $\mu_2(x,y)$ are two independent random variables with the normal distribution $N(0,\sigma^2)$ where σ is often unknown. One explanation of the rician noise model (4.1) is that, when $Z(x,y)$ is reconstructed from the frequency signal $S(k_x,k_y)$ using the inverse Fourier transformation, the imaginary part of the complex output of the transformation does not contain any information about $m(x,y)$ (Nowak, 1999).

From (4.2), it can be seen that the rician noise involved in the observed image $Z(x,y)$ does not have the traditional zero-mean and additive structure. Instead, its noise level depends on the true image intensity $m(x,y)$, and it contaminates $m(x,y)$ nonlinearly. Kristoffersen (2007) and Zhu et al. (2009). systematically studied the statistical properties of rician noise in MRI images. Because of non-additivity and nonlinearity of rician noise, many conventional image denoising techniques, including the total variance minimization (Rudin et al., 1992), anisotropic diffusion (Perona and Malik, 1990), non-local means (Buades et al., 2005), jump surface estimation (Qiu, 1998), and so forth, would result in biased estimates of the true image $m(x,y)$. The contrast of these biased estimates is usually lower than that of the true image (Nowak, 1999), because the magnitude of bias depends on observed image intensities and the bias is usually larger at places where the observed image intensities are smaller. Therefore, it is important to reduce the bias and obtain a better estimate of the true image, so that the true image structures can be better revealed.

In the literature, a number of methods have been proposed to estimate noise level in MRI images and to reduce the bias caused by the rician noise. Aja-Fernández et al. (2008), Coupe et al. (2010), Rajan et al. (2010) etc. recently proposed different methods to estimate the noise level in MRI images. Sijbers and den Dekker (2004) suggested a denoising method based on the maximum likelihood estimation (MLE) of the true image $m(x,y)$ in model (4.2). However, the MLE is not easy to obtain due to the nonlinearity of the model and to the fact that the true image usually has many edge curves hidden in the observed image intensities. Gudbjartsson and Patz (1995) suggested the bias-correction formula

$$\widehat{m}_1(x,y) = \sqrt{|\widetilde{m}^2(x,y) - \sigma^2|}, \tag{4.3}$$

where $\widetilde{m}(x,y)$ denotes the denoised image by a conventional image denoising procedure, e.g., the non-local means estimate (Coupe et al., 2008b), and $\widehat{m}_1(x,y)$ is the bias-corrected estimate of $m(x,y)$. The formula (4.3) was derived using the normal distribution approximation to the rician noise distribution, which may not provide an accurate approximation in practice. Wiest-Daessle et al. (2008) noticed that $E(Z^2(x,y)) = m^2(x,y) + 2\sigma^2$, based on which they proposed the following bias-correction procedure:

$$\widehat{m}_2(x,y) = \sqrt{\widetilde{m}^2(x,y) - 2\sigma^2}\, I\left(\widetilde{m}(x,y) \geq \sqrt{2}\sigma\right), \tag{4.4}$$

where $I(\cdot)$ is an indicator function taking the value of 1 if "$\cdot$=True" and 0 otherwise. Note that the case $\widetilde{m}(x,y) < \sqrt{2}\sigma$ can happen in reality, especially in the background regions of the MRI images. It has been shown that formula (4.4) can reduce bias. However, it is clear that we do not have the equality $m(x,y) = \sqrt{E(Z(x,y))^2 - 2\sigma^2}$ in cases when $m^2(x,y) = E(Z^2(x,y)) - 2\sigma^2$. Therefore, this formula should have room for improvement.

In this Chapter, we propose a novel bias-correction method using regression analysis. From (4.2), it can be noticed that conventional image denoising methods actually estimate $E(Z(x, y))$, instead of $m(x, y)$. The resulting bias is mainly contributed by $E(Z(x, y)) - m(x, y)$ which is a function of $m(x, y)$. This function can be estimated accurately using regression and Monte Carlo simulation. Numerical results show that this method can provide a better bias correction, compared to the existing methods mentioned above. The details of this method are described in Section 4.2. Some numerical results are presented in Section 4.3.

4.2 Proposed bias correction method

In this section, we describe our proposed bias correction method in details. First, we re-write model (4.2) as

$$Z(x, y) = f(m(x, y)) + \varepsilon_m(x, y), \tag{4.5}$$

where $f(m(x, y)) = E(Z(x, y))$ is the mean of the observed image $Z(x, y)$, and $\varepsilon_m(x, y)$ is the zero-mean random part of $Z(x, y)$. Obviously, $\varepsilon_m(x, y)$ depends on the true image intensity $m(x, y)$, which is indicated by its subscript. As discussed in Section 4.1, the denoised image $\widetilde{m}(x, y)$ by a conventional image denoising procedure is usually an asymptotically unbiased estimator of $f(m(x, y))$, which would have an asymptotic bias of $f(m(x, y)) - m(x, y)$. If the function f has an inverse $g = f^{-1}$ and g can be properly specified, then a reasonable, bias-corrected, estimate of $m(x, y)$ would be

$$\widehat{m}(x, y) = g(\widetilde{m}(x, y). \tag{4.6}$$

From models (4.2) and (4.5), we have

$$Z(x,y) = \sigma f_1\left(\frac{m(x,y)}{\sigma}\right) + \sigma\varepsilon_{m,1}(x,y),$$

where f_1 and $\varepsilon_{m,1}$ denote the values of f and ε_m when $\sigma = 1$. Since $t = m(x,y)/\sigma$ can be interpreted as the signal-to-noise ratio (SNR), f_1 can also be regarded as a function of SNR. Further, the above expression implies that f and g have the following properties

$$f(t\sigma) = \sigma f_1(t), \quad g(s\sigma) = \sigma g_1(s), \tag{4.7}$$

where g_1 denotes the value of g when $\sigma = 1$ (i.e., $g_1 = f_1^{-1}$), and $s = f_1(t)$. By (4.6) and (4.7), if f_1 or g_1 can be properly specified, then the bias-corrected estimate of $m(x,y)$ can be defined by

$$\widehat{m}(x,y) = \widehat{\sigma} g_1\left(\frac{\widetilde{m}(x,y)}{\widehat{\sigma}}\right), \tag{4.8}$$

where $\widehat{\sigma}$ is a reasonable estimate of σ.

The closed-form formula of the function f_1 or g_1 is difficult to derive. However, their expressions can be approximated accurately by a numerical approach described as follows. Let us first discuss the expression of $f_1(t)$. Because the image intensity of a real image is always non-negative, let us constrain ourselves to $t \geq 0$. For a given t, we can generate $\widetilde{n}$ observed image intensity values by (4.2), in which $m(x,y)$ is set to be t and $N_1(x,y)$ and $N_2(x,y)$ are replaced by two independent random numbers from the standard normal distribution. Then, by (4.5), $f_1(t)$ can be estimated by the sample mean of the $\widetilde{n}$ observed image intensity values. When $\widetilde{n}$ is chosen large, such an estimate should be accurate by the strong law of large numbers (Chapter 5 of Chung (1974)). For instance, when $\widetilde{n} = 10,000$ and t changes its values in $[0, 10]$ with a step 0.01, the estimated $f_1(t)$ is shown in Figure 4.2(a) by the dark solid line. From the plot, it can be seen that it is an increasing function when $t \geq 0$, and its value when $t = 0$ can be easily computed to be 1.253. As a reference, the light solid line in the plot denotes the identity transformation. As described above, a conventional image denoising procedure would have an asymptotic bias of $f(m(x, y)) - m(x, y) = \sigma[f_1(t) - t]$. Therefore, from Figure 4.2(a), it can be seen that (i) the conventional image denoising procedure would have positive biases across the image, and (ii) the biases would be larger at places with smaller SNR values.

The estimated $g_1(s)$, which is obtained from the estimated $f_1(t)$ shown in Figure 4.2(a) by switching the x and y axes, is shown in Figure 4.2(b) by the dark solid line. As a comparison, in the case when $\sigma = 1$, the corresponding transformations of the bias correction formulas (4.3) and (4.4) described in Section 4.1, suggested by Gudbjartsson and Patz (1995) and Wiest-Daessle et al. (2008), respectively, are $\sqrt{|s^2 - 1|}$ and $\sqrt{s^2 - 2}I(s \geq \sqrt{2})$. They are also displayed in the same plot by long-dashed and short-dashed lines, respectively. From the plot, we can see that all three bias correction methods try to pull down the estimated image intensities of a conventional image denoising procedure. However, compared to the proposed formula (4.8), the formula (4.3) seems not to correct the bias big enough especially when the SNR $s \leq 1$, while the formula (4.4) seems to over-correct the bias.

In practice, we need to obtain a good estimator $\widehat{\sigma}$ of σ before we can use formula (4.8) for bias correction. To this end, one simple method is to use the sample variance of $Z^2(x, y)$ in the background region of an MRI image, because $m(x, y)$ is close to zero in the background and thus the sample variance of $Z^2(x, y)$ in that region would be approximately $4\sigma^4$. In all the numerical examples presented in this paper, $\widehat{\sigma}$ is computed using this method, based on the first $[1, 32] \times [1, 32]$ pixels of an MRI image. In the literature, there are a few alternative methods for estimating σ (Aja-Fernández et al., 2008; Coupe et al., 2010; Rajan et al., 2010) but these methods are more computationally extensive. We checked the numerical performance of our

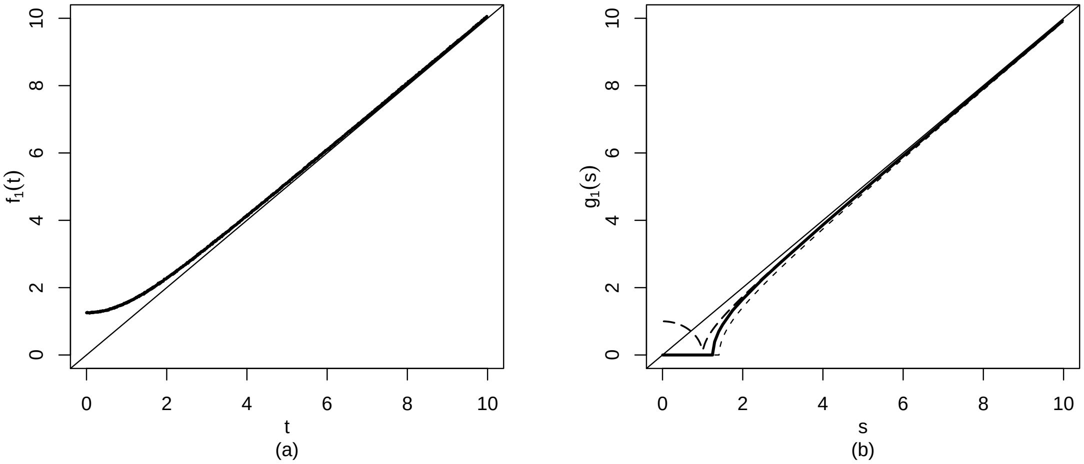

Figure 4.2: (a) Function $f_1(t)$ (dark solid line) estimated by simulation, and the identity transformation (light solid line). (b) Transformation function $g_1(s)$ in the proposed bias correction formula (4.8) (dark solid line), the corresponding transformation functions of the formulas (4.3) and (4.4) (long-dashed and short-dashed lines, respectively), and the identity transformation (light solid line).

proposed bias-correction method, and found that its performance using the simple estimator $\widehat{\sigma}$ described above is similar to its performance when we pretend that σ is known.

