Indian Statistical Institute

Doctoral Dissertation

A Regression Tree Framework for Denoising and Monitoring of Image Data

A thesis submitted to the Indian Statistical Institute in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Statistics

Author: **Subhasish Basak**

Supervisor: **Prof. Partha Sarathi Mukherjee**

Interdisciplinary Statistical Research Unit
Statistical Sciences Division

Declaration of Authorship

I, Subhasish Basak, a Senior Research Fellow at Interdisciplinary Statistical Research Unit of Indian Statistical Institute, Kolkata, declare that this dissertation titled "A Regression Tree Framework for Denoising and Monitoring of Image Data" and the works presented in it are my own. I confirm the following statements.

- The work was done while in candidature for a research degree at the Indian Statistical Institute, Kolkata.
- No part of this dissertation has been submitted for a degree or any other qualification at this institute or elsewhere.
- I have clearly cited all the previously published works that I have consulted for writing this dissertation.
- I have properly acknowledged all main sources of data used in this dissertation and the methods on which I have built my work.
- I have properly acknowledged all my co-authors who helped in developing different parts of this dissertation.

Subhasish Basak

Senior Research Fellow in Statistics

Interdisciplinary Statistical Research Unit

Indian Statistical Institute, Kolkata

Certificate from Supervisor

This is to certify that this doctoral dissertation entitled "A Regression Tree Framework for Denoising and Monitoring of Image Data," submitted by Subhasish Basak to the Indian Statistical Institute, is a record of bona fide research work under my guidance and supervision. The work contained in this dissertation is original and has not been submitted elsewhere for any degree.

Prof. Partha Sarathi Mukherjee
Professor
Interdisciplinary Statistical Research Unit
Indian Statistical Institute, Kolkata

List of Publications

1. **Basak, S.** and Mukherjee, P.S. "An Efficient Image Denoising Method Integrating Multi-resolution Local Clustering and Adaptive Smoothing", *Statistical Methods & Applications*, 2025, 34(4), 767–785. [Link to the article]
   Note: This paper is based on Chapter 2 of this dissertation.

2. **Basak, S.**, Roy, A. and Mukherjee, P.S. "Estimation of Piecewise Continuous Regression Function in Finite Dimension using Oblique-axis Regression Tree with Applications in Image Denoising", *Statistica Sinica*, To appear.
   DOI: 10.5705/ss.202025.0120. [Link to the article] [Supplementary]
   Note: This paper is based on Chapter 3 of this dissertation.

3. **Basak, S.**, Roy, A. and Mukherjee, P.S. "A Decision Tree Framework for Monitoring Drift Patterns in Image Data", Pursuing publication in a journal. [ArXiv Link]
   Note: This paper is based on Chapter 4 of this dissertation.

Abstract

The proliferation of advanced image acquisition technologies has led to the routine collection of large-scale image data across numerous scientific domains. This widespread reliance on image data accentuates the imperative to develop robust and efficient imaging techniques, which are essential for supporting modern applications across various scientific and industrial domains. This dissertation focuses on the development and analysis of methods for image denoising and image monitoring, two fundamental tasks in modern image analysis. A wide array of image denoising techniques exists in the literature, each tailored to handle specific types of noise or structural characteristics. However, no single method proves universally optimal, as each comes with its own advantages and trade-offs. In the first part of the dissertation, different configurations of local neighborhoods are investigated, and an adaptive framework is proposed that combines these with local clustering-based smoothing to effectively harness the advantages of both methodologies. The dissertation then introduces a regression tree-based framework utilizing Oblique-axis Regression Trees (ORT) to estimate discontinuous regression functions in finite-dimensional spaces, and applies this methodology to achieve effective image denoising. Due to an alternative set of assumptions on the underlying regression function, the overall structure of the proofs is substantially simpler than those typically found in the existing literature on regression trees. Finally, leveraging the ORT framework, the dissertation introduces an original approach to monitor drift patterns within an image sequence. Even though gradual temporal variations, known as drifts, are frequently observed in image sequences, drift monitoring remains an underexplored research area. This dissertation thus makes an effort to address that gap. Theoretical analysis and numerical studies, conducted on both simulated and real-world data, demonstrate the broad applicability and effectiveness of the proposed methods.

Acknowledgements

I am grateful to the Indian Statistical Institute, Kolkata (ISI), for awarding me the research fellowship and for providing an intellectually vibrant and supportive academic environment. The academic atmosphere at ISI has played a pivotal role in shaping my research journey. I am particularly thankful for the state-of-the-art computing facilities and access to a wide range of academic journals, which were instrumental for carrying out my research work efficiently.

I owe my deepest gratitude to my advisor, Prof. Partha Sarathi Mukherjee, for his unwavering support, constant encouragement, and profound insights throughout this journey. His boundless enthusiasm for research and ability to inspire critical thinking have left a lasting impact on my academic growth. The development of this dissertation has been deeply influenced by his mentorship.

I would also like to extend my sincere thanks to Prof. Subir Kumar Bhandari and Prof. Sourabh Bhattacharya for their continued support and valuable suggestions at various stages of my research. Their thoughtful feedback, motivating discussions, and generous guidance have significantly contributed to the quality and direction of my work and have helped nurture my interest in research.

I am also grateful to my collaborator, Anik Roy, for his invaluable contributions to our joint research endeavors. His clarity of thought, technical expertise, and collaborative spirit have enriched both the research process and the outcomes we have achieved together.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the modern world, image processing holds significant importance and is widely utilized across various scientific disciplines. With digital images being integral to many areas of life, from healthcare to space research, there is a growing need for tools that can transform this visual data into clearer and more understandable forms. Image processing offers these capabilities, serving as a multidisciplinary field focused on the manipulation and analysis of digital images. In manufacturing industries, applications include stress and strain analysis of products, anomaly detection in the rolling process, steel and tile surface monitoring, etc. In medical science, various imaging modalities such as X-ray, CT-scan, MRI, and fMRI are being widely used for medical diagnosis. For surveillance of the Earth's surface, satellite images are commonly used. It becomes the basic tool for studying agriculture, forest science, ecology, ecosystems, and many more. Note that in all of these applications, image comparisons and sequential image monitoring are important. These images often contain noise, anomalies, etc., that make eventual image analysis unreliable. Therefore, image analysis as a modern-day research area is gaining a lot of importance.

At its simplest form, an image can be a binary image, which assigns values zero and one to each point in the image, which represent the dark and brighter parts of the image. However, we consider grayscale images only in this dissertation, which are essentially bivariate functions $f(x, y)$ of co-ordinates (x, y), where the functional value represents the intensity or brightness of the image at that given point. This function f, is referred to

as the image intensity function. Although in reality, coordinates are continuous, working with continuous functions is impractical in computational contexts. Therefore, images are typically discretized or digitized, allowing them to be efficiently processed by computers and captured by digital imaging devices. In these digital images, both the coordinates and the intensity value of the images are digitized. Co-ordinates usually represented in the form (i, j) where $i \in 1, 2, \ldots, m,$ and $j \in 1, 2, \ldots, n$ and $m \times n$ is called the resolution of the image. The intensity values are usually represented as numbers between 0 and 255, with 0 being the darkest and 255 being the brightest. These values can also be represented in the interval $[0, 1]$ by applying appropriate scaling. From these representations, we can see that an image can also be represented as an $m \times n$ matrix shown below, where each entry in the matrix is the intensity value of the image at that corresponding pixel.

$$\text{Image intensity matrix} = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \cdots & f(0,n) \\ f(1,0) & f(1,1) & f(1,2) & \cdots & f(1,n) \\ f(2,0) & f(2,1) & f(2,2) & \cdots & f(2,n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f(m,0) & f(m,1) & f(m,2) & \cdots & f(m,n) \end{bmatrix}.$$

In the case of color images, we use three matrices representing the colors red, green, and blue. As we consider grayscale images only, we convert a color image to its grayscale version using the standard formula:

$$f(x, y) = 0.2989 * f_r(x, y) + 0.5870 * f_g(x, y) + 0.1140 * f_b(x, y).$$

For a more detailed description of various topics of image processing, refer to Gonzalez and Woods (2019) and Qiu (2005).

1.1 An Overview of the Common Tasks in Image Processing

In this section, we briefly discuss various commonly used image processing tasks and their challenges.

1.1.1 Edge detection

Edge detection is a fundamental task in image processing and computer vision, addressing the problem of identifying points in an image where there are significant changes in intensity or texture. These points often correspond to object boundaries, surface discontinuities, or other important structural features. Edge detection is essential in applications such as image segmentation, object recognition, tracking, and scene reconstruction. Additional details about how edge detection is used in various image analysis tasks can be found in Gonzalez and Woods (2019); Marr and Hildreth (1980).

Edges are typically characterized by sharp changes in pixel intensity. Classical edge detection methods rely on differential operators to estimate these changes using gradients. Early techniques, such as the Sobel, Prewitt, and Roberts operators, apply convolution techniques to approximate the first derivative of the image. More details about these methods can be found in Gonzalez and Woods (2019). Among these, the Canny edge detector Canny (1986) is particularly influential due to its multi-stage process that includes Gaussian smoothing, gradient magnitude computation, non-maximum suppression, and hysteresis thresholding. It is widely regarded as a benchmark for edge detection performance.

Figure 1.1: Edge detection using Canny's edge detection technique on the Astronaut image.

More recent approaches incorporate multiscale analysis Lindeberg (1998), anisotropic diffusion Perona and Malik (1990), and machine learning techniques to achieve robust edge detection, especially in noisy or textured images. In complex imaging domains, such as medical imaging or remote sensing, edge detection must be particularly sensitive to fine structural details, while remaining resilient to noise and imaging artifacts.

In addition to the methods discussed above, several alternative nonparametric frameworks have been proposed for estimating unknown functions. Gaussian process (GP) models provide a flexible probabilistic approach, while basis function expansions such as splines and wavelets represent an unknown function through a linear combination of predefined basis elements. These approaches offer complementary trade-offs in terms of flexibility, interpretability, and computational complexity. Such frameworks have also been extended to boundary detection problems. In particular, GP-based approaches model the underlying surface and exploit the fact that derivatives of a Gaussian process are themselves Gaussian processes, enabling boundary identification through changes in the gradient structure (Halder et al., 2024). While these methods are theoretically appealing and highly flexible, they can be computationally demanding for large-scale problems. Additionally, a related line of work in spatial statistics considers boundary detection in areal data through Bayesian areal wombling. In this setting, spatial dependence is modeled via random effects, and boundaries are identified through local discontinuities in these effects (Fitzpatrick et al., 2010). Extensions using Bayesian nonparametric formulations further improve flexibility and uncertainty quantification (Guhaniyogi, 2017). These approaches provide a principled statistical framework but are often computationally intensive. In contrast, the focus of this dissertation is not on developing new boundary detection methodologies. Instead, we employ lightweight edge detection techniques as pre-processing tools, motivated by their computational efficiency and the availability of theoretical guarantees such as convergence in Hausdorff distance under suitable conditions.

Despite decades of development, edge detection remains an active area of research due to the wide variety of image types and applications, and the inherent trade-off between sensitivity, precision, and computational efficiency.

1.1.2 Image denoising

Along with the increasing popularity of digital imaging systems, there is a growing demand for near-perfect images. Image denoising is essential in many real-life problems. For example, denoising in MRI, CT scan, and X-ray images leads to clearer images and more accurate diagnoses. In photography, noise removal enhances the clarity of photos captured in dark conditions or with high ISO settings. In astronomy, efficient noise removal aids in the identification of celestial bodies and phenomena. Satellite imaging benefits from denoising by refining images captured from space, allowing better analysis and interpretation of geographic and environmental data. Figure 1.2 shows the benefits of efficient noise removal from images.

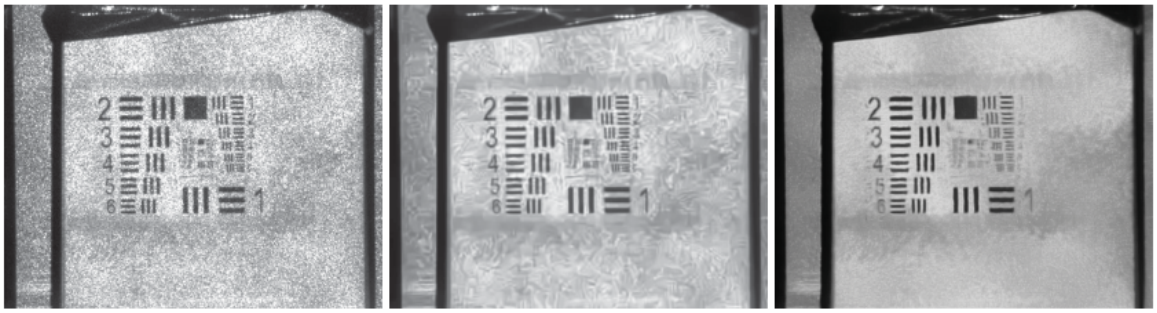


Figure 1.2: Example image restoration of real world images by Dong et al. (2013).

An overview of the long history of image denoising can be found in Gonzalez and Woods (2019); Qiu (2005).