After $\widehat{\sigma}$ is obtained, we can compute the value of $g_1(s)$ with $s = \widetilde{m}(x,y)/\widehat{\sigma}$ by an algorithm similar to the one described above for creating Figure 4.2(b), after $\widetilde{m}(x,y)$ is computed by a conventional image denoising procedure. However, if we can derive an explicit formula that provides a good approximation to the transformation function $g_1(s)$, then the use of our proposed bias correction formula (4.8) will be greatly simplified. To this end, we suggest approximating $g_1(s)$ by the first-order regression spline function

$$\beta_{-1} + \beta_0 s + \sum_{j=1}^{\ell} \beta_j (s - v_j)_+,$$

where $\boldsymbol{\beta} = (\beta_{-1}, \beta_0, \beta_1, \ldots, \beta_\ell)$ are coefficients, $(1, s, (s-v_1)_+, \ldots, (s-v_\ell)_+)$ are the basis functions, $(s-v)_+ = (s-v)$ if $s \geq v$ and 0 otherwise, and $(v_1, v_2, \ldots, v_\ell)$ are the set of knots chosen to be $(1.253, 1.3, 1.4, 1.5, 1.75, 2.0, 2.5, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0)$. From Figure 4.2(b), we can see that $g_1(s)$ is almost straight when $s > 2$, and it is curved mainly when $s \in [1.253, 2]$. That is the reason why we use denser knots in $[1.253, 2]$, and relatively less knots when $s > 2$. Also, when $s > 10$, $g_1(s)$ can be simply approximated by s. So, we constrain ourselves to cases when $s \in [0, 10]$. To estimate the above regression spline function in $[0, 10]$, we generate the data in the same way as we compute the estimated values of $g_1(s)$ shown in Figure 4.2(b), and the estimated values of $g_1(s)$ are used as the data here. Then, the regression spline approximation, denoted as $\widehat{g}_{S,1}(s)$, has the expression

$$\widehat{g}_{S,1}(s) = \widehat{\beta}_{-1} + \widehat{\beta}_0 s + \sum_{j=1}^{\ell} \widehat{\beta}_j (s - v_j)_+$$

where $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_{-1}, \widehat{\beta}_0, \widehat{\beta}_1, \ldots, \widehat{\beta}_\ell)$ is the least squares estimate of $\boldsymbol{\beta}$.

In formula (4.4) suggested by Wiest-Daessle et al. (2008), there is a threshold value $\sqrt{2}$ involved. When the ratio $\widetilde{m}(x,y)/\sigma < \sqrt{2}$, (4.4) simply defines the corrected image intensity to be 0. We studied the legitimacy of the inclusion of such a threshold value, and found that the threshold value is necessary, for the following reason. In a typical MRI image, pixels with lower SNR values are usually located in the background regions, and it is thus reasonable to set the corrected image intensities at such pixels to be 0. We performed many numerical studies, and found that inclusion of a threshold value T in the range of $[\sqrt{2}, 1.5]$ would generally improve the bias-corrected MRI images (see some numerical results reported in Section 4.3). According to Figure 4.2(b), "$T = \sqrt{2}$" corresponds to the corrected SNR $\widehat{m}(x,y)/\sigma$ of about 0.7, and "$T = 1.5$" corresponds to the corrected SNR of about 0.9. Based on all the considerations described above, we suggest using the following formula for approximating $g_1(s)$:

$$\widehat{g}_1(s) = \begin{cases} 0, & \text{when } s < T \\ \widehat{g}_{S,1}(s), & \text{when } T \leq s \leq 10 \\ s, & \text{when } s > 10. \end{cases} \tag{4.9}$$

Then, bias-correction can be accomplished by using (4.8), with g_1 replaced by $\widehat{g}_1$ defined above.

4.3 Numerical studies

In this section, we present some numerical examples to investigate the performance of our proposed bias-correction formulas (4.8)-(4.9), denoted as NEW. We compare the results with those of two commonly used ones in practice: the normal approximation formula (4.3), denoted as GP, and the moment-based formula (4.4), denoted as WD. Both formulas (4.3) and (4.4) are described in Section 4.1. Their performance is evaluated using three popular denoising procedures: the total variance (TV) procedure by Rudin et al. (1992), the anisotropic diffusion (AD) procedure by Perona and Malik (1990), and the optimized non-local means (ONLM) algorithm by Coupe et al. (2008b). For TV and AD, the MATLAB codes presented by Getreuer (2007) and Lopes (2007) are used. To evaluate the performance of a bias-correction method, we use the standard criterion of the estimated mean integrated squared error (MISE), defined as follows. Let $m(x,y)$ be the true image and $\widehat{m}(x,y)$ be the bias-corrected estimate of $m(x,y)$. Then, the integrated squared error is estimated by

$$\text{ISE} = \frac{1}{N} \sum_{(x,y)} \left[\widehat{m}(x,y) - m(x,y)\right]^2$$

where N denotes the total number of pixels and $\sum_{(x,y)}$ denotes the summation over all pixels. Then, MISE is estimated by the sample mean of the 100 ISE values com-

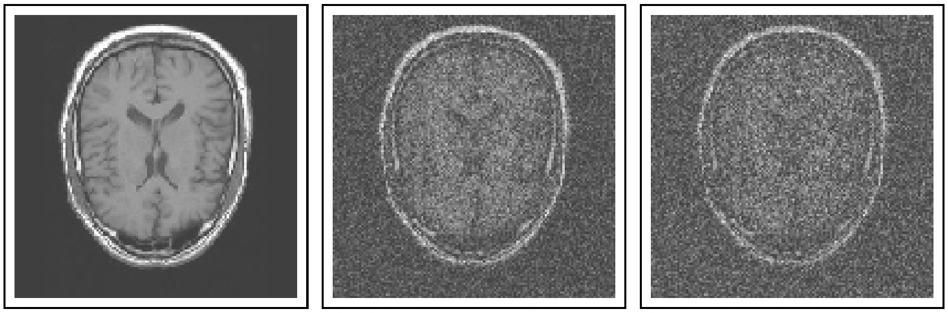

Figure 4.3: The true image and two noisy images with $\sigma = 40$ and 60 of the brain test image.

puted from 100 replicated simulations. By this criterion, the bias-correction method performs better if its MISE value is smaller. In the bias-correction literature, another popular criterion is the "contrast" of an image, defined to be the difference between the intensities of the brightest and the darkest pixels (Coupe et al., 2008b). The rationale of this criterion is that the estimation bias caused by the rician noise would generally decrease the image contrast. So, by this criterion, a bias-correction is better if the contrast of the bias-corrected image is larger. However, the contrast is sensitive to outliers. To make it robust to outliers, in this paper, we use the trimmed contrast, defined to be the contrast of an image, after the 1% largest and the 1% smallest intensities being deleted.

Two MRI images, one of a brain and another of an ankle, are used as test images. The brain image has 350×350 pixels, and the ankle image has 432×432 pixels. In both images, the intensity values range from 0 to 255. We then add rician noise of two different levels $\sigma = 40$ and $\sigma = 60$ to the images (4.2). The true and noisy images of the two test images are presented in Figures 4.3 and 4.4.

We then apply the denoising methods TV, AD, and ONLM to various test images, and then use the bias-correction methods NEW, GP, and WD to correct biases of the denoised images. For each pair of the denoising and bias-correction methods, the procedure parameters are chosen to minimize the estimated MISE value of the bias-

Figure 4.4: The true image and two noisy images with $\sigma = 40$ and 60 of the ankle test image.

corrected image. For the method NEW, four values of the threshold T are considered (4.9). They are 1.253, $\sqrt{2}$, 1.5, and 1.55, which correspond to the corrected SNR values of 0, 0.7, 0.9 and 1.0. The four versions with these four threshold values are denoted as NEW1, NEW2, NEW3, and NEW4, respectively. The estimated MISE values and their standard errors, and the sample mean values of the trimmed contrasts and their standard errors are presented in Tables 4.1 and 4.2.

From Table 4.1, we can have the following conclusions. First, the bias-correction procedure NEW3 is better than its peers GP and WD in all cases in terms of the MISE. Second, all four versions of NEW are better than GP and WD in all cases in terms of the trimmed contrast. Third, NEW2 and NEW3 are generally better than NEW1 and NEW4, and it seems that, NEW3 is slightly better when the noise level is relatively low (i.e., $\sigma = 40$), and NEW2 is slightly better when the noise level is relatively high (i.e., $\sigma = 60$) except the case with AD. So, in practice, we suggest choosing the threshold value T of our proposed bias-correction method to be in the range $[\sqrt{2}, 1.5]$. Similar conclusions can be obtained from the results in Table 4.2.

The bias-corrected denoised images of the noisy brain image with $\sigma = 40$ in Figure 4.3, are shown in Figure 4.5, in which, rows 1–3 present the denoised images by TV, AD and ONLM, and columns 1–3 present the bias-corrected denoised images by GP, WD and NEW3. Row 4 shows the deviation images defined to be (bias-

Table 4.1: Performance of the methods in the brain image example. In each cell, the first row presents estimated MISE value and its standard error, and the second row presents the estimated trimmed contrast and its standard error. The results are based on 100 random simulations.

	$\sigma = 40$			$\sigma = 60$		
Method	TV	AD	ONLM	TV	AD	ONLM
GP	607.7 (37.4)	608.0 (30.9)	592.7 (34.1)	1354.0 (95.0)	1312.0 (87.9)	1305.1 (81.6)
	204.5 (1.7)	217.3 (1.6)	224.2 (2.0)	182.6 (2.8)	199.3 (2.6)	212.2 (3.2)
WD	161.4 (6.5)	179.7 (3.7)	172.5 (5.2)	327.8 (26.1)	380.6 (28.1)	373.0 (29.8)
	224.2 (1.1)	224.8 (1.0)	234.5 (1.0)	210.6 (1.8)	207.2 (1.6)	227.3 (1.8)
NEW1	203.2 (38.9)	249.7 (31.9)	228.5 (32.2)	366.0 (75.6)	469.7 (70.4)	441.7 (74.6)
	227.5 (1.0)	227.7 (1.0)	237.6 (1.0)	218.1 (1.6)	215.1 (1.4)	234.4 (1.6)
NEW2	152.4 (5.5)	197.6 (23.3)	173.7 (15.4)	271.3 (23.4)	360.4 (34.9)	328.4 (28.1)
	227.6 (1.1)	228.1 (1.0)	237.6 (1.0)	218.3 (1.7)	214.8 (1.5)	234.4 (1.6)
NEW3	149.9 (2.3)	171.4 (9.0)	162.0 (4.3)	275.8 (12.0)	341.6 (11.4)	339.9 (16.4)
	227.4 (1.0)	227.9 (0.9)	237.5 (1.0)	217.9 (1.6)	214.7 (1.4)	234.2 (1.7)
NEW4	151.6 (2.4)	167.6 (3.5)	164.1 (3.4)	297.4 (20.9)	359.2 (17.0)	376.4 (29.3)
	227.5 (1.0)	227.9 (1.0)	237.6 (1.0)	218.2 (1.6)	215.0 (1.4)	234.4 (1.6)

Table 4.2: Performance of the methods in the ankle image example. In each cell, the first row presents estimated MISE value and its standard error, and the second row presents the estimated trimmed contrast and its standard error. The results are based on 100 random simulations.

	$\sigma = 40$			$\sigma = 60$		
Method	TV	AD	ONLM	TV	AD	ONLM
GP	531.9 (33.8) 219.4 (2.0)	540.9 (30.2) 231.3 (1.9)	538.8 (27.0) 228.5 (1.7)	1166.5 (81.3) 202.5 (2.4)	1166.9 (72.2) 220.0 (2.4)	1162.4 (67.2) 217.1 (2.8)
WD	198.9 (8.0) 238.6 (0.8)	215.0 (5.3) 238.4 (0.7)	222.7 (6.2) 238.7 (0.8)	381.1 (47.1) 231.0 (1.2)	410.4 (29.1) 231.8 (1.1)	433.2 (35.3) 233.1 (1.3)
NEW1	218.5 (26.0) 241.5 (0.8)	257.5 (22.3) 241.7 (0.7)	252.3 (21.5) 241.7 (0.8)	352.5 (55.1) 238.1 (1.0)	462.9 (55.2) 239.0 (0.8)	439.0 (53.0) 240.1 (1.2)
NEW2	187.7 (6.5) 241.8 (0.6)	229.5 (16.2) 241.6 (0.6)	221.0 (10.2) 241.7 (0.8)	297.8 (10.6) 238.2 (1.1)	390.1 (24.8) 239.0 (0.8)	380.2 (13.9) 240.1 (1.2)
NEW3	188.2 (2.2) 241.5 (0.7)	211.1 (6.7) 241.4 (0.6)	214.1 (2.9) 241.7 (0.8)	335.7 (25.7) 237.8 (1.0)	409.9 (18.7) 238.7 (0.9)	428.7 (30.9) 239.9 (1.2)
NEW4	192.8 (4.0) 241.7 (0.7)	210.5 (4.3) 241.6 (0.6)	217.3 (3.6) 241.7 (0.8)	401.5 (53.2) 238.0 (1.1)	464.0 (43.3) 238.9 (0.8)	495.1 (47.4) 240.1 (1.2)

corrected denoised image - true image) by the denoising procedure TV and by the bias-correction procedures GP, WD and NEW3, respectively. From the bias-corrected denoised images, it can be seen that the ones corrected by NEW3 are slightly sharper than the others. The deviation images show that the bias-corrected denoised image by NEW3 has the smallest deviation, compared the images corrected by the other two procedures, although the difference between the second and the third images is small. Similar conclusions can be made from the images shown in Figure 4.6.

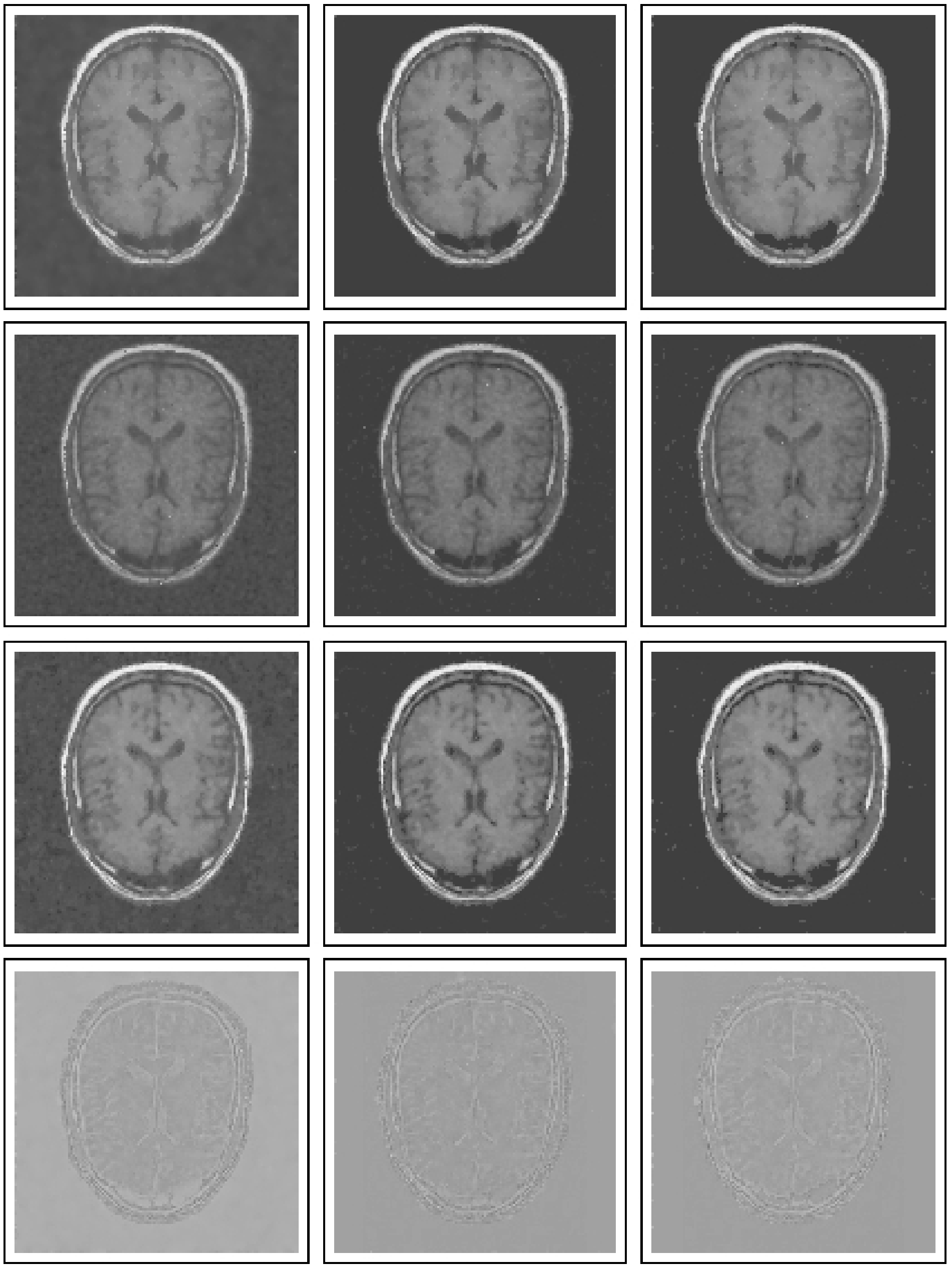

Figure 4.5: Bias-corrected denoised images in the angle image example with $\sigma = 40$. Rows 1-3 presents the denoised images by denoising procedures TV, AD and ONLM. Columns 1-3 presents the bias-corrected denoised images by procedures GP, WD and NEW3. Row 4 shows the bias images defined to be (bias-corrected denoised image - true image) by the denoising procedure TV and by the bias-correction procedures GP, WD and NEW3, respectively.

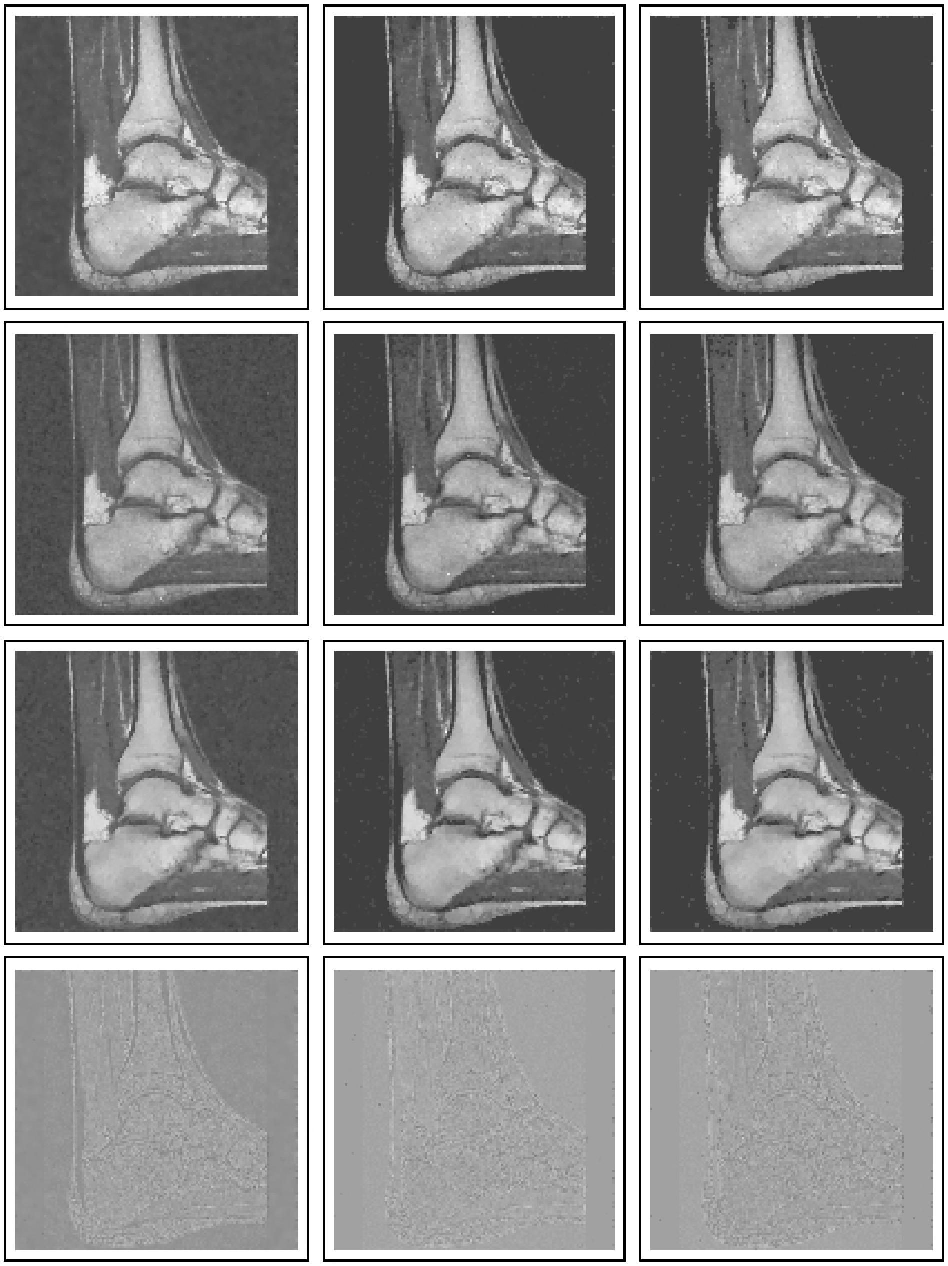

Figure 4.6: Bias-corrected denoised images in the angle image example with $\sigma = 40$. Rows 1-3 presents the denoised images by denoising procedures TV, AD and ONLM. Columns 1-3 presents the bias-corrected denoised images by procedures GP, WD and NEW3. Row 4 shows the bias images defined to be (bias-corrected denoised image - true image) by the denoising procedure TV and by the bias-correction procedures GP, WD and NEW3, respectively.

Chapter 5

Conclusions and Future Research

Edge-structure preserving image denoising is an important research area. In this dissertation, we have proposed three image denoising procedures, one for 2-D images, and two for 3-D images. Also, we proposed a bias correction method to efficiently denoise images that are corrupted with rician noise. In Section 5.1, we briefly summarize the proposed methods, and in Section 5.2, we discuss some related issues for future research.

5.1 Brief summary

We proposed an edge-structure preserving 2-D image denoising method in Chapter 2. Numerical studies show that it works well in various applications. In Chapter 3, we proposed an edge-structure preserving 3-D image denoising method. We also provided some statistical properties of that procedure. Numerical performances were also presented in comparison with some state-of-art methods, and they show that it works well in various applications. In Section 3.5, we modified that procedure to improve its performance. Numerical results of the modified method were also provided. In Chapter 4, we proposed an efficient bias correction method while denoising images that are corrupted with rician noise. This method is based on simple numerical simulation study, and it is convenient to use. After we apply a conventional image denoising method to remove the rician noise from an image, the denoised image would be biased for estimating the true image. When we apply the proposed bias correction method on that biased denoised image, we can get an unbiased estimate of the true image. Its numerical performance in comparison with some existing methods was investigated, and our numerical results show that it works well in various real applications.

5.2 Some future research problems

In Chapter 2, we have presented a 2-D image denoising procedure in the framework of jump regression analysis. The new procedure can efficiently preserve both the parts of edges with small curvature and the parts of edges with large curvature. From the construction of the proposed method, it can be seen that this method would not work well at places where two or more edge segments cross, because the true edge curves can not be approximated well by two half lines around a crossing point of several edge segments. It requires much future research to denoise images properly in such cases. Our proposed method consists of three steps. Although each step is based on local smoothing and the corresponding computation is thus fast, it is ideal to simplify the method by skipping or combining certain steps without sacrificing much of its denoising and edge-structure preserving ability.