1.1.3 Image registration

Proper alignment of images is a critical pre-processing step in numerous image processing applications, including image comparison and temporal monitoring. In many practical scenarios, image data are not inherently aligned within a common spatial frame, making direct comparison challenging. For instance, in satellite imagery, variations in the satellite's position and viewing angle can cause misalignment across images of the same geographic region. In such cases, image registration becomes essential to map the images

onto a shared coordinate system, thereby enabling accurate analysis and interpretation (Feng and Qiu, 2018; Qiu, 2018; Das et al., 2024; Das and Mukherjee, 2024). In Figure 1.3, Das et al. (2024) show three images where we can use image registration to identify similar parts in the images.

Figure 1.3: Misalignment of the objects across images of the same scene.

The problem of image registration can be formulated mathematically as follows:

$$w_{ref}(\tilde{z}) = f_{ref}(\tilde{z}) + \epsilon_{ref}(\tilde{z}),$$
$$w_{obs}(\tilde{z}) = f_{obs}(\tilde{z}) + \epsilon_{obs}(\tilde{z}), \quad \tilde{z} \in [0,1] \times [0,1].$$

In the above equation, $\tilde{z}$ represents the pixel co-ordinates,$w_{ref}(\tilde{z})$ and $w_{obs}(\tilde{z})$ are the reference and observed image intensity, respectively. $\epsilon_{ref}(\tilde{z})$ and $\epsilon_{obs}(\tilde{z})$ are independent and identically distributed random variables with mean 0 and variance unknown. We also assume that the observed image intensity function is a transformed version of the true image intensity function, i.e :

$$f_{obs}(T(\tilde{z})) = f_{ref}(\tilde{z}), \quad \tilde{z} \in [0,1] \times [0,1].$$

In this context, T denotes a transformation applied to the coordinate system. This transformation can represent various real-world phenomena such as zooming, rotation,

and translation. Transformations that involve only rotation and translation are referred to as rigid body transformations, whereas those that include operations such as scaling (zooming) and shearing are classified as non-rigid transformations. In satellite imagery, the transformations encountered are predominantly rigid-body, and can be mathematically expressed as:

$$T(\tilde{z}) = R\tilde{z} + \tilde{v},$$

where R is a rotation matrix and $\tilde{v}$ is the vector corresponding to the translation. See Qiu and Xing (2013b); Xing and Qiu (2011) for a more detailed discussion about image registration.

1.1.4 Image deblurring

Another significant challenge in image processing is the presence of blurred images. Due to the nature of photographic capture, some degree of blurring is inevitable, as image sensors cannot be exposed to light instantaneously because of mechanical limitations inherent in the capturing devices. In addition to noise, blurring can also result from motion, either of the subject being photographed or the camera itself during exposure. To address this, image deblurring methods try to retrieve the true image from the blurred surface, which can often be expressed as a convolution of the true intensity function and a blurring function.

$$w(x, y) = H\{f\}(x, y) + \varepsilon(x, y),$$

In the above equation, f is the true intensity function, H is the blurring function or the *Point Spread Function* (psf), and ε is the point-wise noise. The convolution $H\{f\}$ is given by:

$$H\{f\}(x, y) = \int_{\mathbb{R}^2} h(u, v) f(x - u, y - v)\, du dv.$$

Note that the above equation can have multiple solutions in H and F for a known $H\{f\}$. So this problem is not well-defined if no information about H or f is known. So early literature in this topic, e,g., Figueiredo and Nowak (2003), assumed that h was known. In practice, this assumption does not hold true, as in real images, there are

multiple factors contributing to the blurring of an image. The methods that assume the H is not completely known are referred to as blind image deblurring methods. For more details on blind image deblurring, refer to Qiu and Kang (2015); Kang et al. (2018); Kang (2020); Kang et al. (2026).

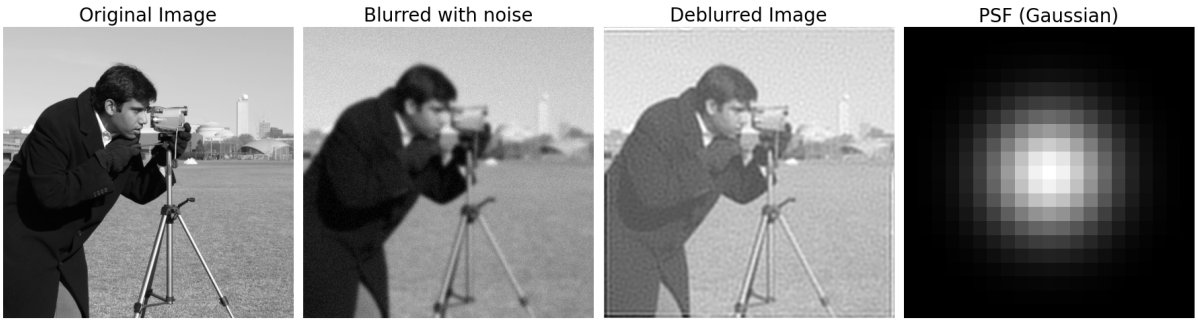


Figure 1.4: Deblurring using Wiener deconvolution on the cameraman image.

1.1.5 Image comparison and monitoring

Another problem in image processing is monitoring a sequence of images, typically in the form of video frames. In this application, images are continuously observed and analyzed, often in real-time, to draw meaningful conclusions without human intervention. This kind of automated interpretation is essential in scenarios where human monitoring is impractical due to scale, speed, or cost.

For example, in security surveillance, video footage from security cameras can be automatically processed to detect potentially harmful or suspicious events such as unauthorized access, unattended baggage, physical altercations, or trespassing. Instead of relying solely on human operators, the use of these tools significantly improves the efficiency, accuracy, and scalability of such operations.

Beyond security, image monitoring is also employed in industrial quality control to detect anomalies, defects, equipment failures, or irregular patterns in production lines. It is widely used in applications such as vehicle detection, motion detection, change-point analysis, and facial recognition, making it a critical component in many modern computer vision systems. In the field of medical imaging, technologies like X-rays, CT scans, MRI,

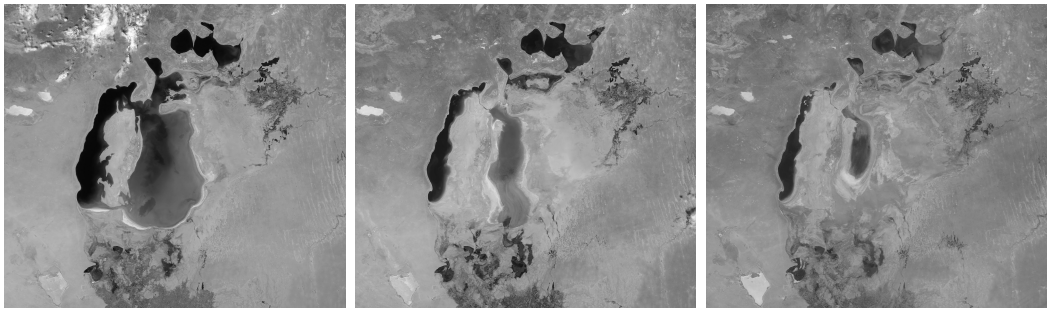


Figure 1.5: Example of image comparison and monitoring in satellite imaging: a sequence of satellite images of the Aral Sea area taken on 25.08.2000 (left), 26.08.2010 (middle), and 21.08.2018 (right).

and fMRI have been widely used for medical diagnosis. Monitoring such images to detect any significant change plays an important role in various medical applications. Nowadays, satellite images have become a basic tool in the surveillance of the earth's surface. It has been widely used in the research of agriculture, forest science, ecology and ecosystem, coastal resources, environment, etc.

Mathematically, we can express a sequence of 2D $n \times n$ grayscale images as:

$$\mathcal{I}_k(i,j) = w(x_i, y_j, t_k) = f(x_i, y_j, t_k) + \varepsilon_{ijk} \text{ for } i,j \in \{1,2,\ldots,n\} \text{ and } k \in \{1,2,\ldots\},$$

where $\mathcal{I}_k$ denotes the observed image at the k-th time point, with w_{ijk} representing the intensity of the observed image at the pixel location (i,j) at the k-th time point t_k. The true but unknown image intensity at the same pixel is denoted by $f(x_i, y_j, t_k)$. The term ε_{ijk} represents independent and identically distributed (i.i.d.) random noise at each pixel, with a mean of 0 and variance σ^2. In this setup, the pixel coordinates are arranged on a regular grid within the design space $\Omega = [0,1] \times [0,1]$, specifically given by $(x_i, y_j) = (i/n, j/n) : i,j = 1,2,\ldots,n$. Additionally, the observed time points are assumed to be equally spaced. In the problem of image monitoring, we are interested in estimating the function f. This topic is discussed in detail in Chapter 4.

The most fundamental step in image-based process monitoring is image comparison, where the primary objective is to statistically compare two images. This task is indeed challenging due to the complex structure of image surfaces, specifically, their spatial nature and the presence of discontinuities, as well as pointwise contamination from noise. In the nonparametric regression literature, a related problem has been approached through

regression curve comparison (Hall and Hart, 1990; King et al., 1991; Kulasekera, 1995). For a comprehensive review on comparing regression curves, readers are referred to Dette and Neumeyer (2001). The central ideas of nonparametric comparison of image surfaces are based on nonparametric ANCOVA with multiple covariates. Bowman (2006) suggests a generalization of an ANOVA-type test for comparing regression surfaces. A more flexible test based on the $\mathcal{L}^2$-distance between the regression surface estimate is proposed by Wang and Ye (2010). These existing methods are the generalizations of the method proposed by Dette and Neumeyer (2001). However, these traditional approaches often overlook the complicated structure of image data, such as edges and other singularities. More recently, due to widespread application of image monitoring, the field has seen increased interest in jump-preserving image comparison methods (Roy and Mukherjee, 2024b; Qiu, 2020; Feng and Qiu, 2018). These methods aim to compare images while preserving the important features like discontinuities and singularities. Recently, Roy and Mukherjee (2026) propose a jump regression-based image comparison method in the presence of rigid-body transformation of the image object. An example of image comparison and monitoring is demonstrated in Figure 1.5.

1.2 Underlying Approaches Related to this Dissertation

This thesis primarily focuses on image denoising and image monitoring. Accordingly, the following sub-sections present a review of existing techniques and methods related to these areas, as documented in the current literature.

1.2.1 Jump regression analysis

In this dissertation, we study the images under the jump regression framework. In jump regression, we assume that the observed image intensities $f_{obs}(x_i, y_j)$, for a two-dimensional grayscale image, follow the following model :

$$f_{obs}(x_i, y_j) = f_{true}(x_i, y_j) + \epsilon_{ij}, \text{ where } i \in \{1, 2, \ldots, m\} \text{ and } j \in \{1, 2, \ldots, n\}$$

Here f_{true} is the true intensity function, x_i, y_j are the pixel co-ordinates, $x_i = i/m$ and $y_j = j/n$. ϵ_{ij} are i.i.d. random variables with mean 0 and variance σ^2. For simplicity, we consider square images only, where $m = n$.

It is also assumed that the true image intensity function contains discontinuities. These discontinuity points are commonly referred to in the literature as edge pixels or jump points. In the observed image, such jump points may arise due to two primary factors: either the underlying true intensity function exhibits an actual discontinuity at that location, or the discontinuity is an artifact induced by random noise. In the former case, these points typically correspond to meaningful structural features of the image, such as object boundaries. In contrast, jump points arising from the latter scenario generally lack interpretative significance and are attributable to random noise. So, in image processing methods, it is important to preserve these features of images while trying to eliminate the other artifacts caused by randomness. This framework is applicable to both denoising and monitoring problems, as well as various other image processing tasks.

1.2.2 Non-parametric regression and neighborhood selection

Denoising can be effectively performed using nonparametric regression applied to the image pixels. By selecting a neighborhood of appropriate size and shape around a given pixel, it is reasonable to assume that the pixels within this neighborhood exhibit similar characteristics to the target pixel. Consequently, computing a locally weighted average of the pixel intensities within this neighborhood yields a reliable estimate of the true image intensity at the central pixel.

It is important to note that the effectiveness of the estimate is highly dependent on the choice of the neighborhood and the weighting scheme employed. The selection of these parameters significantly influences the estimator's ability to preserve important image features while reducing noise. If the selected neighborhood is too small, it may fail to incorporate sufficient information, thereby limiting the potential for improving the estimate. Conversely, choosing a neighborhood that is too large increases the risk of including edge pixels or dissimilar regions, which can introduce bias and adversely affect the accuracy of the estimate. We elaborately discuss this issue in Chapter 2.

1.2.3 Local pixel clustering

Another intuitive approach to image denoising involves partitioning the image data into groups of similar pixels and constructing estimates based on these groupings. This partitioning can be performed either locally, within confined neighborhoods, or globally, across the entire image domain. In the case of local partitioning, it is generally reasonable to assume that only a limited number of distinct groups exist within a small neighborhood. However, this assumption does not hold in the context of global partitioning, where the increased diversity of image content typically results in a larger number of heterogeneous regions. Moreover, the number of such regions is typically unknown, adding an additional complexity to the partitioning process.

In the clustering-based approach, neighborhoods can be partitioned into a small number of groups, and estimates can then be computed based on these partitions. The method proposed by Mukherjee and Qiu (2015) adopt this framework, leveraging local clustering to enhance denoising performance.