As pointed out in Section 2.1, most existing image denoising procedures in the literature can not preserve certain edge-structures well. For instance, the anisotropic diffusion filters control the degree of smoothing around a given pixel by a nonhomogeneous diffusivity which is often chosen to be a decreasing function of an estimated gradient (cf., Section 7.5 of Qiu (2005)). So, if the given pixel is close to an edge curve, then the estimated gradient would be large and consequently there would be less smoothing around that pixel. However, in order to remove noise, some smoothing is still necessary around edge curves. Therefore, a certain degree of edge blurring is inevitable by the diffusion filters. Because the diffusion filters usually do not take the shape of the edge curves into consideration, they could not preserve certain edge structures (e.g., angles) well. These comments can also be applied to the bilateral filters (Tomasi and Manduchi, 1998), because Barash (2002) showed that bilateral filters were just special diffusion filters. Hillebrand and Müller (2007) recently demonstrated that the M-filter discussed by Chu et al. (1998) cannot handle isolated outliers well when denoising, and they proposed a modification to overcome that limitation by combining the ideas of robust estimation and trimming. Although proper selection of certain parameters needs to be further addressed, this modification makes a good contribution to the image denoising literature. The current denoising procedure requires explicit detection of JLSs before estimating the true image intensity function. In practice, it might be more convenient to use an image denoising procedure without explicit detection of the JLSs. Much future research is required to modify other existing denoising methods so that edge-structures can be better preserved while noise and other possible contaminations are better removed.

In case of the proposed 3-D image denoising methods, many of these aforesaid limitations also apply. Further, in the case of 3-D images, the structure of JLSs can be very complicated, and many possible edge-structures were not considered yet. All these are areas where much future research is necessary.

In cases of both 2-D and 3-D image denoising, although the proposed methods can preserve important edge-structure, they would blur some fine details in the images (e.g., blood vessels in a brain MRI). This is because we choose the bandwidth parameters globally. By doing so, the amount of smoothing in a continuity region (cf., Section 3.2) of an image is same as the amount of smoothing near regions of that image that contain fine details. To preserve such details, the shape and size of a related neighborhood should be determined locally. So, one possible way to accomplish this is to consider neighborhoods with various different shapes and sizes, at each pixel/voxel, and select the appropriate one at a given pixel/voxel by local information of image features. This is also an important future research area, which is related to multilevel local smoothing.

Moreover, various types of medical images contain different types of noise. Rician noise is one of them. We have proposed a bias correction method to deal with the rician noise. Much future research is necessary to remove many other types of non-zero mean, intensity-dependent noise.

References

Aja-Fernández, S., Alberola-López, C., and Westin, C. F. (2008). Noise and signal estimation in magnitude mri and rician distributed images: A lmmse approach. *IEEE Transactions on Image Processing*, 17(8):1383–1398.

Barash, D. (2002). A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:844–847.

Bernstein, M. A., Thomasson, D. M., and Perman, W. H. (1989). Improved detectability in low snr mr images by means of phase-corrected real construction. *IEEE Transactions on Image Processing*, 16:813–817.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems (with discussions). *Journal of the Royal Statistical Society (Series B)*, 36:192–236.

Besag, J. (1986). On the statistical analysis of dirty pictures (with discussion). *Journal of the Royal Statistical Society (Series B)*, 48:259–302.

Brownrigg, D. R. K. (1984). The weighted median filtering. *Communications of the ACM*, 27:807–818.

Buades, B., Coll, B., and Morel, J. M. (2005). A non-local algorithm for image denoising. *IEEE Computer Vision and Pattern Recognition*, 2:60–65.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 8:679–698.

Chabat, F., Yang, G. Z., and Hansell, D. M. (1999). A corner orientation detector. *Image and Vision Computing*, 17:761–769.

Chambolle, A. (2004). An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20:89–97.

Chan, S. G., Yu, B., and Vetterli, M. (2000). Spatially adaptive wavelet thresholding with context modeling for image denoising. *IEEE Transactions on Image Processing*, 9:1522–1531.

Chu, C. K., Glad, I. K., Godtliebsen, F., and Marron, J. S. (1998). Edge-preserving smoothers for image processing (with discussion). *Journal of the American Statistical Association*, 93:526–556.

Chung, K. L. (1974). *A Course In Probability Theory (2nd ed.)*. New York: Academic Press, Inc.

Clark, J. J. (1988). Singularity theory and phantom edges in scale space. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 10:720–727.

Coupe, P., Hellier, P., Prima, S., Kervrann, C., and Barillot, C. (2008a). 3d wavelet subbands mixing for image denoising. *International Journal of Biomedical Imaging*, 2008:1–11.

Coupe, P., Manjon, J. V., Gedamu, E., Arnold, D., Robles, M., and Collins, D. L. (2010). Robust rician noise estimation for mr images. *Medical Image Analysis*, 14(5):483–493.

Coupe, P., Yger, P., Prima, S., Hellier, P., Kervrann, C., and Barillot, C. (2008b). An optimized blockwise nonlocal means denoising filter for 3-d magnetic resonance images. *IEEE Transactions on Medical Imaging*, 27:425–441.

Epanechnikov, V. (1969). Non-parametric estimation of a multivariate probability density. *Theory of Probability and its Applications*, 14:153–158.

Fan, J. and Gijbels, I. (1996). *Local Polynomial Modelling and Its Applications*. Chapman & Hall: London.

Fessler, J. A., Erdogan, H., and Wu, W. B. (2000). Exact distribution of edge-preserving map estimators for linear signal models with gaussian measurement noise. *IEEE Transactions on image processing*, 9:1049–1055.

Figueiredo, A. A. T. and Nowak, R. D. (2001). Wavelet-based image estimation: an empirical bayes approach using jeffreys noninformative prior. *IEEE Transactions on Image Processing*, 10:1322–1331.

Fleck, M. M. (1992). Some defects in finite-difference edge finders. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 14:337–345.

Gallagher, Jr., N. C. and Wise, G. L. (1981). A theoretical analysis of the properties of median filtering. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29:1136–1141.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

Getreuer, P. (2007). tvdenoise.m (a matlab program). `http://www.mathworks.nl/matlabcentral/fileexchange/16236`.

Gijbels, I., Lambert, A., and Qiu, P. (2006). Edge-preserving image denoising and estimation of discontinuous surfaces. *EEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1075–1087.

Godtliebsen, F. and Sebastiani, G. (1994). Statistical methods for noisy images with discontinuities. *Journal of Applied Statistics*, 21:459–476.

Gonzalez, R. C. and Woods, R. E. (1992). *Digital Image Processing*. Addison-Wesley Publishing Company, Inc.

Gudbjartsson, H. and Patz, S. (1995). The noisy distribution of noisy mri data. *Magnetic Resonance in Medicine*, 34:910–915.

Hall, P. and Qiu, P. (2007). Blind deconvolution and deblurring in image analysis. *Statistica Sinica*, 17:1483–1509.

Hall, P. and Robinson, A. P. (2009). Reducing variability of crossvalidation for smoothing-parameter choice. *Biometrika*, 96(1):175–186.

Hillebrand, M. and Müller, C. H. (2007). Outlier robust corner-preserving methods for reconstructing noisy images. *The Annals of Statistics*, 35:132–165.

Hostalkova, E., Oldrich, V., and Prochazka, A. (2007). Multi-dimensional biomedical image de-noising using haar transform. In *Proceedings of the 15th International Conference on Digital Signal Processing*, pages 175–178, Cardiff, UK.

Keeling, S. (2003). Total variation based convex filters for medical imaging. *Applied Mathematics and Computation*, 139(1):101–119.

Kristoffersen, A. (2007). Optimal estimation of the diffusion coefficient from non-averaged and averaged noisy magnitude data. *Journal of Magnetic Resonance*, 187:293–305.

Li, S. Z. (1995). On discontinuity-adaptative smoothness prior in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:576–586.

Loader, C. R. (1999). Bandwidth selection: classical or plug-in? *The Annals of Statistics*, 27(2):415–438.

Lopes, D. S. (2007). anisodiff3d.m (a matlab program). `http://www.mathworks.co.uk/matlabcentral/fileexchange/14995-anisotropic-diffusion-perona-malik`.

Lu, H., Jui-Hsi, C., Han, G., Li, L., and Liang, Z. (2001). A 3d distance-weighted wiener filter for poisson noise reduction in sinogram space for spect imaging. In *SPIE proceedings series*, volume 4320, pages 905–913.

Macovski, A. (1996). Noise in mri. *Magnetic Resonance in Medicine*, 36:494–497.

Marron, J. S. (1988). Automatic smoothing parameter selection: a survey. *Empirical Economics*, 13(3-4):187–208.

Marroquin, J. L., Velasco, F. A., Rivera, M., and Nakamura, M. (2001). Gauss-markov measure field models for low-level vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:337–347.

Moussouris, J. (1974). Gibbs and markov systems with constraints. *Journal of Statistical Physics*, 10:11–33.

Nason, G. and Silverman, B. (1994). The discrete wavelet transform in s. *Journal of Computational and Graphical Statistics*, 3:163–191.

Nishii, R. (1984). Asymptotic properties of criteria for selection of variables in multiple regression. *The Annals of Statistics*, 12:758–765.

Nowak, R. (1999). Wavelet-based rician noise removal for magnetic resonance imaging. *IEEE Transactions on Image Processing*, 8:1408–1419.

Perona, P. and Malik, J. (1990). Scale space and edge detection algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639.

Polzehl, J. and Spokoiny, V. G. (2000). Adaptive weights smoothing with applications to image restoration. *Journal of the Royal Statistical Society (Series B)*, 62:335–354.

Portilla, J., Strela, V., Wainwright, M. J., and Simoncelli, E. P. (2003). Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12:1338–1351.

Qiu, P. (1998). Discontinuous regression surfaces fitting. *The Annals of Statistics*, 26:2218–2245.

Qiu, P. (2002). A nonparametric procedure to detect jumps in regression surfaces. *Journal of Computational and Graphical Statistics*, 11:799–822.

Qiu, P. (2004). The local piecewisely linear kernel smoothing procedure for fitting jump regression surfaces. *Technometrics*, 46:87–98.

Qiu, P. (2005). *Image Processing and Jump Regression Analysis.* New York: John Wiley & Sons.

Qiu, P. (2007). Jump surface estimation, edge detection, and image restoration. *Journal of American Statistical Association*, 102:745–756.

Qiu, P. (2009). Jump-preserving surface reconstruction from noisy data. *Annals of the Institute of Statistical Mathematics*, 61:715–751.

Qiu, P. and Bhandarkar, S. M. (1996). An edge detection technique using local smoothing and statistical hypothesis testing. *Pattern Recognition Letters*, 17:849–872.

Qiu, P. and Yandell, B. (1997). Jump detection in regression surfaces. *Journal of Computational and Graphical Statistics*, 6(3):332–354.

Rajan, J., Poot, D., Juntu, J., and Sijbers, J. (2010). Noise measurement from magnitude mri using local estimates of variance and skewness. *Physics in Medicine and Biology*, 55:441–449.

Rivera, M. and Marroquin, J. L. (2002). Adaptive rest condition potentials: first and second order edge-preserving regularization. *Journal of Computer Vision and Image Understanding*, 88:76–93.

Rudin, L., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268.

Saint-Marc, P., Chen, J., and Medioni, G. (1991). Adaptive smoothing: a general tool for early vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:514–529.

Seber, G. A. F. (1977). *Linear Regression Analysis.* New York: John Wiley.

Sijbers, J. and den Dekker, A. J. (2004). Maximum likelihood estimation of signal amplitude and noise variance from mr data. *Magnetic Resonance in Medicine*, 51:586–594.

Sijbers, J., den Dekker, A. J., Scheunders, P., and Dyck, D. V. (1998). Maximum-likelihood estimation of rician distribution parameters. *IEEE Transactions on Medical Imaging*, 17:357–361.

Sinha, S. S. and Schunck, B. G. (1992). A two-stage algorithm for discontinuity-preserving surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:36–55.

Sonka, M., Hlavac, V., and Boyle, R. (2008). *Image Processing, Analysis, and Machine Vision (3rd ed.).* Toronto, Thomson Learning.

Sun, J. and Qiu, P. (2007). Jump detection in regression surfaces using both first-order and second-order derivatives. *Journal of Computational and Graphical Statistics*, 16:289–311.

Sun, T., Gabbouj, M., and Neuvo, Y. (1994). Center weighted median filters: some properties and their applications in image processing. *Signal Processing*, 35:213–229.

Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Proceedings of the 1998 IEEE International Conference on Computer Vision*, pages 839–846, Bombay, India.

Torre, V. and Poggio, T. O. (1986). On edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 8:147–163.

Vlaardingerbroek, M. T. and den Boer, J. A. (1999). *Magnetic Resonance Imaging (2nd ed.).* New York: Springer-Verlag.

Wang, Y. and Zhou, H. (2006). Total variation wavelet-based medical image denoising. *International Journal of Biomedical Imaging*, 2006:1–6.

Webb, S. (1988). *The Physics of Medical Imaging.* Bristol, U.K.: Adam Hilger.

Weickert, J., Romeni, B.M., t., and Viergever, M. A. (1998). Efficient and reliable schemes for nonlinear filtering. *IEEE Transactions on Image Processing*, 7(3):398–410.

Wiest-Daessle, N., Prima, S., Coupe, P., Morrissey, S., and Barillot, C. (2008). Rician noise removal by non-local means filtering for low signal-to-noise ratio mri: Application to dt-mri. In *Lecture notes in Computer Science*, pages 171–179. Proceedings of MICCAI, New York/USA.

Woiselle, A., Starck, J. L., and Fadili, J. (2008). 3d image restoration with the curvelet transform. In *2008 Proceedings of astronomical data analysis V*, Heraklion (Crete).

Yang, G. Z., Burger, P., Firmin, D. N., and Underwood, S. R. (1996). Structure adaptative anisotropic image filtering. *Image and Vision Computing*, 14:135–145.

Yi, J. H. and Chelberg, D. M. (1995). Discontinuity-preserving and viewpoint invariant reconstruction of visible surfaces using a first order regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:624–629.

Zhu, H. T., Li, Y., Ibrahim, J. G., Shi, X., An, H., Chen, Y., Gao, W., Lin, W., Rowe, D. B., and Peterson, B. S. (2009). Regression models for identifying noise sources in magnetic resonance images. *Journal of American Statistical Association*, 104:623–637.

Appendix A

Proofs and Derivations

A.1 Proof of Theorem 3.3.1

We begin with two Lemmas. For simplicity, in the appendix, we use "$\sum$" to denote the sum over all design points $\{(x_i, y_j, z_k),\ \ i, j, k = 1, 2, \ldots, n\}$, unless otherwise mentioned.

***Lemma A.1*:** Under the conditions stated in Theorem 3.3.1, we have, for $i_1, i_2, i_3 = 0, 1, 2$,

$$\left\| \frac{1}{n^3 (h_n^*)^3} \sum \left(\frac{x_i - x}{h_n^*}\right)^{i_1} \left(\frac{y_j - y}{h_n^*}\right)^{i_2} \left(\frac{z_k - z}{h_n^*}\right)^{i_3} K\left(\frac{x_i - x}{h_n^*}, \frac{y_j - y}{h_n^*}, \frac{z_k - z}{h_n^*}\right) - \nu_{i_1 i_2 i_3} \right\|_{\Omega_{h_n^*}} = O\left(\frac{1}{n h_n^*}\right)$$

and

$$\left\| \frac{1}{n^3 h^3} \sum \varepsilon_{ijk} K\left(\frac{x_i - x}{h}, \frac{y_j - y}{h}, \frac{z_k - z}{h}\right) \right\|_{\Omega_{h_n^*}} = o\left(\frac{\beta_n \log(n)}{nh}\right) \quad a.s.,$$

where $\nu_{i_1 i_2 i_3} = \int\int\int u^{i_1} v^{i_2} s^{i_3} K(u, v, s)\, dudvds$, for $i_1, i_2, i_3 = 0, 1, 2$. ■

Proof of *Lemma A.1*: This is a straightforward generalization of Proposition 2 in Qiu (2009) from 2-D to 3-D cases. ■

Lemma A.2: Under the conditions in Theorem 3.3.1, we have

$$\begin{aligned}
\|\widehat{a}-f\|_{\Omega_{\bar{J},h_n^*}} &= O\left((h_n^*)^2\right)+o\left(\frac{\beta_n \log n}{nh_n^*}\right) \quad a.s.\\
\|\widehat{b}-f'_x\|_{\Omega_{\bar{J},h_n^*}} &= O\left(h_n^*\right)+o\left(\frac{\beta_n \log n}{n(h_n^*)^2}\right) \quad a.s.\\
\|\widehat{c}-f'_y\|_{\Omega_{\bar{J},h_n^*}} &= O\left(h_n^*\right)+o\left(\frac{\beta_n \log n}{n(h_n^*)^2}\right) \quad a.s.\\
\|\widehat{d}-f'_z\|_{\Omega_{\bar{J},h_n^*}} &= O\left(h_n^*\right)+o\left(\frac{\beta_n \log n}{n(h_n^*)^2}\right) \quad a.s.
\end{aligned}$$

If $(x,y,z) \in J_{h_n^*} \backslash S_\epsilon$, then we have

$$\begin{pmatrix} \widehat{a}(x,y,z) \\ \widehat{b}(x,y,z) \\ \widehat{c}(x,y,z) \\ \widehat{d}(x,y,z) \end{pmatrix} = \begin{pmatrix} f_-(x,y,z) \\ f'_x(\tilde{x},\tilde{y},\tilde{z}) \\ f'_y(\tilde{x},\tilde{y},\tilde{z}) \\ f'_z(\tilde{x},\tilde{y},\tilde{z}) \end{pmatrix} +$$

$$\begin{pmatrix} \phi_0(x,y,z)C(x,y,z)+O((h_n^*)^2)+o(\frac{\beta_n \log n}{nh_n^*}) \\ \phi_1(x,y,z)C(x,y,z)+\gamma_1(x,y,z)C_x(x,y,z)+O(h_n^*)+o(\frac{\beta_n \log n}{n(h_n^*)^2}) \\ \phi_2(x,y,z)C(x,y,z)+\gamma_2(x,y,z)C_y(x,y,z)+O(h_n^*)+o(\frac{\beta_n \log n}{n(h_n^*)^2}) \\ \phi_3(x,y,z)C(x,y,z)+\gamma_3(x,y,z)C_z(x,y,z)+O(h_n^*)+o(\frac{\beta_n \log n}{n(h_n^*)^2}) \end{pmatrix} \quad a.s.$$

where $f_-(x,y,z)$ is the smaller one of the two one-sided (due to JLS) limits of f at (x,y,z), $(\tilde{x},\tilde{y},\tilde{z})$ is some point around (x,y,z) that satisfies (i) it is a continuity point of f that is on the same side of the JLS as (x,y,z), and (ii) $d_E((\tilde{x},\tilde{y},\tilde{z}),(x,y,z)) \sim O(1/n)$, $C(x,y,z)$, $C_x(x,y,z)$, $C_y(x,y,z)$, $C_z(x,y,z)$ are absolute jump magnitudes of $f(x,y,z)$ and its first order x, y and z partial derivatives, $\phi_1(x,y,z)$, $\phi_2(x,y,z)$ and $\phi_3(x,y,z)$ are three constants satisfying

$$\sqrt{\phi_1^2(x,y,z)+\phi_2^2(x,y,z)+\phi_3^2(x,y,z)} = O(1/h_n^*) \quad a.s.,$$

$\gamma_1(x,y,z)$, $\gamma_2(x,y,z)$ and $\gamma_3(x,y,z)$ are three constants between -1 and 1, and

$\phi_0(x,y,z)$ is some constant between 0 and 1. ■

Proof of *Lemma A.2*: When $(x,y,z) \in \Omega_{\bar{J},h_n^*}$, by the Taylor's expansion, for any $(x_i, y_j, z_k) \in O^*(x,y,z)$, we have

$$\begin{aligned}
\xi_{ijk} &= f(x_i,y_j,z_k) + \varepsilon_{ijk} \\
&= f(x,y,z) + (x_i - x)f'_x(x,y,z) + (y_j - y)f'_y(x,y,z) + (z_k - z)f'_z(x,y,z) \\
&+ O((h_n^*)^2) + \varepsilon_{ijk}
\end{aligned}$$

So, we have

$$\begin{pmatrix} \sum \xi_{ijk}K_{ijk} \\ \sum \xi_{ijk}(x_i - x)K_{ijk} \\ \sum \xi_{ijk}(y_j - y)K_{ijk} \\ \sum \xi_{ijk}(z_k - z)K_{ijk} \end{pmatrix} = \begin{pmatrix} w_{000} & w_{100} & w_{010} & w_{001} \\ w_{100} & w_{200} & w_{110} & w_{101} \\ w_{010} & w_{110} & w_{020} & w_{011} \\ w_{001} & w_{101} & w_{011} & w_{002} \end{pmatrix} \begin{pmatrix} f(x,y,z) \\ f'_x(x,y,z) \\ f'_y(x,y,z) \\ f'_z(x,y,z) \end{pmatrix} + \begin{pmatrix} \sum O((h_n^*)^2)K_{ijk} + \sum \varepsilon_{ijk}K_{ijk} \\ \sum O((h_n^*)^2)(x_i - x)K_{ijk} + \sum \varepsilon_{ijk}(x_i - x)K_{ijk} \\ \sum O((h_n^*)^2)(y_j - y)K_{ijk} + \sum \varepsilon_{ijk}(y_j - y)K_{ijk} \\ \sum O((h_n^*)^2)(z_k - z)K_{ijk} + \sum \varepsilon_{ijk}(z_k - z)K_{ijk} \end{pmatrix}. \tag{A.1}$$

By (3.3), (A.1), and *Lemma A.1*, we have

$$\begin{pmatrix} \widehat{a}(x,y,z) \\ \widehat{b}(x,y,z) \\ \widehat{c}(x,y,z) \\ \widehat{d}(x,y,z) \end{pmatrix} = \begin{pmatrix} f(x,y,z) \\ f'_x(x,y,z) \\ f'_y(x,y,z) \\ f'_z(x,y,z) \end{pmatrix} + \begin{pmatrix} O((h_n^*)^2) + o(\frac{\beta_n \log n}{nh_n^*}) \\ O(h_n^*) + o(\frac{\beta_n \log n}{n(h_n^*)^2}) \\ O(h_n^*) + o(\frac{\beta_n \log n}{n(h_n^*)^2}) \\ O(h_n^*) + o(\frac{\beta_n \log n}{n(h_n^*)^2}) \end{pmatrix} a.s. \tag{A.2}$$

Under the conditions in Theorem 3.3.1, it is clear that (A.2) is uniformly true for $(x,y,z) \in \Omega_{\bar{J},h_n^*}$. This an easy application of *Lemma A.1*.