1.2.4 Partitioning and decision trees

For the partitioning approach, we can use the method Classification and Regression Trees (CART) to partition the data into smaller parts. Tree-based methods hold a pivotal place in modern statistics and machine learning. Classification and Regression Trees (CART) (Breiman et al., 1984) are recognized as very powerful and widely used tools among supervised learning algorithms. Over time, ensemble learning techniques that combine multiple trees, such as Random Forest (RF) (Breiman, 2001), bagging, and boosting, have become increasingly popular for classification and regression applications. Decision trees are the building blocks of all these algorithms. The popularity of the techniques based on decision trees is attributed to their intuitive construction and appealing interpretability. However, it has been noticed from the time when CART was first proposed that marginal variables as splitting rules can pose challenges in preserving complex structures of the data while smoothing. This issue can be well addressed by taking a linear combination of the predictors as a splitting rule instead of relying on marginal variables. This procedure is

popularly known as the oblique regression tree (ORT) algorithm. In literature, various versions of the ORT algorithms (Cattaneo et al., 2024) are available. However, nearly all of the existing methods assume that the true functional relationship is continuous in nature. However, in practice, there are many situations where the true regression function is clearly discontinuous. Such situations can be recognized when the response variable changes abruptly with small variations in predictor variables. For example, in materials science, rapid phase changes are common during phase transitions. In geostatistics, sensing data from rock strata often exhibits abrupt changes near sediment layers. Similarly, in image analysis, the intensity function of an image changes abruptly around object boundaries or edges.

1.3 A Survey of Various Image Processing Methods

The following section provides a concise review of the literature pertinent to this dissertation.

1.3.1 Image denoising methods in literature

One major type of image denoising algorithm does not detect edges explicitly, but uses the edge information from the image intensities. Bilateral filtering methods, e.g., Chu et al. (1998) use the edge information to assign weights in the local intensity averaging procedure. An anisotropic diffusion method proposed by Perona and Malik (1990) use the edge information to control the direction and the amount of local smoothing. The major drawback of these approaches is that the edges become blurred while smoothing, especially around complicated edge structures such as corners. This is because all pixels in a neighborhood get non-zero weights in the process of local smoothing. Markov random field (MRF) based denoising methods (Geman and Geman, 1984; Godtliebsen and Sebastiani, 1994) introduce a line process to use the edge information, whereas methods such as Rudin et al. (1992); Keeling (2003); Wang and Zhou (2006), etc., introduce various penalty terms in appropriate cost minimization problems. These approaches blur the

fine image details to some extent. Other popular types of image denoising methods include wavelet transformation methods (Portilla et al., 2003), non-local means algorithms (Buades et al., 2005, 2010; Liu et al., 2008), pointwise shape adaptive methods (Foi et al., 2007), channel or orientation space methods (Felsberg et al., 2006; Franken and Duits, 2009), to name a few. One major approach to solve the problem of image denoising is through jump regression analysis (Qiu, 2005). The majority of the initial such attempts (Qiu, 1997, 1998) detect the edges first, and then use this edge information to denoise the image. Later, Mukherjee and Qiu (2011) extend similar ideas to denoise 3-D images. Jump regression-based image denoising techniques that do not explicitly detect edges include methods by Qiu (2004); Gijbels et al. (2006). Another major type of methods explicitly detects the edges first, and then uses the information to denoise (Qiu, 1998; Mukherjee and Qiu, 2011). See Qiu (2007) for a detailed description. The explicit edge detection-based methods do not work well on various types of images, such as low-resolution images, textured images, etc. To resolve this issue, Mukherjee and Qiu (2015) propose local pixel clustering-based image denoising where two different regions within a local neighborhood are detected by pixel clustering based on image intensity values. An adaptive smoothing method using various shapes and sizes of the neighborhood for local smoothing depending on local edge information is another approach (e.g., Takeda et al. (2007)). Even though the neighborhoods are elongated along the edges, they often contain pixels on both sides of the edges, resulting in image blurring. In spite of having all these methods, no one method outperforms the others on a large class of images. It is to be noted that each of these methods has its specific weaknesses alongside its strengths.

Several types of modern image analyses, such as image comparisons (Roy and Mukherjee, 2024b), image monitoring (Roy and Mukherjee, 2024a) require feature-preserving image denoising as a preliminary step, or several such state-of-the-art techniques use the central ideas of various image smoothing methods as building blocks that suit their own specific requirements. The performance of image comparison and monitoring by such methods depends heavily on the effectiveness of image smoothing. Therefore, it is an urgent requirement that integration of various types of state-of-the-art image denoising methods is done to combine the strengths and remove the weaknesses of the concerned methods as much as possible. Another important aspect to note in this context is that deep learning-based denoising methods (e.g., Zamir et al. (2022)) require a large set of

images for training or learning. Therefore, such a type of denoising methods cannot be easily integrated with image monitoring procedures, which itself is a computationally heavy problem. Therefore, the requirement of developing integrated denoising methods such as the one we propose still exists very much.

1.3.2 Tree-based smoothing methods in literature

So far in the literature, there has been a very limited discussion on the estimation of discontinuous regression function. Proposed by Breiman et al. (1984), CART is one of the most popular nonparametric regression methods that fits the piecewise constant function quite well. However, it fails to capture the complex shape, even if it forms a tree with high depth. A similar nonparametric regression approach in the literature is dyadic regression tree (DRT), proposed by Donoho (1997). The central idea is to split the input space at the midpoint along a dimension. The major difference between traditional CART and the DRT is that it allows a recursive splitting at any point of the input space. Although it has computational advantages, and it fits piecewise constant functions reasonably well, it faces similar issues when we are interested in accommodating complex shape boundaries (Chaudhuri and Chatterjee, 2023). Recently, ORT-based piecewise continuous function estimation and its theoretical properties have become an active research area (Cattaneo et al., 2024). ORT-based decision tree is a natural extension of CART and DRT to accommodate the boundaries of the complex shape. Nowadays, ORT-based decision tree has got remarkable attention among the research communities. Zhan et al. (2026) show the consistency of the ORT-based algorithm under the assumption that the underlying regression function is continuous. Additionally, several studies in the Gaussian Process (GP) literature are also useful for estimating piecewise continuous regression functions. These methods involve partitioning the input domain into multiple regions and modeling the data in each region with a separate GP model. A Bayesian treed GP, proposed by Gramacy and Lee (2008), uses a dyadic treed partitioning to split the input domain along axis-aligned directions, fitting a stationary GP model to the data in each tree leaf. Similarly, Konomi et al. (2014) use a Bayesian CART tree to partition the input domain. However, one major limitation of these piecewise GP models is that their partitioning schemes, such as axis-aligned partitions or triangulations, are too restrictive to capture

the complex, curvy boundaries often present in various real-world images.

Random forests, also, have been widely used for image restoration tasks by formulating denoising as a regression problem from noisy observations to clean signal estimates. For instance, Criminisi et al. (2013) demonstrate that regression forests can efficiently model complex conditional distributions for structured prediction tasks, providing a flexible framework that naturally extends to image denoising. Kontschieder et al. (2015) explore hybrid approaches combining decision forests with deep representations, further enhancing predictive performance while retaining the efficiency of tree-based methods. Collectively, these research works in computer vision highlight the suitability of random forests for denoising tasks, particularly when fast inference and the ability to model complex, non-linear relationships are required. In contrast, this dissertation focuses more on statistical rigour and its novelty rather than algorithmic novelty.

In Chapter 3, we explore the use of Oblique Regression Trees (ORT) within the framework of jump regression analysis. We assume that the underlying image intensity function f is piecewise continuous and can be represented as:

$$f(\boldsymbol{z}) = \sum_{l=1}^{P} g_l(\boldsymbol{z}) \mathbb{I}_{\Gamma_l}(\boldsymbol{z}) \quad \boldsymbol{z} \in \Omega = [0,1]^p$$

where each g_l are continuous functions defined on the subdomain Γ_l, and $\{\Gamma_l\}_{l=1}^{P}$ are partitions are Ω. Our objective is to construct an accurate estimate of f by leveraging the flexibility of ORT to learn these partitions and the associated continuous components g_l.

1.3.3 Deep learning-based image denoising methods

In recent years, deep learning-based approaches have significantly advanced the state-of-the-art in image denoising by learning complex, non-linear mappings directly from data. Unlike classical methods that rely on explicit modeling of edge structures or local smoothness assumptions, these methods implicitly capture both local and global features through hierarchical representations.

Convolutional Neural Networks (CNNs) are among the earliest and most widely used

deep learning models for image denoising. A seminal work by Zhang et al. (2017) develops DnCNN, which employs residual learning to predict noise rather than the clean image, leading to improved performance and faster convergence. Similarly, Mao et al. (2016) introduce RED-Net, a deep convolutional encoder-decoder architecture with skip connections that helps preserve fine image details while removing noise. These CNN-based approaches are particularly effective in capturing local spatial structures due to the inductive bias of convolutional filters.

Beyond standard CNNs, fully connected Deep Neural Networks (DNNs) have also been explored for denoising tasks. Early work by Burger et al. (2012) demonstrates that multi-layer perceptrons (MLPs) trained on large datasets can achieve competitive denoising performance. However, due to their lack of spatial inductive bias and high parameter complexity, DNNs are generally less efficient compared to CNN-based architectures for high-dimensional image data.

More recently, Transformer-based architectures have emerged as powerful tools for image restoration tasks, including denoising. These models leverage self-attention mechanisms to capture long-range dependencies and global contextual information. For instance, Chen et al. (2021) propose Image Processing Transformer (IPT), which applies a Vision Transformer framework to various low-level vision tasks. Similarly, Zamir et al. (2022) introduce Restormer, a Transformer architecture specifically designed for high-resolution image restoration, achieving state-of-the-art performance by efficiently modeling both local and global interactions.

Despite their strong empirical performances, deep learning-based denoising methods typically require large amounts of training data and computational resources. Moreover, their black-box nature makes theoretical analysis challenging compared to classical statistical methods. In contrast, tree-based methods such as ORT offer greater interpretability and can be more naturally integrated into frameworks like jump regression analysis, particularly when the goal is to explicitly model discontinuities and boundary structures.

1.3.4 Brief literature survey of image monitoring

Traditional techniques of *statistical process control* (SPC) have been widely used in manufacturing industries to monitor *univariate processes* over time. The main goal of SPC is to observe the process consistently to detect any systematic or non-random variations in the observed data, which is very important for maintaining the quality standards of a product. Recently, there has been a growing interest among researchers in developing control charts specifically designed for monitoring image data. Not only in manufacturing industries, but it also has diverse applications across various disciplines of science and engineering. For a related discussion on image monitoring within the framework of SPC methods, see Qiu (2018, 2020). The problem of image surveillance is a challenging problem for the following reasons. First of all, the task of image surveillance is a high-volume, high-velocity data-rich application. Secondly, the real images often contain noise, and along with noise, the image data itself is complicated in nature. The image intensity surface is not continuous in nature. Therefore, conventional smoothing techniques often fail to accommodate the discontinuous intensity function. Additionally, a typical grayscale image contains a large number of pixels, leading to a high-dimensional problem. Therefore, conventional multivariate SPC methods are not appropriate for such applications. A comprehensive review of statistical image monitoring can be found in Megahed et al. (2011). Prior research in this field mainly follows the conventional control chart after extracting various features from the images. For instance, Megahed et al. (2012) and He et al. (2016) consider a set of regions of interest (ROIs) for individual images and use existing GLR (generalized likelihood ratio) and MGLR (multivariate-GLR) control charts, respectively, based on average image intensity at each ROI. For high-resolution images, the variance-covariance matrix in the MGLR control chart is not invertible, and Okhrin et al. (2020) address this issue by proposing a more sophisticated GLR control chart. The methods proposed by Koosha et al. (2017) and Kang (2022) follow the profile monitoring approach for images, relying on feature-based image monitoring schemes utilizing wavelet coefficients of individual images over time. Roy and Mukherjee (2025) and Yan et al. (2017) discuss anomaly detection in a noisy image sequence with applications in manufacturing industries. Bui and Apley (2021, 2018) propose several approaches for monitoring patterns in the images of textured surfaces. In recent years, there has been a

growing body of research focused on monitoring high-dimensional tensor data. Examples include methods proposed by Zhen et al. (2023); Yang et al. (2023); Fang et al. (2019). Additionally, closely related research includes image comparison and image monitoring schemes based on the jump location curves (JLCs) in the image intensity functions, e.g., Roy and Mukherjee (2024a,b); Feng and Qiu (2018).

1.4 Contribution and Novelty of the Dissertation

In this dissertation, we propose three distinct methods for addressing various problems in image denoising and monitoring.

The novelty of the first method lies in its integration of multiple existing techniques. As discussed earlier, the current literature offers a wide array of denoising algorithms, each tailored to specific aspects of the problem. Many of these methods perform well in particular niches, emphasizing certain features over others. By combining the strengths of these diverse approaches, we develop a more general algorithm capable of handling a broader range of scenarios. The theoretical analysis supporting this integration provides further insight into its effectiveness. Moreover, the proposed merging framework is flexible: it requires only an edge detection algorithm that converges to the truth in the Hausdorff distance, along with two complementary denoising methods, one suited for smooth regions and the other for areas near edges. The specific methods used in this dissertation are not critically important; with minor modifications to the proof, other compatible algorithms could also be employed within this framework.

The second contribution of this dissertation is more theoretical in nature and focuses on the problem of clustering in image data. While decision tree-based clustering methods are well-established in the literature and perform effectively on a variety of general-purpose datasets, they often rest on assumptions that may not hold when applied to image data. Digital images, however, tend to exhibit unique structural properties such as spatial continuity, local smoothness, and grid-based organization. These features not only invalidate some of the assumptions made by general clustering methods but also allow us to impose stronger, more realistic assumptions that are specific to images. By leveraging these properties, we are able to reformulate the clustering problem under a more image-appropriate

framework. As a result, we are able to provide theoretical assurances, such as consistency, that are both more meaningful in the context of image data and simpler to prove due to the regularity of the domain. This theoretical framework thus bridges the gap between generic clustering methods and the specialized demands of image processing tasks.

As the third major contribution, we incorporate the oblique regression tree-based algorithm discussed earlier into the problem of image monitoring. While the existing literature offers many methods for detecting and analyzing sudden changes in image sequences, most of these approaches do not adequately address the issue of gradual changes, called drifts, and the detection of pattern changes in these drifts. This dissertation proposes a regression tree-based framework to monitor the evolving patterns in an image sequence and to detect not only significant changes in the images themselves but also shifts in the patterns of these gradual changes. Such a use of a regression tree framework in the context of image monitoring is novel.

1.5 Organization of this Dissertation

The remainder of this dissertation is organized into four major chapters.

In Chapter 2, we propose a novel hybrid approach that combines two distinct denoising algorithms. The key advantage of this method is its ability to select the most appropriate algorithm based on the underlying edge structure of the image. This dynamic selection improves the performance of image denoising by tailoring the algorithm choice to the specific characteristics of the image. A notable aspect of the hybrid approach is that it merges two algorithms with differing computational costs. For example, a more computationally expensive algorithm, such as a clustering method, is applied only to the regions of the image where it is most effective, such as areas with complex structures or edges. For the rest of the image, a simpler, faster algorithm is used, thus saving computation time. This strategy allows the method to optimize both denoising quality and computational efficiency by applying the right algorithm to the right areas.

Chapter 3 introduces a novel technique that employs an ORT-based (Oblique Regression Trees) approach for global clustering within a given image. This method aims to

partition the image into regions that exhibit similar characteristics. By grouping these similar regions, the algorithm effectively identifies clusters that share common properties, which can be crucial for accurate image analysis. Once the image is segmented into these clusters, the method then uses this grouping information to generate a denoised estimate of the image. By focusing on clusters of similar regions, the denoising process can be more targeted and efficient, improving the overall quality of the image while preserving essential details.

Chapter 4 applies the method from Chapter 3 to the problem of image monitoring. By doing so, we propose a monitoring technique that ignores regular, gradual changes in an image sequence, but detects when the pattern of change or the image itself undergoes a significant alteration.

Finally, this dissertation concludes with a discussion on potential future extensions, applications, and implications of the methods developed in the preceding chapters.

Chapter 2

An Efficient Image Denoising Method Integrating Multi-resolution Local Clustering and Adaptive Smoothing[†]

2.1 Introduction

The objective of this chapter is to integrate the strengths of local clustering-based methods and those adaptive methods that use neighborhoods of various shapes and sizes. In this way, complicated edge structures and fine image details should be preserved while removing noise from an image. The steering kernel method by Takeda et al. (2007) choose small neighborhood near complicated edge structures but gives positive weights to all pixels in the neighborhood. In the proposed method, we use a bigger neighborhood in such regions as well and preserve the complicated edge structures by employing local clustering in that neighborhood and averaging the intensities only from a meaningful partition of that neighborhood (Mukherjee and Qiu, 2015; Mukherjee, 2017). The proposed method utilizes explicit edge detection to determine neighborhood shapes and sizes specifically for pixels

[†]This chapter is based on the publication Basak and Mukherjee (2025): **Basak, S.** and Mukherjee, P.S. "An Efficient Image Denoising Method Integrating Multi-resolution Local Clustering and Adaptive Smoothing", *Statistical Methods & Applications*, 2025, 34(4), 767–785.

located far from the edges. For pixels close to edges, we employ local clustering methods to achieve enhanced performance, particularly around complicated edge structures. For pixels far away from the edges, we use a large enough elliptical neighborhood to smooth the region as much as possible. Numerical studies show that the practical performance of the proposed method is superior to both these approaches as well as most state-of-the-art denoising methods.

The organization of the rest of this chapter is as follows. Section 2.2 describes the proposed integrated image denoising method. Useful theoretical results are discussed in Section 2.3. Section 2.4 shows the performance of the proposed method in comparison with a number of state-of-the-art image denoising methods on both the simulated and real images. A few concluding remarks in Section 2.5 finish the chapter.

2.2 Proposed Methodology

Assume that the observed image intensities follow the following jump regression model

$$z_{ij} = f(x_i, y_i) + \epsilon_{ij} \quad \text{for} \quad i, j = 1, 2, \ldots, n,$$

where $\{(x_i, y_j) = (i/n, j/n), i, j = 1, 2, \ldots, n\}$ are equally spaced design points or pixels. $\{\epsilon_{ij}\}$ are independent and identically distributed (i.i.d.) random variables with mean 0 and variance σ^2. Further assume that $f(x, y)$ is continuous over $[0, 1] \times [0, 1]$ except on some curves which we call the *jump location curve* (JLC). A point (i, j) is called a singular point if it satisfies at least one of the following conditions:

- There exists a constant $\zeta_0 > 0$ such that for any $0 < \zeta < \zeta_0$, the neighborhood of $(i/n, j/n)$ of diameter ζ is divided into more than two connected regions by the JLCs.

- There does not exist any constant $\rho_0 > 0$ such that there exist two orthogonal lines crossing at $(i/n, j/n)$ and two vertical quadrants formed by these two lines belong to two different regions separated by a JLC in a neighborhood of $(i/n, j/n)$ with diameter ρ_0.

Detailed explanations are provided by Qiu and Yandell (1997).

2.2.1 Detection of edge pixels

The first step of our proposed method is to detect the edge pixels. There are several methods available in the literature. Any reasonable method can be used in this step. We select the following edge detection method under the jump regression framework. Consider the following square neighborhood

$$N(x_i, y_i) = \{(x_{i+s}, y_{j+t}), s, t = -k, -k+1, \ldots, 0, \ldots, k-1, k\},$$

and we fit a least square plane in this neighborhood around (x_i, y_j).

$$\widehat{z}_{ij}(x, y) = \widehat{\beta}_0^{(i,j)} + \widehat{\beta}_1^{(i,j)}(x - x_i) + \widehat{\beta}_2^{(i,j)}(y - y_j), \quad (x, y) \in N(x_i, y_j).$$

We can estimate the gradients of the intensity function using $\left(\widehat{\beta}_1, \widehat{\beta}_2\right)^T$. Considering B_1 and B_2 to be two pixels nearest to (i, j) along the gradient, whose neighborhoods do not intersect the neighborhood we choose for (i, j). In this chapter, we use the following edge detection criterion:

$$\delta_{ij} = \min\left(||\vec{v}_{ij} - \vec{v}_{B_1}||, ||\vec{v}_{ij} - \vec{v}_{B_2}||\right), \quad \text{where } \vec{v}_{ij} = \left(\widehat{\beta}_1^{(i,j)}, \widehat{\beta}_2^{(i,j)}\right).$$

This approach is similar to the one proposed by Qiu and Yandell (1997). δ_{ij} should be large if and only if (x_i, y_j) is close to a JLC. Qiu and Yandell (1997) show that under certain regularity assumptions, $\delta_{ij} > \widehat{\sigma}\sqrt{\frac{\chi^2_{2,\alpha_n}}{kS_x^2}}$ can be used as an edge pixel detection criterion. Here, k is the window width and χ^2_{2,α_n} is the $(1 - \alpha_n)$ quantile of χ^2_2 distribution, and S_x^2 is the sample variance of the X-coordinates in a local neighborhood.

2.2.2 Local smoothing

In this step of the proposed method, we find an elliptical neighborhood around each pixel of the image such that no detected edge pixels are inside it. To accomplish this, we execute the following steps:

1. We first find out the nearest edge pixel to a given pixel P. Call that edge pixel P_1.

2. Next, we consider the point M, which is the reflection of the point P_1 with respect to the given point. Then, we find the nearest edge pixel in the strip perpendicular to the line joining M and P_1, say P_2.

3. Now, consider the ellipse E, centered at P, with half-lengths of the major and minor axes given by $|PP_2| - \gamma$ and $|PP_1| - \gamma$, respectively. The tuning parameter γ ensures that the selected ellipse neither intersects nor touches any JLC. See Figure 2.1 for demonstration.

The point on the ellipse E along the minor axis that is closest to P_1 corresponds to the intersection of the line segment PP_1 and E. This intersection point, denoted by P^*, lies exactly γ distance away from P_1, i.e., $|P^*P_1| = \gamma$. If a point $\overline{P}$ on a JLC lies within a distance less than γ from the ellipse, then, by the triangle inequality, it follows that $|P\overline{P}| < |PP_2|$, which contradicts the assumption that P_2 is the nearest point on the JLC to P within the strip. Therefore, the ellipse maintains a minimum distance of γ from all points on any JLC.

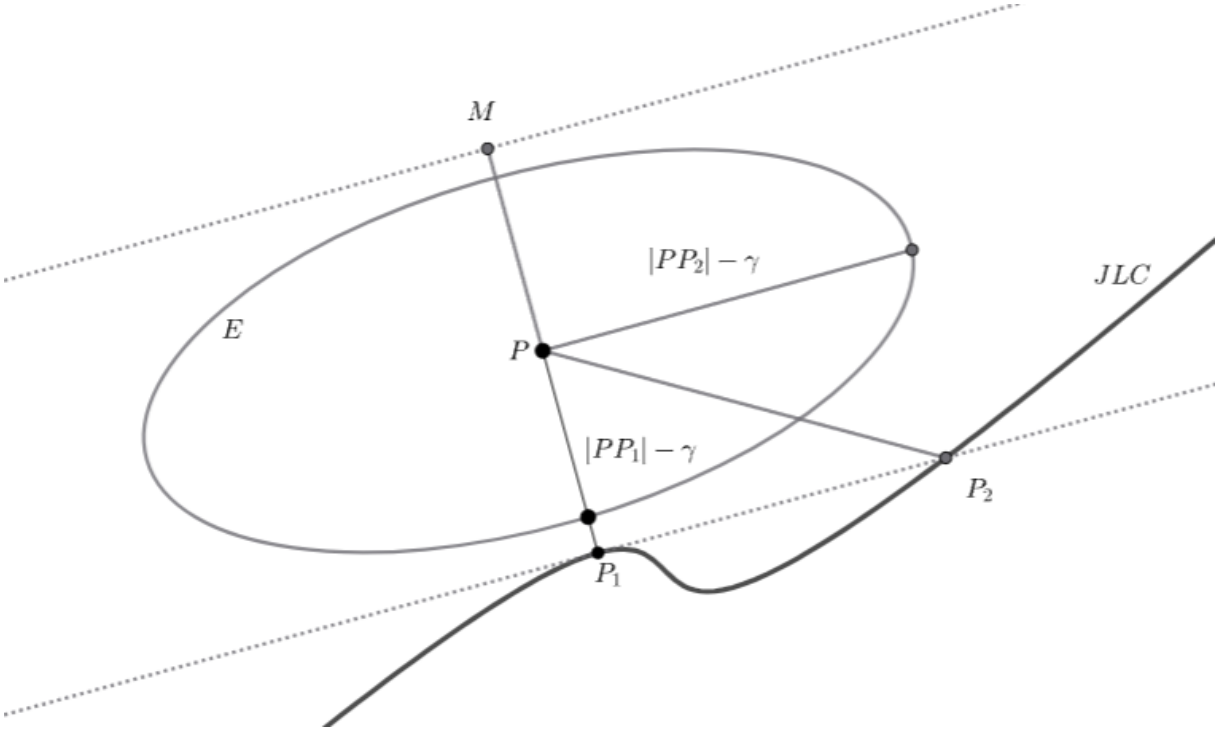


Figure 2.1: Figure demonstrating how the elliptical neighborhood around the pixel P is determined.