Now, if $(x,y,z) \in J_{h_n^*} \backslash S_\epsilon$ and n is large enough so that $h_n^* < \epsilon$, then $O^*(x,y,z)$ is divided into two parts I_1 and I_2 by the JLS. Without loss of generality, let us assume that there is a positive jump from I_1 to I_2 at (x,y,z). Then, when $(x_i,y_j,z_k) \in I_1$, we have

$$\begin{aligned}
\xi_{ijk} &= f(x_i,y_j,z_k) + \varepsilon_{ijk} \\
&= f_-(x,y,z) + (x_i - x)f'_x(\tilde{x},\tilde{y},\tilde{z}) + (y_j - y)f'_y(\tilde{x},\tilde{y},\tilde{z}) + (z_k - z)f'_z(\tilde{x},\tilde{y},\tilde{z}) \\
&+ O((h_n^*)^2) + \varepsilon_{ijk}
\end{aligned}$$

Similarly, when $(x_i,y_j,z_k) \in I_2$, we have

$$\begin{aligned}
\xi_{ijk} &= f(x_i,y_j,z_k) + \varepsilon_{ijk} \\
&= f_-(x,y,z) + (x_i - x)f'_x(\tilde{x},\tilde{y},\tilde{z}) + (y_j - y)f'_y(\tilde{x},\tilde{y},\tilde{z}) + (z_k - z)f'_z(\tilde{x},\tilde{y},\tilde{z}) \\
&+ C(x,y,z) + (x_i - x)C_x(x,y,z) + (y_j - y)C_y(x,y,z) + (z_k - z)C_z(x,y,z) \\
&+ O((h_n^*)^2) + \varepsilon_{ijk}
\end{aligned}$$

where $(\tilde{x},\tilde{y},\tilde{z})$ is some point in I_1 that satisfies the conditions stated in *Lemma A.2*. By (3.3) and the above two expressions, we have

$$\begin{pmatrix} \widehat{a}(x,y,z) \\ \widehat{b}(x,y,z) \\ \widehat{c}(x,y,z) \\ \widehat{d}(x,y,z) \end{pmatrix} = \begin{pmatrix} f_-(x,y,z) \\ f'_x(\tilde{x},\tilde{y},\tilde{z}) \\ f'_y(\tilde{x},\tilde{y},\tilde{z}) \\ f'_z(\tilde{x},\tilde{y},\tilde{z}) \end{pmatrix} +$$

$$\begin{pmatrix} \phi_0(x,y,z)C(x,y,z) + O((h_n^*)^2) + o(\frac{\beta_n \log n}{n h_n^*}) \\ \phi_1(x,y,z)C(x,y,z) + \gamma_1(x,y,z)C_x(x,y,z) + O(h_n^*) + o(\frac{\beta_n \log n}{n (h_n^*)^2}) \\ \phi_2(x,y,z)C(x,y,z) + \gamma_2(x,y,z)C_y(x,y,z) + O(h_n^*) + o(\frac{\beta_n \log n}{n (h_n^*)^2}) \\ \phi_3(x,y,z)C(x,y,z) + \gamma_3(x,y,z)C_z(x,y,z) + O(h_n^*) + o(\frac{\beta_n \log n}{n (h_n^*)^2}) \end{pmatrix} a.s.$$

where

$$
\begin{aligned}
\phi_0(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\phi_1(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (x_i-x)K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (x_i-x)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\phi_2(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (y_j-y)K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (y_j-y)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\phi_3(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (z_k-z)K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (z_k-z)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\gamma_1(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (x_i-x)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (x_i-x)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\gamma_2(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (y_j-y)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (y_j-y)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}\\
\gamma_3(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in I_2} (z_k-z)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}{\sum (z_k-z)^2 K(\frac{x_i-x}{h_n^*},\frac{y_j-y}{h_n^*},\frac{z_k-z}{h_n^*})}.
\end{aligned}
$$

From the above expressions, it is obvious that $\gamma_1(x,y,z)$, $\gamma_2(x,y,z)$ and $\gamma_3(x,y,z)$ are constants between 0 and 1, and $\phi_0(x,y,z)$ is a constant between 0 and 1. Without loss of generality, let $C_x(x,y,z)$, $C_y(x,y,z)$ and $C_z(x,y,z)$ denote absolute jump magnitudes of f'_x, f'_y and f'_z, then $\gamma_1(x,y,z)$, $\gamma_2(x,y,z)$ and $\gamma_3(x,y,z)$ are constants between -1 and 1. By similar arguments to those in Qiu and Yandell (1997) it is not difficult to check that $\sqrt{\phi_1(x,y,z)^2+\phi_2(x,y,z)^2+\phi_3(x,y,z)^2} = O(1/h_n^*)$ a.s. So, *Lemma A.2* is proved. ■

Proof of Theorem 3.3.1: For a design point $(x,y,z)\in\Omega_{\bar{S},\epsilon}$, if it is more than h_n^* away from any JLS, then at least one of $O^*(x_{N_1},y_{N_1},z_{N_1})$ and $O^*(x_{N_2},y_{N_2},z_{N_2})$ is located in a same continuous region as (x,y,z). So, we have

$$\delta(x,y,z) \le \|\widehat{\boldsymbol{\beta}}(x,y,z)-\widehat{\boldsymbol{\beta}}_{N_1}(x,y,z)\| = O\left(h_n^*\right)+o\left(\frac{\beta_n\log n}{n(h_n^*)^2}\right) \quad a.s.$$

The above expression is a direct conclusion of *Lemma A.2.* Using the fact that $\chi^2_{3,\alpha_n} = O(-\log\alpha_n)$, the expression (3.8) and *Lemma A.1*, it is not difficult to check that the threshold value $u_n = O(\frac{n\sqrt{-\log\alpha_n}}{(nh_n^*)^{5/2}})$ a.s. The fact that $\chi^2_{3,\alpha_n} = O(-\log\alpha_n)$ can be proved easily by using $\chi^2_{3,\alpha_n} \le 3\chi^2_{1,\alpha_n/3}$ and the Mill's inequality regarding normal tail probabilities. So, under the condition that $\frac{(nh_n^*)^{7/2}}{n^2\sqrt{-\log\alpha_n}} = o(1)$, we have $\frac{\delta(x,y,z)}{u_n} = o(1)$ a.s. resulting $\delta(x,y,z) < u_n$ a.s. (i.e., (x,y,z) is not detected as an edge point) when n is large enough, and this is uniformly true for all $(x,y,z) \in \Omega_{\bar{S},\epsilon} \cap \Omega_{\bar{J},h_n^*}$. Therefore,

$$\sup_{(x,y,z)\in\widehat{D}_n\bigcap\Omega_{\bar{S},\epsilon}} \inf_{(x',y',z')\in D\bigcap\Omega_{\bar{S},\epsilon}} d_E((x,y,z)^T,(x',y',z')^T) = O(h_n^*) \;\; a.s. \tag{A.3}$$

On the other hand, if (x,y,z) is a non-singular point on a JLS, then by *Lemma A.2*, we have

$$\begin{aligned}\delta(x,y,z) \sim \quad & C(x,y,z)\sqrt{\phi_1(x,y,z)^2+\phi_2(x,y,z)^2+\phi_3(x,y,z)^2} + O(h_n^*) + \\ & o(\frac{\beta_n\log n}{n(h_n^*)^2}) \;\; a.s.\end{aligned}$$

Since $\sqrt{\phi_1(x,y,z)^2+\phi_2(x,y,z)^2+\phi_3(x,y,z)^2} = O(1/h_n^*)$ a.s., by the the condition that $\sqrt{-\log\alpha_n/(nh_n^*)^3} = o(1)$, we have $\delta(x,y,z) > u_n$ a.s. So, (x,y,z) would be detected as an edge pixel when n is large enough. Since $\min_{(x,y,z)\in D\bigcap\Omega_{\bar{S},\epsilon}} C(x,y,z) > 0$ (see the definition of *singular points* in Section 3.3), the above result is uniformly true for $(x,y,z) \in D\bigcap\Omega_{\bar{S},\epsilon}$. Therefore,

$$\sup_{(x,y,z)\in D\bigcap\Omega_{\bar{S},\epsilon}} \inf_{(x',y',z')\in\widehat{D}_n\bigcap\Omega_{\bar{S},\epsilon}} d_E((x,y,z)^T,(x',y',z')^T) = O(h_n^*) \;\; a.s. \tag{A.4}$$

By (A.3) and (A.4), Theorem 3.3.1 is proved. ■

A.2 Proof of Theorem 3.3.2

We begin with three Lemmas.

Lemma A.3: Besides the conditions in Theorem 3.3.1, let us further assume that $(x, y, z) \in J_{h_n} \backslash S_\epsilon$, the JLS has unique tangent plane at (x_*, y_*, z_*), the point on D that is closest to (x, y, z), and the bandwidth h_n satisfies the conditions that $h_n = o(1)$, $1/(nh_n) = o(1)$, and $h_n^*/h_n = o(1)$. Then, the local plane fitted by the algorithm (approximation of the JLS by a local plane) in Section 3.2.2 converges almost surely to the tangent plane of the JLS at (x_*, y_*, z_*) both in normal direction and pointwise as $n \to \infty$. ■

Proof of *Lemma A.3*: Assume that the normal direction of the tangent plane of the JLS at (x_*, y_*, z_*) is $(\rho_{x_*}, \rho_{y_*}, \rho_{z_*})^T$ with $\sqrt{\rho_{x_*}^2 + \rho_{y_*}^2 + \rho_{z_*}^2} = 1$. Without loss of generality, we further assume that $(\rho_{x_*}, \rho_{y_*}, \rho_{z_*})^T = (0, 0, 1)^T$ and that n is large enough so that $h_n < \epsilon$. So, the equation of the tangent plane of the JLS at (x_*, y_*, z_*) is $z = z_*$ and any nonsingular point on the JLS in $O(x, y, z)$ satisfies $z = z_* + O(h_n^2)$. Therefore, the gradient direction at any point on JLS in $O(x, y, z)$ can be written as $(O(h_n^2), O(h_n^2), 1 + O(h_n^2))^T$. Now, if $(x^*, y^*, z^*) \in \left(J_{h_n^*} \backslash S_\epsilon\right) \cap O(x, y, z)$, then

$$\widehat{\boldsymbol{\beta}}^*(x^*, y^*, z^*) = \left(\frac{\widehat{b}(x^*, y^*, z^*)}{\|\widehat{\boldsymbol{\beta}}(x^*, y^*, z^*)\|}, \frac{\widehat{c}(x^*, y^*, z^*)}{\|\widehat{\boldsymbol{\beta}}(x^*, y^*, z^*)\|}, \frac{\widehat{d}(x^*, y^*, z^*)}{\|\widehat{\boldsymbol{\beta}}(x^*, y^*, z^*)\|} \right)^T.$$

The expressions of $\widehat{b}(x^*, y^*, z^*)$, $\widehat{c}(x^*, y^*, z^*)$, $\widehat{d}(x^*, y^*, z^*)$ and $\widehat{\boldsymbol{\beta}}(x^*, y^*, z^*)$ can be obtained from *Lemma A.2*. From *Lemma A.1*, it is easy to check that $\phi_3(x^*, y^*, z^*) \sim \frac{O(n^3(h_n^*)^4)}{O(n^3(h_n^*)^5)} = O(\frac{1}{h_n^*})$ a.s. Since the gradient direction at any point of the JLS in $O(x, y, z)$ can be written as $(O(h_n^2), O(h_n^2), 1 + O(h_n^2))^T$, from the expressions of $\phi_1(x^*, y^*, z^*)$ and $\phi_2(x^*, y^*, z^*)$ in *Lemma A.2*, we can see that both of them are of the order $\frac{O(n^3(h_n^*)^3 h_n^2 h_n^*)}{O(n^3(h_n^*)^5)} = O\left(\frac{h_n^2}{h_n^*}\right)$ a.s. Then, we have $\widehat{\boldsymbol{\beta}}^*(x^*, y^*, z^*) = (O(h_n^2), O(h_n^2), 1 + O(h_n^2))^T$ a.s. So, the matrix $G = (g_{i_1,i_2}, i_1, i_2 = 1, 2, 3)$ defined in (3.5) has the property that

$$
\begin{aligned}
g_{i_1,i_2} &= O(h_n^2) \ \ a.s., \text{ if } (i_1,i_2) \neq (3,3), \\
g_{3,3} &= 1 + O(h_n^2) \ \ a.s.
\end{aligned}
$$