We then use a suitable kernel function which is non-negative only inside E and zero outside it, and estimate the true intensity of the pixel at P using kernel regression. In kernel regression, the observed pixel intensities are given by:

$$z_{\tilde{p}} = f(\tilde{p}) + \epsilon_{\tilde{p}},$$

where f is an unknown function and $\tilde{p}$ is the coordinates of a pixel on the image. We assume that the true intensity function f is smooth up to N-th order. If $\tilde{p}_i$ is close to $\tilde{p}$, we can use the Taylor series expansion to get:

$$\begin{aligned} z(\tilde{p}_i) &\approx z(\tilde{p}) + (\nabla z(\tilde{p}))^T(\tilde{p}_i - \tilde{p}) + \frac{1}{2}(\tilde{p}_i - \tilde{p})^T \mathcal{H}_z(p)(\tilde{p}_i - \tilde{p}) + \dots \\ &= \theta_0 + \theta_1^T(\tilde{p}_i - \tilde{p}) + \theta_2^T vech(\tilde{p}_i - \tilde{p})(\tilde{p}_i - \tilde{p})^T + ... \end{aligned}$$

where $\theta_0 = z(\tilde{p})$, $\theta_1 = \nabla z(\tilde{p})$, $\theta_2 = \frac{1}{2}\left[\frac{\partial^2 z(\tilde{p})}{\partial \tilde{p}_1^2}, 2\frac{\partial^2 z(\tilde{p})}{\partial \tilde{p}_1 \partial \tilde{p}_2}, \frac{\partial^2 z(\tilde{p})}{\partial \tilde{p}_2^2}\right]$, $vech\begin{pmatrix} a & b \\ b & d \end{pmatrix} = (a, b, d)$ and $\mathcal{H}_z(p)$ is the Hessian matrix evaluated at $\tilde{p}$. Therefore, we formulate this as a regression problem and estimate θ_0, θ_1, ... by minimizing :

$$\sum_{i=1}^{N_p} \left[z(\tilde{p}_i) - \theta_0 - \theta_1^T(\tilde{p}_i - \tilde{p}) - \theta_2^T vech(\tilde{p}_i - \tilde{p})(\tilde{p}_i - \tilde{p})^T + ... \right]^2 K\left(\frac{\tilde{p}_i - \tilde{p}}{h}\right)$$

where N_p is the number of pixels in the local neighborhood, K is a kernel of our choice, and h is our chosen bandwidth. We do this only for those pixels where the distance between P and P_1 is large enough.

For pixels which are closer to the detected edge pixels, i.e., within γ distance of an edge pixel, we are using the clustering method proposed by Mukherjee and Qiu (2015). Consider a neighborhood $O(x, y, h_n)$ around one such point (x, y). We can choose a threshold s, so that the pixels with intensity value $z_{ij} \leq s$ are in one cluster. Then we are choosing s such that it maximizes $T(x, y, h_n, s)$, the ratio of between-group and within-group variability of the intensity values.

$$T(x, y, h_n, s) = \frac{|O_1(x, y, h_n, s)|(\bar{z}_1 - \bar{z})^2 + |O_2(x, y, h_n, s)|(\bar{z}_2 - \bar{z})^2}{\sum\limits_{(x_i, y_j) \in O_1(x, y, h_n, s)} (z_{ij} - \bar{z}_1)^2 + \sum\limits_{(x_i, y_j) \in O_2(x, y, h_n, s)} (z_{ij} - \bar{z}_2)^2},$$

where $O_1(x, y, h_n, s)$ and $O_2(x, y, h_n, s)$ are the two clusters created using threshold s. And $\bar{z}_1$ and $\bar{z}_2$ are the average of intensity values in the respective clusters. We can choose the optimal value of s using:

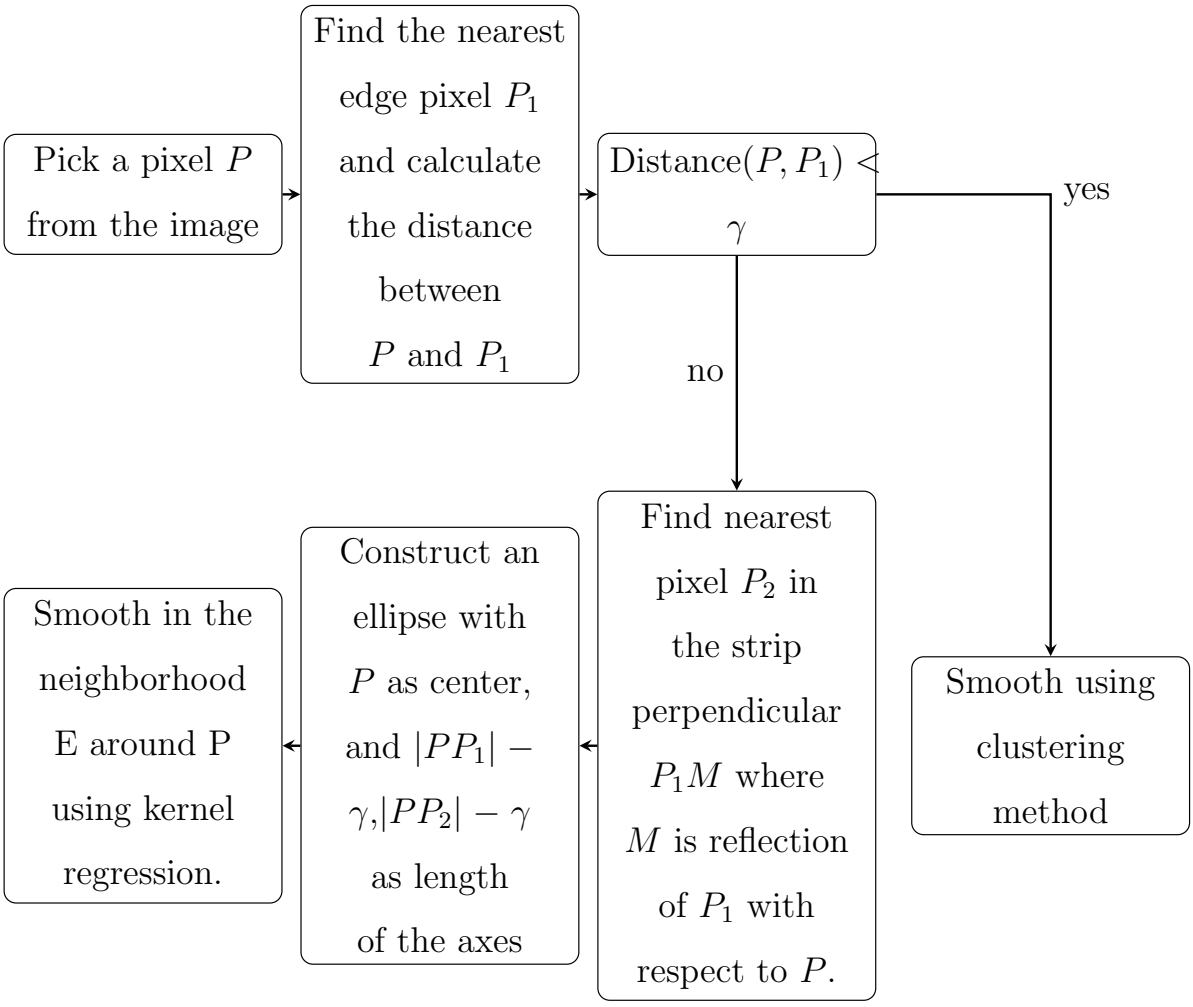
$$S_0 = \arg\max T(x, y, h_n, s).$$

After dividing the neighborhood $O(x, y, h_n)$ into clusters, we estimate the true image intensity at (x, y) using the weighted average of intensity values of the pixels in the cluster containing (x, y). In this chapter, we use a similarity measure between pixels as weights, which is given by :

$$\tilde{W}_{ij} = exp\left(- \frac{||\tilde{O}(x_i, y_j) - \tilde{O}(x, y)||_2^2}{2\hat{\sigma}^2 |\tilde{O}(x_i, y_j)| B_n} \right),$$

where $||\tilde{O}(x_i, y_j) - \tilde{O}(x, y)||_2^2$ is the sum of squares of the pixelwise differences of the intensities between the neighborhoods $\tilde{O}(x_i, y_j)$ and $\tilde{O}(x, y)$, $|\tilde{O}(x_i, y_j)|$ is the number of pixels in the neighborhood $\tilde{O}(x_i, y_j)$, $\hat{\sigma}$ is the estimated standard deviation of errors. and B_n is a tuning parameter. We are using the L_2 distance between the observed intensity values as the distance here.

The following flow chart summarizes the proposed integrated image denoising method. Please read the chart along with Figure 2.1.



2.2.3 Selection of procedure parameters

For our proposed algorithm, we need four parameters: the bandwidth and cutoff for the edge detection step, the bandwidth parameters for the local clustering step and smoothing step, and one tuning parameter for the smoothing step after local clustering. In our implementation, we introduce only one new parameter, the maximum possible axis length of the elliptical neighborhood. From numerical experience, we set this parameter to 6 pixels for images having a resolution less than 100×100, and 10 pixels for images with higher resolution. Similarly, the bandwidth for the clustering method is chosen to be the same as the minimum distance γ to apply the method, which we choose to be 3 pixels for images with resolution less than 100×100, and 5 pixels otherwise. The rest of the parameters are chosen according to the guidelines mentioned by Mukherjee and Qiu (2015) and Qiu and Yandell (1997).

The following table illustrates the effect of the edge-detection bandwidth parameter and cutoff on the performance of the proposed algorithm. As the bandwidth decreases, the quality of edge detection deteriorates, leading to poorer overall performance. This behavior can also be interpreted as a measure of the robustness of the proposed algorithm when the underlying denoising method performs suboptimally.

Cutoff	Bandwidth	Mean RMSE	SD RMSE
0.9	1.5	7.039	0.214
	2.0	7.726	0.194
	2.5	6.344	0.254
	3.0	6.808	0.340
0.925	1.5	6.760	0.245
	2.0	7.673	0.238
	2.5	6.207	0.254
	3.0	6.602	0.320
0.95	1.5	6.279	0.212
	2.0	7.514	0.162
	2.5	5.935	0.204
	3.0	6.386	0.322
0.975	1.5	5.864	0.196
	2.0	7.306	0.189
	2.5	5.791	0.190
	3.0	6.053	0.202
0.99	1.5	5.726	0.150
	2.0	6.572	0.357
	2.5	5.647	0.169
	3.0	5.940	0.314

Table 2.1: Simulation results for the denoising algorithm on the square-circle image using different weights and bandwidths for edge detection

2.2.4 Computational efficiency

As our algorithm integrates two different algorithms with distinct computation times, comparison of their computational efficiencies is essential. Suppose $\mathcal{C}_1$ denotes the computation time of the edge detection algorithm, $\mathcal{C}_2$ and $\mathcal{C}_3$ denote the computation times of the regression method and the clustering method per pixel, respectively. Hence, the total computation time $\mathcal{C}_{new}$ of the integrated algorithm is given by:

$$\mathcal{C}_{new} = \mathcal{C}_1 + \sum_{x,y}(P(x,y)\mathcal{C}_2 + (1 - P(x,y))\mathcal{C}_3),$$

where $P(x, y)$ represents the probability that the nearest edge pixel to the point (x, y) lies at a distance greater than γ. Noting that $\mathcal{C}_1 + n^2 \min(\mathcal{C}_2, \mathcal{C}_3) \leq \mathcal{C}_{new} \leq \mathcal{C}_1 + n^2 \max(\mathcal{C}_2, \mathcal{C}_3)$, we conclude that the computation time of the proposed integrated method lies between those of a single iteration of the two baseline algorithms. Moreover, the regression-based smoothing by Takeda et al. (2007) involves multiple iterations, hence its computation time is $\mathcal{T} n^2 \mathcal{C}_2$, $\mathcal{T}$ being the number of iterations. The computation time of the local clustering method by Mukherjee and Qiu (2015) is $(\mathcal{C}_1 + n^2 \mathcal{C}_3)$ with $\mathcal{C}_3$ being slightly larger than $\mathcal{C}_2$, but of similar order with respect to n. Hence, for sufficiently large $\mathcal{T}$ and n, $\mathcal{C}_{new}$ is smaller than both $\mathcal{T} n^2 \mathcal{C}_2$ and $(\mathcal{C}_1 + n^2 \mathcal{C}_3)$.

Although the proposed integrated algorithm involves a two-step procedure, apart from initial edge detection, the rest of the algorithm can be performed independently on each pixel. Therefore, the second step of this integrated algorithm can be parallelized. Furthermore, by choosing a computationally lightweight edge detection algorithm, we can significantly reduce the overall computation time.

2.3 Statistical Properties

In this section, we discuss important theoretical properties of the proposed integrated procedure. The following result justifies its asymptotic consistency.

Theorem 1: Assume that f has continuous first-order derivatives over $(0, 1) \times (0, 1)$ except on JLCs and the first-order right and left derivative exists at the JLCs, ϵ_{ij} are independent and have distribution $N(0, \sigma^2)$ with $0 < \sigma^2 < \infty$. If we have $h_n^* = O(n^\beta)$ with $\beta < 1$, $\alpha_n = O(1/\log\log n)$, $h_n = o(1)$, $1/nh_n = o(1)$ and the clustering bandwidth $\gamma_n < h_n$ with $\gamma_n = O(n^{-1} \log n)$, where h_n^* is the bandwidth parameter for edge detection and h_n is the bandwidth parameter for smoothing. Then, for any non-singular point (x, y) with the property $d((x, y), (x^*, y^*)) > \epsilon$ for all singular points (x^*, y^*), we have $\widehat{f}(x, y) = f(x, y) + O(\max(h_n, \gamma_n^{1/2}))$. where $\widehat{f}(x, y)$ is the estimate of $f(x, y)$ produced by the proposed integrated method.

Remark: Although we make a normality assumption of the additive noise for simplicity of the proof, it is not critically important. As long as the density of the error distribution is symmetric around zero and has finite moments at least up to a certain order, point-wise

convergence of $\widehat{f}(x,y)$ holds. If these assumptions on the error distribution do not hold, then we need to use other techniques, such as noise-adaptive transforms or pre-processing for non-Gaussian noise. Proof of the above theorem is based on Mukherjee and Qiu (2015) and Qiu and Yandell (1997).

Let us define the following notations:

$$\Omega_\epsilon = [\epsilon, 1-\epsilon] \times [\epsilon, 1-\epsilon],$$
$$S_\epsilon = \{(x,y) : (x,y) \in \Omega, d_E((x,y),(x^*,y^*)) \le \epsilon \text{ for a singular point } (x^*,y^*) \in D\},$$
$$\Omega_{\bar{S},\epsilon} = \Omega_\epsilon \backslash S_\epsilon.$$

Next, we proceed with the following Lemmas:

Lemma 1. *Assume that $f(x,y)$ has continuous first-order partial derivatives over $(0,1)\times(0,1)$ except on the JLCs, and it has the first-order right and left partial derivatives at the JLCs. And also $\lim_{n\to\infty} h^* = \infty$ and $\lim_{n\to\infty} h^*/n = 0$ where h^* is the window width. If there is no jump in $N(x_i,y_j)$, then*

$$\hat{\beta}_1^{(i,j)} = f'_x(x_i,y_j) + O\left(\frac{n\sqrt{\log\log h^*}}{{h^*}^2}\right) \quad a.s.$$

and

$$\hat{\beta}_2^{(i,j)} = f'_y(x_i,y_j) + O\left(\frac{n\sqrt{\log\log k}}{k^2}\right) \quad a.s.$$

If (x_i,y_j) is on a JLC and it is not a singular point, then

$$\hat{\beta}_1^{(i,j)} = f'_x(\tilde{x}_i,\tilde{y}_j) + h_1^{(i,j)}C(i,j) + \gamma_1 C_x(i,j) + O\left(\frac{n\sqrt{\log\log h^*}}{{h^*}^2}\right) \quad a.s.$$

and

$$\hat{\beta}_2^{(i,j)} = f'_y(\tilde{x}_i,\tilde{y}_j) + h_2^{(i,j)}C(i,j) + \gamma_2 C_y(i,j) + O\left(\frac{n\sqrt{\log\log h^*}}{{h^*}^2}\right) \quad a.s.,$$

where $(\tilde{x}_i,\tilde{y}_j)$ is some point around (x_i,y_j) on the same side of the JLC as (x_i,y_j) and the distance between $(\tilde{x}_i,\tilde{y}_j)$ and (x_i,y_j) tends to zero; $C(i,j), C_x(i,j)$, and $C_y(i,j)$ are the jump magnitudes of $f(x,y)$ and its first order x and y partial derivatives; $h_1^{(i,j)}$ and

$h_2^{(i,j)}$ *are two constants satisfying*

$$\sqrt{\left(h_1^{(i,j)}\right)^2 + \left(h_2^{(i,j)}\right)^2} = O(n/h^*),$$

γ_1 *and* γ_2 *are two constants between* -1 *and 1.*

Lemma 2. *Besides the conditions stated above, if there are only a finite number of singular points on JLCs and* α_n *is chosen such that*

i. $\lim_{n\to\infty} \alpha_n = 0$;

ii. $\lim_{n\to\infty} \log(\alpha_n) / \log(\log(h^*)) = -\infty$;

iii. and $\lim_{n\to\infty} \log(\alpha_n) / h^{*2} = 0$,

the point-set of the detected jumps is strongly consistent in the Hausdorff distance, and the convergence rate is $O(n^{-1}\log(n))$.

Lemma 3. *Assume that* f *has continuous first-order derivatives over* $(0,1)\times(0,1)$ *except on the JLCs, its first order derivatives have one-sided limits at non-singular points of the JLCs,* $\{\varepsilon_{ij}\}$ *are i.i.d. and have the common distribution* $N(0,\sigma^2)$*, where* $0 < \sigma < \infty$*,* $h_n = o(1), 1/nh_n = o(1), u_n = \kappa + \delta$*, where* $\kappa = \left(\frac{\phi^2(0)}{\Phi(0)(1-\Phi(0))}\right) / \left[1 - \left(\frac{\phi^2(0)}{\Phi(0)(1-\Phi(0))}\right)\right]$*,* δ *is any positive number, and* ϕ *and* Φ *are the probability density function and the cumulative distribution function of the* $N(0,1)$ *distribution, respectively. If we further assume that* $B_n = O\left(h_n^{1/2}\right)$*, then for any non-singular* $(x,y) \in \Omega_{\bar{S},\epsilon}$*, we have* $\widehat{f}(x,y) = f(x,y) + O\left(h_n^{1/2}\right)$*, a.s.*

Proofs of Lemmas 1 and 2 are provided in Qiu and Yandell (1997), whereas Mukherjee and Qiu (2015) provide a sketch of the proof of Lemma 3.

From Lemmas 1 and 2, we can say that if we choose $\alpha_n = O(1/log(log(n))$ and $h^* = O(n^\beta)$ where $\beta < 1$, all the above conditions are satisfied for edge detection.

Now, suppose we are using h_n^* window width and α_n confidence level for edge detection and h_n window width with $h_n = o(1)$ and γ_n as bandwidth parameter for clustering

according to Lemma 3 for the smoothing process, all of the conditions required regarding the parameters of the above lemmas are satisfied. We now proceed to prove Theorem 1.

Proof of Theorem 1:

Proof. Let J denote the points on JLCs and $\hat{J}_n$ denote the detected edge pixels. Suppose we are given the point $(\tilde{x}, \tilde{y})$.

Suppose the point is not on J. Then, as $\mathcal{H}(J, \hat{J}_n) \to 0$, for large enough n, it is not in $\hat{J}_n$ with probability n. So, without loss of generality, we can assume it is not on $\hat{J}_n$. Suppose the nearest point on $\hat{J}_n$ is (x^*, y^*).

There exists N such that for all $n > N$, we have $\mathcal{H}(J, \hat{J}_n) < \frac{\gamma_n}{2}$. So we can say $\forall n > N$ there is a circle around (x^*, y^*) of radius $d((x^*, y^*), (\tilde{x}, \tilde{y})) - \gamma_n$ which does not touch $\hat{J}_n$.

Now, for choosing the second closest point in our algorithm (point P_2 in Figure 2.1), we are doing the same as above while restricting the image in that strip. So we can take $\hat{\Omega} = \Omega \cap S$ and $\hat{J}_n^{'} = \hat{J}_n \cap S$, where S is the strip chosen using the first step. So, from the fact that the ellipse we define is completely inside the strip and using the same argument as choosing the first point, we can say that the ellipse we are choosing has no point that is on a JLC. So, $f(x, y)$ has continuous first-order derivatives inside the elliptical neighborhood E. As the maximum possible axis length, $h_n \to 0$, the Taylor series converges, Hence $\hat{f}(\tilde{x}, \tilde{y}) \to f(\tilde{x}, \tilde{y})$ i.e $\widehat{f}(\tilde{x}, \tilde{y}) = f(\tilde{x}, \tilde{y}) + O(h_n - \gamma_n) = f(\tilde{x}, \tilde{y}) + O(h_n)$, a.s.

Now if $(\tilde{x}, \tilde{y})$ is on J. Then as $\mathcal{H}(J, \hat{J}_n) \to 0$, we can say, $d((\tilde{x}, \tilde{y}), \hat{J}_n) < \gamma_n$ for large enough n. So, for large enough n, we will be using the method proposed by Mukherjee and Qiu (2015). So, by Lemma 3, we get, $\widehat{f}(\tilde{x}, \tilde{y}) = f(\tilde{x}, \tilde{y}) + O\left(\gamma_n^{1/2}\right)$ a.s. Hence, Theorem 1 is proved. ■

□

2.4 Numerical Studies

In this section, we present several numerical experiments where we study the performance of the proposed integrated method, which we call NEW, in comparison with four competing methods. We consider three noise levels representing low, medium, and high, and various state-of-the-art denoising techniques for performance comparison. We use integrated root mean square error (IRMSE) to measure the performance of various image denoising methods, which is defined as:

$$\text{IRMSE} = E\left(\sqrt{\int_{z\in\Omega} (\widehat{f}(z) - f(z))^2}\right), \text{ where } \Omega = [0,1]\times[0,1].$$

In our numerical studies, we estimate IRMSE by performing multiple instances of repeated independent simulations and calculating the sample mean of root mean square error (RMSE) values.

$$\widehat{\text{IRMSE}} = \frac{1}{L}\sum_{\ell=1}^{L}\left(\sqrt{\sum_{(x_i,y_j)} (\widehat{f}_\ell(x_i,y_j) - f(x_i,y_j))^2}\right).$$

2.4.1 A brief description of the competing methods

Below, we provide a brief overview of the competing methods that we consider for the numerical study.

(i) Method based on local pixel clustering: The clustering method proposed by Mukherjee and Qiu (2015) is the one we use in our integrated algorithm when the given pixel is close to a jump location curve in the given image. We include it in our comparison study to demonstrate that our proposed method is at least as good as this method. Moreover, the proposed method can denoise a given image in a much lesser time by eliminating the need to perform clustering in smoother regions. We choose the window width based on empirical optimality while selecting other parameters as suggested by Mukherjee and Qiu (2015). For implementation, we use the computer codes provided in the supplementary materials of Mukherjee and Qiu (2015).

(ii) Steering kernel algorithm: The steering kernel method proposed by Takeda

et al. (2007) uses an adaptive approach to denoise a given image. Instead of calculating the edge pixels directly, it uses gradient information to find a direction that should be parallel to the nearby edge, and determines an elliptical neighborhood elongated in that direction. Since we incorporate a similar approach near the jump location curves of the given image, it is natural to include this method in our comparison study. It is to be noted that the steering kernel method is an iterative method, i.e., it applies the denoising method several times until the estimated mean squared error is small enough, whereas our proposed method is not iterative, and hence computationally less expensive. We select the parameters of the steering kernel method as suggested by Takeda et al. (2007). We use the computer codes provided by Takeda (2007) for implementation.

(iii) Jump preserving local linear kernel (JPLLK) method: This procedure, proposed by Qiu (2009), involves a two-step process for surface reconstruction using local linear kernel smoothing. At a given point, a plane is fitted using local linear kernel smoothing to find a standard local linear kernel estimator of the image intensity function. The surrounding area is split into two sections by a line that passes through the point and is perpendicular to the gradient of the fitted plane. In each section, a half-plane is fitted using local linear kernel smoothing, resulting in two one-sided estimators of the image intensity function at that point. Subsequently, the best estimate among the two one-sided estimates, and the standard two-sided estimate is chosen as the final estimate at that point. This process is iterated twice to further smooth the surface. We select the bandwidth parameter of this smoothing procedure to get numerically optimal performance. Kang (2023) provides computer codes for execution.

(iv) Method by total variation regularization (TV): This is a very popular method used extensively in various applications. In this method, the total variation of an image is minimized using certain constraints related to the statistics of the noise. Rudin et al. (1992) show that reducing the total variation of an image removes noise while preserving essential structural details. Moreover, Chambolle and Pock (2011) develop a first-order primal-dual algorithm based on this method that works well for image denoising purposes alongside various other applications. We select parameters following the suggestion by Chambolle and Pock (2011). We use the computer codes provided by You (2021).

2.4.2 Simulation studies

We first construct a simple simulated image consisting of a square and a part of a circle. This image has various types of edges and edge structures, such as linear edges, corners, intersection of a line with a curve, and so forth. A noisy version of the simulated image is presented in the $(1, 1)$-th panel of Figure 2.2. We select such an image to demonstrate the performance of the proposed integrated method in comparison with its competitors. The outcomes are presented in Table 2.2 and Figure 2.2.

From Figure 2.2, we see that our method visually performs better than its competitors. Although the difference between our algorithm and the clustering algorithm is minimal, we can see that the clustering algorithm over-smooths the image, i.e., smudges the image around edges, while our method performs better in that regard. This is due to the fact that the clustering step in our integrated method selects a smaller neighborhood.

		Proposed Algorithm	Competing Methods			
Resolution	Noise SD	NEW	Steering Kernel	Clustering Method	JPLLK	TV
64×64	20	11.79 (0.317)	15.97 (0.300)	12.81 (0.991)	16.09 (0.278)	18.87 (0.252)
	10	5.47 (0.141)	7.98 (0.150)	8.28 (0.138)	19.34 (0.535)	8.93 (0.134)
	5	3.36 (0.081)	3.98 (0.075)	7.24 (0.057)	28.19 (0.857)	4.35 (0.104)
128×128	20	9.22 (0.119)	16.53 (0.123)	9.32 (0.105)	16.01 (0.112)	18.75 (0.132)
	10	4.43 (0.039)	8.26 (0.061)	7.96 (0.061)	12.10 (0.062)	8.82 (0.081)
	5	2.30 (0.035)	4.11 (0.031)	7.42 (0.031)	10.97 (0.035)	4.24 (0.099)

Table 2.2: Performance comparisons for simulated images using estimated IRMSE. Note that the contrasts of the true images are 255.