Since G is a real symmetric matrix, the eigenvalues of G are $\tau + 2\sqrt{p}\cos\gamma$, $\tau - \sqrt{p}(\cos\gamma + \sqrt{3}\sin\gamma)$ and $\tau - \sqrt{p}(\cos\gamma - \sqrt{3}\sin\gamma)$, where $3\tau = trace(G)$, $2q = det(G - \tau I)$, $6p$ is the sum of squares of the elements of $(G - \tau I)$, and $\gamma = \frac{1}{3}\tan^{-1}\frac{\sqrt{p^3 - q^2}}{q}$ with $0 \leq \gamma \leq \pi$. Therefore, we have, $\tau = \frac{1}{3} + O(h_n^2)$ a.s., $6p = \frac{2}{3} + O(h_n^2)$ a.s., $q = \frac{1}{27} + O(h_n^2)$ a.s. and $\gamma = O(h_n^2)$ a.s. and thus the three eigenvalues of G are of the orders $1 + O(h_n)$, $O(h_n)$, and $O(h_n)$, a.s., respectively. The eigenvector corresponding to a eigenvalue λ can be found by finding the solution for $\mathbf{e} = (e_1, e_2, e_3)^T$ from the two equations $G\mathbf{e} = \lambda\mathbf{e}$ and $\mathbf{e}'\mathbf{e} = 1$. If $\lambda = \lambda_1$ with λ_1 being the largest eigenvalue of G, we have

$$
\begin{aligned}
e_1 &= \eta\left(\frac{g_{1,2}g_{2,3} - g_{1,3}(g_{2,2} - \lambda_1)}{(g_{1,1} - \lambda_1)(g_{2,2} - \lambda_1) - g_{1,2}g_{2,1}}\right), \\
e_2 &= \eta\left(\frac{g_{1,3}g_{2,1} - g_{2,3}(g_{1,1} - \lambda_1)}{(g_{1,1} - \lambda_1)(g_{2,2} - \lambda_1) - g_{1,2}g_{2,1}}\right), \\
e_3 &= \eta,
\end{aligned}
$$

where η is such that $e_1^2 + e_2^2 + e_3^2 = 1$. After combining this result and the results about the elements of G and λ_1, we have $e_1 = O(h_n)$ a.s., $e_2 = O(h_n)$ a.s. and $e_3 = 1 + O(h_n)$ a.s., from which we have $\mathbf{e}$ converges to $(0,0,1)^T$ a.s., as $n \to \infty$. Therefore, the normal direction of the fitted plane by the algorithm (approximation of the JLS by a local plane) in Section 3.2.2 converges to the normal direction of the JLS at (x_*, y_*, z_*) almost surely. From Theorem 3.3.1, it is clear that the center of $\widehat{D}_n \cap O(x,y,z)$ converges almost surely to some point on the tangent plane of the JLS at (x_*, y_*, z_*), and the convergence rate is $O(h_n)$. This concludes the proof of *Lemma*

A.3.

Lemma A.4*:* Besides the conditions in Theorem 3.3.1, let us further assume that $(x, y, z) \in J_{h_n} \backslash S_\epsilon$, the JLS has two different one-sided tangent planes at some point $(x_*, y_*, z_*) \in O(x, y, z)$, and the bandwidth h_n satisfies the conditions that $h_n = o(1)$, $1/(nh_n) = o(1)$, and $h_n^*/h_n^3 = o(1)$. Then, the two half-planes fitted by the algorithm (approximation of the JLS by two crossing half-planes) in Section 3.2.2 converges almost surely to the two one-sided tangent planes of the JLS at (x_*, y_*, z_*).

Proof of *Lemma A.4*: Without loss of generality we can assume that two different one-sided tangent planes at (x_*, y_*, z_*) are $x = x_*$ and $y - y_* = \kappa(x - x_*)$ where κ is a constant, which are labeled P_1 and P_2 respectively. They intersect at a straight line L and clearly $(x_*, y_*, z_*) \in L$. Let us consider a plane S that passes L and separates P_1 and P_2 in $O(x, y, z)$. Then, S divides $O(x, y, z)$ into two parts $N_1(x, y, z)$ and $N_2(x, y, z)$, and it divides $D \cap O(x, y, z)$ into two parts $D \cap N_1(x, y, z)$ and $D \cap N_2(x, y, z)$. From the proof of *Lemma A.3*, we see that if a point (x^*, y^*, z^*) is more than h_n^* away from L and $(x^*, y^*, z^*) \in (J_{h_n} \backslash S_\epsilon) \cap N_1(x, y, z)$, then $\widehat{\boldsymbol{\beta}}^*(x^*, y^*, z^*) = (1 + O(h_n^2), O(h_n^2), O(h_n^2))^T$ a.s., which converges to the normal direction of P_1 as $n \to \infty$. Likewise, if (x^*, y^*, z^*) is more than h_n^* away from L and $(x^*, y^*, z^*) \in (J_{h_n} \backslash S_\epsilon) \cap N_2(x, y, z)$, then $\widehat{\boldsymbol{\beta}}^*(x^*, y^*, z^*) = \left(\frac{\kappa + O(h_n^2)}{\sqrt{1+\kappa^2}}, \frac{1+O(h_n^2)}{\sqrt{1+\kappa^2}}, O(h_n^2)\right)^T$ a.s. which converges to the normal direction of P_2 as $n \to \infty$. Define,

$$
\begin{aligned}
m_1 &= \text{number of elements in } \widehat{D}_n \cap N_1(x, y, z) \text{ that are more than } h_n^* \text{ away from } L. \\
m_2 &= \text{number of elements in } \widehat{D}_n \cap N_2(x, y, z) \text{ that are more than } h_n^* \text{ away from } L. \\
m_3 &= \text{number of elements in } \widehat{D}_n \cap O(x, y, z) \text{ that are at most } h_n^* \text{ away from } L.
\end{aligned}
$$

Since m is the number of points in $\widehat{D}_n \cap O(x, y, z)$ and $\frac{h_n^*}{h_n^3} = o(1)$, we have $\frac{m_3}{m} = O\left(\frac{h_n^*}{h_n}\right) = o(h_n^2)$ a.s. Similarly we can show that $\frac{m_1}{m}$ and $\frac{m_2}{m}$ are strictly between 0 and 1 when n is large enough. Then the matrix G has the properties that

$$\begin{aligned}
g_{1,1} &= \frac{1}{m}\left(m_1 + \frac{\kappa^2 m_2}{1+\kappa^2} + m_3\zeta\right) + O(h_n^2) \;\; a.s. \\
g_{1,2} &= g_{2,1} = \frac{1}{m}\left(\frac{\kappa^2 m_2}{1+\kappa^2} + m_3\zeta\right) + O(h_n^2) \;\; a.s. \\
g_{2,2} &= \frac{1}{m}\left(\frac{m_2}{1+\kappa^2} + m_3\zeta\right) + O(h_n^2) \;\; a.s. \\
g_{3,3} &= \frac{m_3\zeta}{m} + O(h_n^2) \;\; a.s. \\
g_{1,3} &= g_{3,1} = \frac{m_3\zeta}{m} + O(h_n^2) \;\; a.s. \\
g_{2,3} &= g_{3,2} = \frac{m_3\zeta}{m} + O(h_n^2) \;\; a.s. \qquad \text{(A.5)}
\end{aligned}$$

where ζ is a number between -1 and 1. Consequently, when n is sufficiently large $|g_{1,1}|$, $|g_{1,2}|$, $|g_{2,2}| > 0$ a.s. and $|g_{1,3}|$, $|g_{2,3}|$ and $|g_{3,3}|$ are all of the order $o(1)$ a.s. If λ is an eigenvalue of G, then we have $\det(G - \lambda I) = 0$. Combining this with (A.5), we have $(g_{3,3} - \lambda)\left((g_{1,1} - \lambda)(g_{2,2} - \lambda) - g_{1,2}g_{2,1}\right) = O(h_n^2)$ a.s. Therefore, when n is large enough, one solution of λ is $O(h_n^2)$ a.s. and the other two solutions differ from 0 by at least a non-zero constant because of the Cauchy-Schwarz inequality that $|g_{1,1}g_{2,2} - g_{1,2}g_{2,1}| > 0$. Now, proceeding similarly as in the proof of *Lemma A.3*, we can check that the eigenvector corresponding to the smallest eigenvalue of G is $(O(h_n^2), O(h_n^2), 1 + O(h_n^2))^T$ a.s., which converges to $(0, 0, 1)^T$, the direction of L.

Now, we can check that

$$\overline{\boldsymbol{\beta}}^* = \left(\frac{m_1}{m} + \frac{m_2\kappa}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2), \frac{m_2}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2), \right.$$
$$\left. O(h_n^2) + o(h_n^2)\right)^T a.s.$$

So, the orthogonal direction of the plane P, defined in the description of the algorithm to approximate the JLS by two crossing half-planes, in Section 3.2.2 is

$$\vec{\boldsymbol{l}} = \Big(-\frac{m_2}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2), \frac{m_1}{m} + \frac{m_2\kappa}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2), \\ O(h_n^2) + o(h_n^2)\Big)^T \; a.s.$$

The inner product of this orthogonal direction with $\widehat{\boldsymbol{\beta}}(x^*, y^*, z^*)$ is

$$\begin{cases} -\frac{m_2}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2) \;\; a.s., \\ \text{if } (x^*, y^*, z^*) \in \widehat{D}_n \cap N_1(x, y, z) \text{ and is more than } h_n^* \text{ away from } L. \\ \frac{m_1}{m\sqrt{1+\kappa^2}} + O(h_n^2) + o(h_n^2) \;\; a.s., \\ \text{if } (x^*, y^*, z^*) \in \widehat{D}_n \cap N_2(x, y, z) \text{ and is more than } h_n^* \text{ away from } L. \end{cases}$$

when n is large enough the first number is negative and the second number is positive almost surely. Define

$$G_1(x, y, z) = \{(x^*, y^*, z^*) \colon (x^*, y^*, z^*) \in \widehat{D}_n \cap O(x, y, z) \text{ and } \vec{\boldsymbol{l}}^T \widehat{\boldsymbol{\beta}}(x^*, y^*, z^*) \leq 0\}.$$
$$G_2(x, y, z) = \{(x^*, y^*, z^*) \colon (x^*, y^*, z^*) \in \widehat{D}_n \cap O(x, y, z) \text{ and } \vec{\boldsymbol{l}}^T \widehat{\boldsymbol{\beta}}(x^*, y^*, z^*) > 0\}.$$

Then, when n is large enough, $G_1(x, y, z)$ includes all points that are more than h_n^* away from L and in $\widehat{D}_n \cap N_1(x, y, z)$ and $G_2(x, y, z)$ includes all points that are more than h_n^* away from L and in $\widehat{D}_n \cap N_2(x, y, z)$. By Theorem 3.3.1 and the fact that $\frac{m_3}{m} = o(h_n^2)$, the center of $G_1(x, y, z)$ would converge almost surely to some point on P_1 and the center of $G_2(x, y, z)$ would converge almost surely to some point on P_2 and both convergence rates would be $O(h_n)$. *Lemma A.4* follows from this result and the results about convergence of $\widehat{\boldsymbol{\beta}}^*$'s in the first paragraph of this proof.

***Lemma A.5**:* Besides the conditions in *Lemma A.4* on f and certain procedure parameters, we further assume that $\widetilde{h}_n = o(1)$, $\frac{h_n}{\widetilde{h}_n} = o(1)$, and $D \cap O(x, y, z)$ is a circular cone, then the local cone fitted by the algorithm (local approximation to the JLS by a cone) described in Section 3.2.2 converges pointwise to $D \cap O(x, y, z)$ almost surely.