Table 2.2 shows the performance of the concerned methods in terms of IRMSE. We run each of the concerned methods on images with different instances of noise 30 times independently, and calculate the sample mean of RMSE as our estimated IRMSE. The numbers in parentheses show the standard error of our estimates. Table 2.2 reflects the same phenomenon we observe from Figure 2.2. The numerical performance of the proposed method is quite similar to the clustering method, but much better than other competitors. It is expected to be similar to the clustering method, as the proposed method is an improvement in terms of computational complexity. As seen from Table 2.2 and Figure 2.2, the proposed integrated method performs slightly better in the regions where the true image is smooth. This result is a small improvement in estimated IRMSE. In comparison with the steering kernel algorithm, the proposed method is better both visually and with respect to the estimated IRMSE values. JPLLK performs quite well in terms

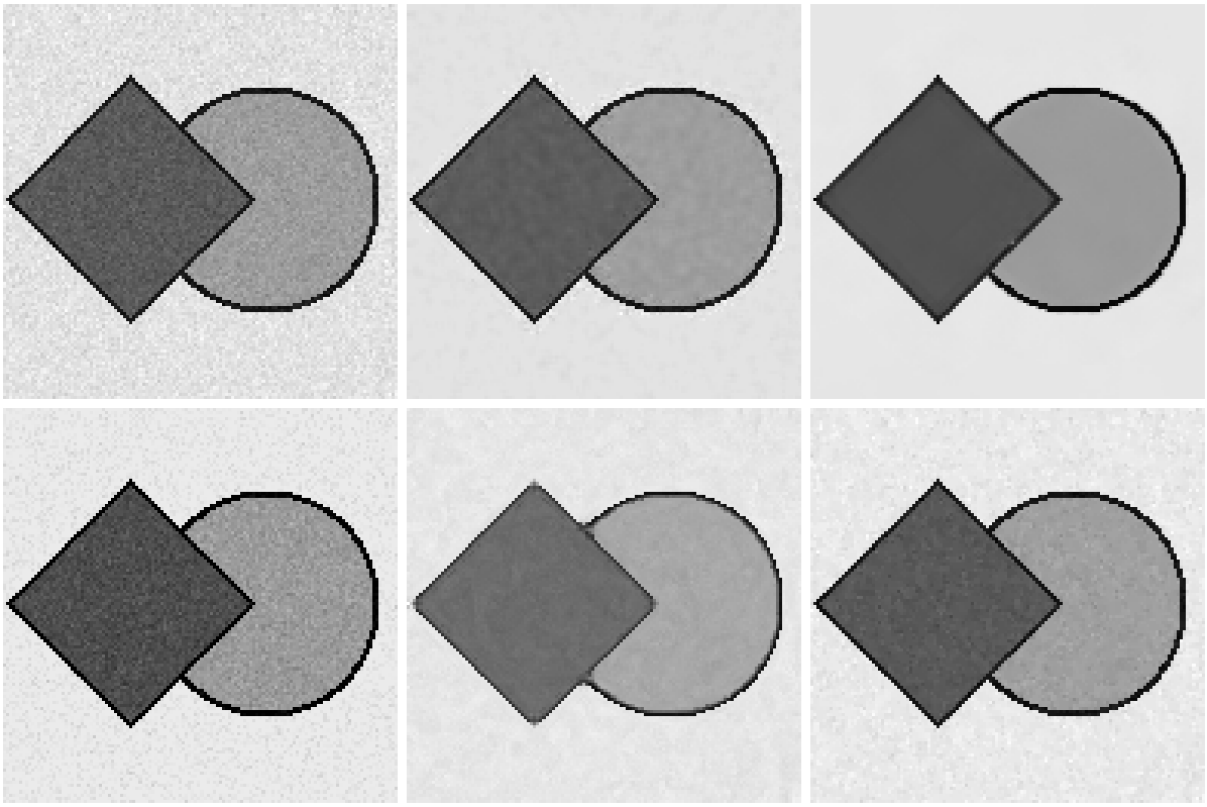


Figure 2.2: Denoising performance on the simulated image with added Gaussian noise of standard deviation 20. Images for top-left to bottom-right are the noisy image, denoised images by the proposed method, clustering algorithm, steering kernel, JPLLK, and TV, respectively.

of smoothing visually, but it performs poorly where two jump location curves intersect, which is reflected in the estimated IRMSE values. While the TV method performs well in low noise levels, it performs poorly in high noise levels. Our proposed integrated method performs better in comparison to its competitors, especially when the noise level is high. The performance of the proposed integrated method, along with the baseline methods are very similar when the noise distribution is non-Gaussian but symmetric around zero with finite first few moments. Performances of the jump regression-based image denoising methods have been explored in the literature in the presence of such kinds of non-Gaussian noise, and these performances are very similar to the Gaussian noise cases (Mukherjee and Qiu, 2015; Qiu and Mukherjee, 2010). These observations are in line with our remark after Theorem 1.

2.4.3 Performance comparison on real images

In this subsection, we consider two real images, a cityscape and a house, to evaluate the performance of the denoising methods. The corresponding noisy versions of these images are shown in the first panels of Figures 2.3 and 2.4.

We also include a state-of-the-art deep learning model among the competing methods. It is important to note that this does not constitute a direct comparison, as deep learning models are typically trained on large and diverse image datasets, whereas the proposed method is designed for a single-image setting. The model, *Restormer*, proposed by Zamir et al. (2022), is a Transformer-based architecture developed specifically for image restoration tasks such as denoising. Unlike conventional convolutional approaches, it utilizes multi-head self-attention along with feed-forward networks to capture both fine local structures and long-range spatial dependencies. This enables *Restormer* to effectively remove noise while preserving structural details and overall image fidelity, leading to state-of-the-art performance in real-world image denoising and related restoration tasks. The outcomes are presented in Table 2.3, and Figures 2.3 and 2.4.

	City by water			House		
	SD=5	SD=10	SD=20	SD=5	SD=10	SD=20
Proposed Method	2.82 (0.010)	4.69 (0.016)	7.92 (0.029)	2.51 (0.009)	4.37 (0.017)	7.65 (0.028)
Steering Kernel	5.33 (0.012)	9.96 (0.017)	15.34 (0.069)	4.99 (0.139)	9.78 (0.020)	15.22 (0.073)
Clustering Method	2.88 (0.011)	4.87 (0.026)	8.52 (0.059)	2.49 (0.018)	4.60 (0.031)	8.63 (0.065)
JPLLK	7.42 (0.015)	7.81 (0.025)	9.12 (0.042)	6.40 (0.012)	6.86 (0.024)	8.35 (0.046)
TV	3.84 (0.012)	8.63 (0.026)	18.62 (0.033)	3.87 (0.020)	8.86 (0.029)	18.66 (0.044)
Restormer	3.26 (0.022)	4.40 (0.018)	8.10 (0.046)	2.64 (0.020)	3.77 (0.043)	7.86 (0.061)

Table 2.3: Performance comparisons for real images using estimated IRMSE. Note that the contrasts of the true images are 255.

From Table 2.3, we observe similar patterns as in the case of the simulated image. The proposed algorithm outperforms the steering kernel method, JPLLK, and TV in terms of estimated IRMSE for both of the images across all three noise levels that we consider. The performance of the proposed algorithm and the local clustering method continues to be similar both visually and in terms of estimated IRMSE. In fact, the estimated IRMSE of the proposed method is smaller than that of the local clustering method in several cases. The steering kernel algorithm and TV method perform reasonably well when the noise level is low. The proposed method outperforms them in all cases with bigger margins at higher noise levels. In contrast, the performance of JPPLK is good at higher noise levels,

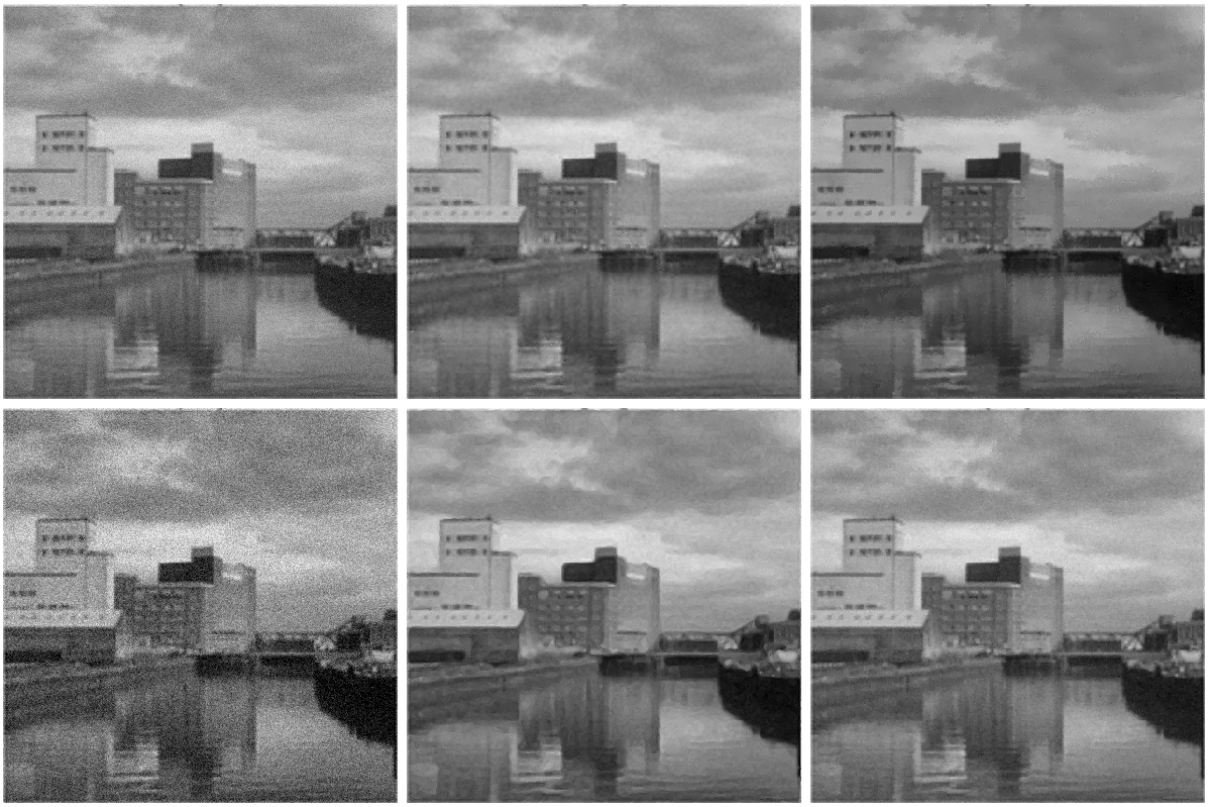


Figure 2.3: Performance comparison on the cityscape image with added Gaussian noise of standard deviation 10. Images for top-left to bottom-right are the noisy image, denoised images by the proposed method, clustering algorithm, steering kernel, JPLLK, and TV, respectively.

but it performs rather poorly at lower noise levels. Therefore, the proposed integrated algorithm performs well in nearly all situations without compromising the smoothing performance in the continuity regions while being less computationally expensive than the local clustering method.

Another important aspect to note is that the proposed algorithm performs better when the underlying edge structures of images are defined more clearly. This is because it relies heavily upon the edge detection algorithm we use. Therefore, in cases where edge pixels are not detected well, the proposed method ends up smoothing those edges. However, when the noise level is very high, this issue is usually overshadowed due to the heavy noise in comparison with the jump size. As for our comparison to *Restormer*, the proposed algorithm fairs almost as well as the transformer-based approach despite not being trained on multiple images and using simpler soothing algorithms.