Proof of *Lemma A.5*: Without loss of generality, let us assume that the central axis of the true cone is parallel to the x-axis with a vertex at $\boldsymbol{v}$, and the angle between the central axis and any generatrix of the cone is θ. In such cases, the direction of the central axis is $\boldsymbol{\beta}_C = (1, 0, 0)^T$. From *Lemma A.2*, for a given detected edge pixel (x_l^*, y_l^*, z_l^*) in $O(x, y, z)$, if it is more than $\frac{h_n^*}{\sin\theta}$ away from $\boldsymbol{v}$, then the angle between $\widehat{\boldsymbol{\beta}}_l^{\,*}$ and the central axis of the cone is $\theta + O(h_n^*)$ a.s. Therefore, the sample variance, denoted as $\tilde{\sigma}_n^2$, of the inner products of $\{\widehat{\boldsymbol{\beta}}_l^{\,*}, l = 1, 2, \ldots, m\}$ and $\boldsymbol{\beta}_C$ is $O((h_n^*)^2)$ a.s. For a given direction $\tilde{\boldsymbol{\beta}}$, if $h_n^*/(\tilde{\boldsymbol{\beta}} - \boldsymbol{\beta}_C) = o(1)$, then $\tilde{\sigma}_n^2$ would have the property that $(h_n^*)^2/\tilde{\sigma}_n^2 = o(1)$, which is uniformly true for all such $\tilde{\boldsymbol{\beta}}$. So, the direction minimizing $\tilde{\sigma}_n^2$ among all possible directions is $(1, O(h_n^*), O(h_n^*))$ a.s. Therefore, the estimated direction of the central axis of the true cone, as described in item (i) of the algorithm to locally approximate the JLS by a cone, in Section 3.2.2, has the property that $\widehat{\theta} = \theta + O(h_n^*)$ a.s.

From item (iii) of the that algorithm in Section 3.2.2, plane $\widetilde{P}$ divides $\widetilde{O}(x, y, z)$ into two parts. Let us define $\widetilde{O}_1(x, y, z)$ to be the part where the vertex $\boldsymbol{v}$ of the cone lies, and the other part is denoted as $\widetilde{O}_2(x, y, z)$. It is clear that the distance of the center of the detected edge pixels in $\widetilde{O}_1(x, y, z)$ from $\widetilde{P}$ is $O(h_n)$, and the center of the detected edge pixels in $\widetilde{O}_2(x, y, z)$ from $\widetilde{P}$ is of the order $\widetilde{h}_n$. So, the center of $\widehat{D}_n \cap \widetilde{O}_1(x, y, z)$, denoted as (c_x^*, c_y^*, c_z^*) in Section 3.2.2, is within $O(h_n^*)$ from the central axis of the true cone, because by Theorem 3.3.1 all the detected edge pixels are within $O(h_n^*)$ from the true JLSs.

Suppose, the estimated vertex location is $\widehat{\boldsymbol{v}}$. By the fact that the detected edge pixels are within $O(h_n^*)$ from the true JLSs (cf., Theorem 3.3.1), the orthogonal distance between the fitted cone and the detected edge pixels in $O(x, y, z)$ is $O(\|\widehat{\boldsymbol{v}} -$

$\boldsymbol{v}\|) + O(h_n^*)$ a.s. Moreover, by the algorithm (to locally approximate the JLS by a cone) in Section 3.2.2, $\widehat{\boldsymbol{v}}$ is chosen by minimizing the orthogonal distance. By similar arguments to those in the first paragraph of the proof, we have $\|\widehat{\boldsymbol{v}} - \boldsymbol{v}\| = O(h_n^*)$ a.s. By this result and the results obtained in the previous paragraphs, the fitted cone converges pointwise to $D \cap O(x, y, z)$ almost surely.

Proof of Theorem 3.3.2: From the first part of *Lemma A.2*, it is obvious that $\|\widehat{f} - f\|_{\Omega_{\bar{J},h_n}} = O(h_n^2)$, a.s., under the conditions stated in the theorem. Now, let us consider a given point $(x, y, z) \in J_{h_n} \cap S_\epsilon$ in the following three cases.

Case I: The JLS has unique tangent plane at any of its points in $O(x, y, z)$.

Assume that (x_*, y_*, z_*) is the nearest point on the JLSs to (x, y, z). From the proof of *Lemma A.3*, it can be seen that the local plane fitted by the corresponding algorithm in Section 3.2.2 converges almost surely to the tangent plane of the JLSs at (x_*, y_*, z_*). By *Lemma A.4*, the two half-planes fitted by the corresponding algorithm in Section 3.2.2 also converges almost surely to the tangent plane at (x_*, y_*, z_*). So does the fitted cone by the corresponding algorithm in Section 3.2.2, as justified by *Lemma A.5*.

Recall that $O_1(x, y, z)$ and $O_2(x, y, z)$ are the two parts of $O(x, y, z)$ separated by the true JLS, with $O_1(x, y, z)$ containing the point (x, y, z). Similarly, let us define $E_1(x, y, z)$ to be the part of $O(x, y, z)$ separated by the estimated JLS that contains the point (x, y, z). From the first paragraph of the proof of *Lemma A.3*, we know that the number of design points in $E_1(x, y, z) \bigcap O_1(x, y, z)$ and $E_1(x, y, z) \bigcap O_2(x, y, z)$ are of orders $O(n^3h_n^3)$ and $O(n^3h_n^5)$, a.s., respectively. By expression (3.3), we have

$$\begin{aligned}\widehat{f}(x,y,z) =& \frac{\sum\limits_{(x_i,y_j,z_k)\in O_1(x,y,z)\cap E_1(x,y,z)} w^*(x_i,y_j,z_k)f(x_i,y_j,z_k)}{\sum\limits_{(x_i,y_j,z_k)\in O_1(x,y,z)\cap E_1(x,y,z)} w^*(x_i,y_j,z_k)} . \frac{|O_1(x,y,z)\cap E_1(x,y,z)|}{|E_1(x,y,z)|} \\ &+ \frac{\sum\limits_{(x_i,y_j,z_k)\in O_2(x,y,z)\cap E_1(x,y,z)} w^*(x_i,y_j,z_k)f(x_i,y_j,z_k)}{\sum\limits_{(x_i,y_j,z_k)\in O_2(x,y,z)\cap E_1(x,y,z)} w^*(x_i,y_j,z_k)} . \frac{|O_2(x,y,z)\cap E_1(x,y,z)|}{|E_1(x,y,z)|} \\ =& \; (f(x,y,z)+O(h_n)) . \left(1+O(h_n^2)\right) + O(h_n^2) \quad a.s. \qquad \text{(A.6)} \\ =& \; f(x,y,z)+O(h_n) \quad a.s.,\end{aligned}$$

where $w^*(x_i,y_j,z_k)$ denote the weights in the LLK estimator defined in (3.3), and $|A|$ denotes the number of design points in the region A.

Case II: The JLS has two different one-sided tangent planes at some of its point (x_*,y_*,z_*) in $O(x,y,z)$.

From *Lemma A.4*, the fitted two half-planes by the corresponding algorithm in Section 3.2.2 converges almost surely to the two one-sided tangent planes with rate $O(h_n)$. Therefore, $RSS_2(x,y,z)/m = O(h_n^2)$, a.s. However, it is obvious that the fitted plane and cone by the algorithms described in Sections 3.2.2 and 3.2.2 both do not converge to the two one-sided tangent planes; further, $RSS_1(x,y,z)/m$ and $RSS_3(x,y,z)/m$ would converge almost surely to two positive constants. So, when n is large enough, we have $BIC_2(x,y,z) < \min(BIC_1(x,y,z), BIC_3(x,y,z))$, a.s. Consequently, the two fitted half-planes by the corresponding algorithm in Section 3.2.2 will be selected for estimating the JLS in $O(x,y,z)$ by the BIC procedure (3.6). From the proof of *Lemma A.4*, we can see that $|O_1(x,y,z)\cap E_1(x,y,z)|/|E_1(x,y,z)| = 1+O(h_n^2)$ a.s., and $|O_2(x,y,z)\cap E_1(x,y,z)|/|E_1(x,y,z)| = O(h_n^2)$ a.s. Therefore, by similar arguments to (A.7), we have $\widehat{f}(x,y,z) = f(x,y,z)+O(h_n)$ a.s.

Case III: $D \cap O(x,y,z)$ is a circular cone.

By *Lemma A.5* and similar arguments to those in Case II, we can show that the BIC procedure (3.6) would select the fitted cone by the corresponding algorithm in Section 3.2.2 for estimating the JLS in $O(x,y,z)$, and consequently $\widehat{f}(x,y,z) = f(x,y,z) + O(h_n)$ a.s. ■

A.3 Derivation of the threshold T_n in (3.11)

By definition, we have

$$\begin{aligned}
\widehat{a}_1(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in O_1^*(x,y,z)} \xi_{ijk} K_{h_n^*}(x_i,y_j,z_k)}{\sum_{(x_i,y_j,z_k)\in O_1^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k)} \\
\widehat{a}_2(x,y,z) &= \frac{\sum_{(x_i,y_j,z_k)\in O_2^*(x,y,z)} \xi_{ijk} K_{h_n^*}(x_i,y_j,z_k)}{\sum_{(x_i,y_j,z_k)\in O_2^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k)}.
\end{aligned}$$

So,

$$\begin{aligned}
& \mathrm{Var}\left(\widehat{a}_1(x,y,z) - \widehat{a}_2(x,y,z)\right) \\
= & \frac{\sum_{(x_i,y_j,z_k)\in O_1^*(x,y,z)} \sigma^2 K_{h_n^*}^2(x_i,y_j,z_k)}{[\sum_{(x_i,y_j,z_k)\in O_1^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k)]^2} \\
+ & \frac{\sum_{(x_i,y_j,z_k)\in O_2^*(x,y,z)} \sigma^2 K_{h_n^*}^2(x_i,y_j,z_k)}{[\sum_{(x_i,y_j,z_k)\in O_2^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k)]^2}. \qquad (A.7)
\end{aligned}$$

In cases when K is chosen spherically symmetric, we have the results that

$$\begin{aligned}
& \sum_{(x_i,y_j,z_k)\in O_1^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k) \\
= & \sum_{(x_i,y_j,z_k)\in O_2^*(x,y,z)} K_{h_n^*}(x_i,y_j,z_k) \\
= & \frac{1}{2}\sum K_{h_n^*}(x_i,y_j,z_k),
\end{aligned}$$

where $\sum$ is over all voxels. Note that, because the kernel function K is defined in a unit ball, the $\sum$ in the last equation of the above expression is actually over all voxels in $O^*(x, y, z)$ only. After applying these results to the denominators on the right-hand-side of equation (A.7), we have

$$\begin{aligned} & \operatorname{Var}\left(\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)\right) \\ = \; & \frac{4\sigma^2 \sum K^2_{h^*_n}(x_i, y_j, z_k)}{[\sum K_{h^*_n}(x_i, y_j, z_k)]^2}. \end{aligned} \tag{A.8}$$

Therefore, by the Central Limit Theorem (CLT) in statistics, $\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)$ is asymptotically normally distributed with mean 0 and the variance given in (A.8). In practice, σ^2 in (A.8) is often unknown, and should be replaced by its estimator $\widehat{\sigma}^2$. Thus, a reasonable choice for T_n is the one given in (3.11), based on the asymptotic distribution of $\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)$. In such cases, when there are no edge voxels in $O^*(x, y, z)$, it is asymptotically true that

$$P\left(|\widehat{a}_1(x, y, z) - \widehat{a}_2(x, y, z)| > T_n\right) = \alpha_n.$$