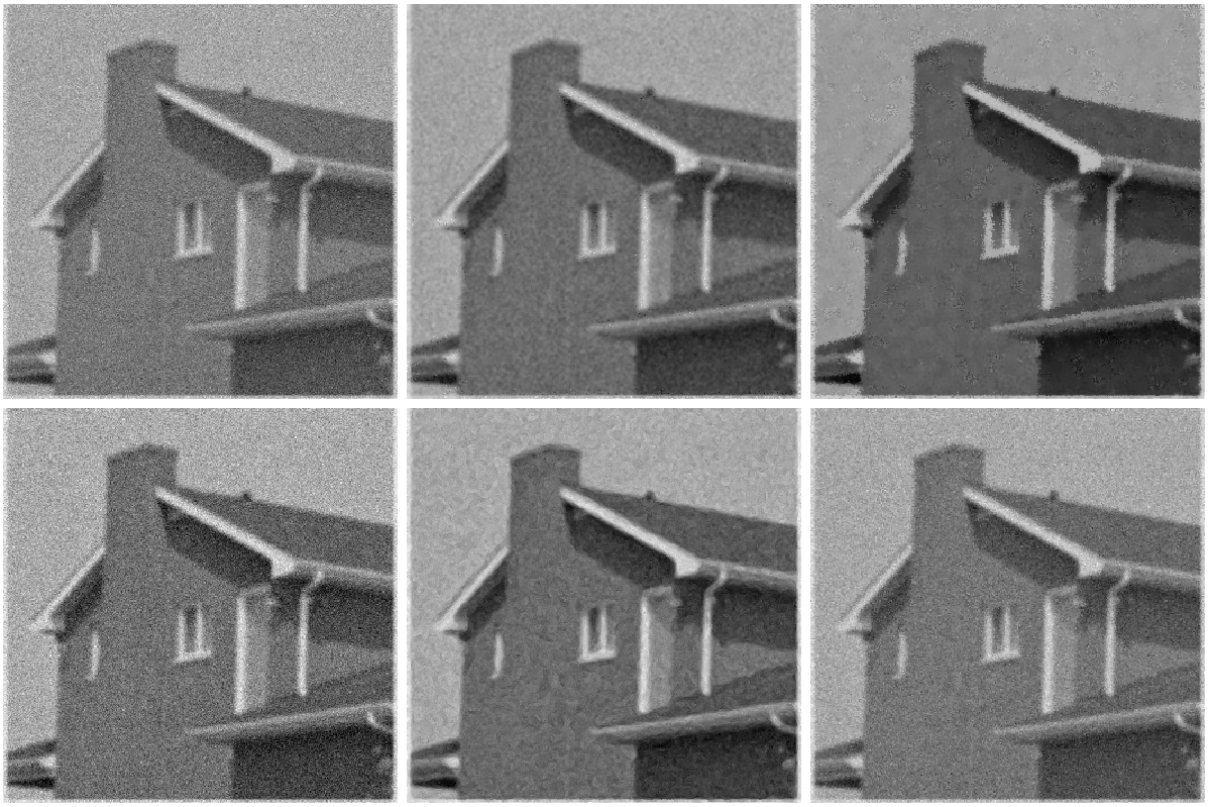


Figure 2.4: Performance comparison on the house image with Gaussian added noise of standard deviation 10. Images for top-left to bottom-right are the noisy image, denoised images by the proposed method, clustering algorithm, steering kernel, JPLLK, and TV, respectively.

2.5 Concluding Remarks

In this chapter, we propose an integrated image denoising procedure that detects the edge pixels first, and then proceeds to denoise using either adaptive kernel regression or local clustering depending on local information regarding edges. This chapter shows better performance when compared to similar methods already in the literature. We also provide a theoretical justification that the image intensity function of the denoised image converges point-wise to the true image intensity function for large image size, under mild regularity assumptions. However, there are scopes for extending the proposed approach. In this chapter, we consider the local clustering algorithm developed by Mukherjee and Qiu (2015) as a baseline for denoising around edges. This can be improved by using more recent image denoising methods. Though it is restricted by the framework of the problem that we consider in this chapter, one may expand this approach to use other types of

denoising methods, such as wavelet-based methods, diffusion-based methods, and many more. Another direction of potential improvement is the number of methods we can stitch together using this approach. In this chapter, we divide the problem into just two parts, namely, smoothing near the edges and smoothing far away from the edges. However, in future research, one may divide the problem into more parts, and thus have an assortment of methods that perform well in specific scenarios which altogether produce a superior overall result. One immediate application of the proposed image denoising method is in designing techniques for image comparison and image monitoring.

Chapter 3

Estimation of Piecewise Continuous Regression Function in Finite Dimension using Oblique-axis Regression Tree with Applications in Image Denoising[†]

3.1 Introduction

Image denoising can be formulated as a surface estimation problem, where the underlying surface representing image intensity is not necessarily continuous. As observed in real-world images, these discontinuities, commonly referred to as edge pixels, often align along smooth curves that signify prominent image features such as object boundaries. If the structure of these edges can be accurately identified, denoising can be performed in a manner that preserves these critical regions. This objective can be achieved using decision tree–based algorithms, which aim to learn partitioning rules that segment the image data

[†]This chapter is based on the publication Basak et al. (2026): **Basak, S.**, Roy, A. and Mukherjee, P.S. "Estimation of Piecewise Continuous Regression Function in Finite Dimension using Oblique-axis Regression Tree with Applications in Image Denoising", *Statistica Sinica*, to appear.
DOI: 10.5705/ss.202025.0120.

into homogeneous regions, thereby facilitating noise reduction while maintaining edge integrity. As discussed earlier, CART (Classification and Regression Trees), introduced by Breiman et al. (1984), is one of the most well-known methods for constructing decision trees. While CART relies on axis-aligned splits based on a single predictor variable at each node, this framework has been extended by Oblique Regression Trees, which generalize the decision-making process by allowing splits based on linear combinations of multiple predictor variables. This added flexibility enables more accurate partitioning, especially in scenarios where the underlying decision boundaries are not aligned with the coordinate axes. However, these methods often rely on assumptions about the functional form of the underlying surface, which may not hold in the context of image data. The complex and heterogeneous nature of real-world images frequently violates such assumptions, potentially limiting the effectiveness of these approaches in accurately capturing image structures. So in this chapter, we introduce a nonparametric smoothing technique based on ORT, designed for scenarios where the true regression function is piecewise continuous. Given that a grayscale image intensity function can be represented as a piecewise continuous regression function, we demonstrate the application of the proposed algorithm on the problem of grayscale image denoising in the latter part of the chapter.

As previously discussed, existing literature on JRA offers useful tools for estimating discontinuous functional relationships. However, most of the methods focus on cases with only one or two predictor variables and are not easily extendable to scenarios with multiple predictors. Recently, Kang et al. (2021) propose a jump regression model that accommodates multiple predictors, but their approach assumes an additive structure in the functional relationship, effectively disregarding interaction terms among the predictors. To elucidate this limitation, consider the model: $w = f(z_1, z_2) + \varepsilon$ with two predictor variables z_1 and z_2. Now, the additive model i.e., $f(z_1, z_2) = f_0 + f_1(z_1) + f_2(z_2)$ is suitable for preserving the jump location curves parallel to X-axis and Y-axis. However, this model is not flexible enough to describe and preserve the jump location curves in other directions. The proposed jump regression model overcomes such limitations of the JRA literature mentioned above. The proposed algorithm does not require pruning for estimating the underlying function; however, we use some constraints to limit the depth of the tree, which may lead to significant computational advantages. Many of the recent tree-based algorithms, e.g., Zhan et al. (2026) assume the underlying function can be expressed as

a *ridge* function. In practice, it is often not a valid assumption. For instance, the image intensity functions are discontinuous in nature and not necessarily a ridge function. In this chapter, we do not make any such strong assumptions. Instead, we impose certain assumptions on the underlying discontinuous points to address this issue. The proofs of the theoretical properties of the proposed tree-based method are much simpler, and the overall framework of the proofs is very different from what is available in the literature of regression trees. This is due to a different set of assumptions on the underlying regression function. Theoretical properties and their proofs, thus, demonstrate the novelty of this chapter, in addition to excellent numerical results.

The remainder of this chapter is organized as follows. Section 3.2 describes the proposed methodology. We study its statistical properties in Section 3.3. Section 3.4 shows one major application of the proposed method in the form of image denoising. Numerical performance of the proposed algorithm is shown in Section 3.5. A few remarks in Section 3.6 conclude this chapter.

3.2 Proposed Methodology

In this paper, we work with the standard regression framework, and the statistical model can be expressed as

$$w(\boldsymbol{z}) = f(\boldsymbol{z}) + \varepsilon, \tag{3.1}$$

where w is real-valued response or output variable, $\boldsymbol{z} = (z_1, z_2, \dots z_p)$ is the predictor (input, feature, or covariate vector) variable which lie on an equally space lattice at the points $\left(\frac{i_1}{n}, \dots, \frac{i_p}{n}\right)$, where each $i_j \in 1, 2, \dots, n$ on the design space $\Omega = [0,1]^p$ and, ε is the error variable. We have a dataset $\mathcal{D}_n = \{(w_i, \boldsymbol{z_i}) : 1 \leq i \leq n^p\}$, consisting of independent samples drawn from the model in (3.1). Additionally, we assume that f is *piecewise continuous*, and has the following form:

$$f(\boldsymbol{z}) = \sum_{l=1}^{P} g_l(\boldsymbol{z}) \mathbb{I}_{\Gamma_l}(\boldsymbol{z}), \tag{3.2}$$

where $g_l(\boldsymbol{z}) = g(\boldsymbol{z}) + \delta_l$, and δ_l is the jump size in Γ_l. In literature, this is popularly known as jump regression analysis (JRA). Note that, $\{\Gamma_1, \Gamma_2, \dots, \Gamma_P\}$ is a finite partition

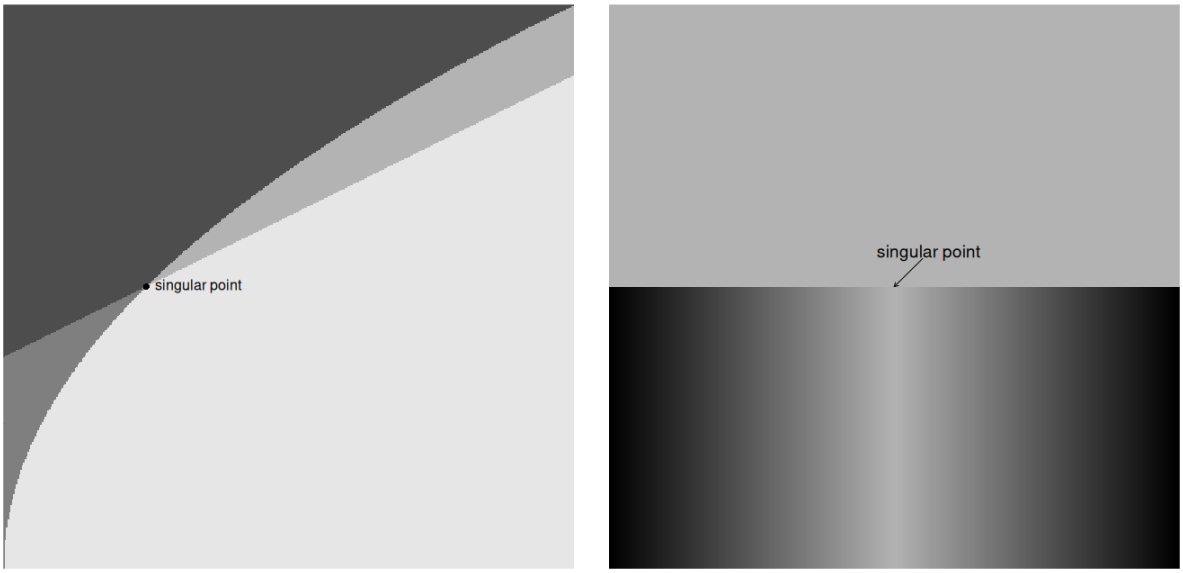


Figure 3.1: Two types of singular points are illustrated. The first image depicts the intersection of two JLCs, corresponding to condition (a) in the definition. The second image depicts a singular point defined via condition (b) in the definition.

of Ω such that: (i) Each Γ_l is a connected region in the design space. (ii) Defining $\partial\Gamma_l$ as the set of boundary points in Γ_l, $f(\boldsymbol{z})$ is continuous in $\Gamma_l \setminus \partial\Gamma_l$, for $l = 1, 2, \ldots, P$. (iii) $\bigcup_{i=1}^{P} \Gamma_l = \Omega$. (iv) There exists at most finitely many points $\{\boldsymbol{z}^*, k = 1, 2, \ldots, K^*\}$ in $[\bigcup_{i=1}^{n} \partial\Gamma] \bigcap \Omega$ such that for each $\boldsymbol{z}^*$ with $k = 1, 2, \ldots K^*$, there are $\Gamma_{k_1}^*, \Gamma_{k_2}^* \in \{\Gamma_l, l = 1, 2, \ldots P\}$ satisfying

(a) $\boldsymbol{z}^* \in \Gamma_{k_1}^* \bigcap \Gamma_{k_2}^*$, and
(b) $\lim\limits_{\substack{\boldsymbol{z} \to \boldsymbol{z}^* \\ \boldsymbol{z} \in \Gamma_{k_1}^*}} f(\boldsymbol{z}) = \lim\limits_{\substack{\boldsymbol{z} \to \boldsymbol{z}^* \\ \boldsymbol{z} \in \Gamma_{k_2}^*}} f(\boldsymbol{z})$.

Condition (a) implies that the point lies on multiple JLCs, whereas condition (b) implies that the limiting value across two partitions created by the JLCs match. In the imaging context, this means that two different partitions of the image are leaking into each other through a point on the relevant JLC. Such continuity points are sporadic and should be treated differently from the usual continuity points. Figure 3.1 demonstrates these two types of singular points.

Our primary objective is to develop a smoothing technique that effectively preserves the jump location curves (JLCs) in the regression surfaces. To achieve this, we partition

$\mathcal{D}_n$ in such a manner that each partition contains observations from one side of a JLC. To perform this, we construct a tree using the ORT algorithm, where each leaf node of the tree indicates one partition of the data and contains observations from one side of a JLC. To this end, each partition yields an estimator $\widehat{f}_n$ of f in model (3.2) via *leaf-only* averaging.

3.2.1 Recursive partitioning of the data

In this subsection, we introduce the algorithm by which we construct the recursive ORT, a pivotal step in the context of the proposed jump-preserving surface-smoothing technique. Note that we consider partitioning with respect to hyperplanes only. This restriction is imposed to ensure that the partitions remain convex, which is necessary for the subsequent theoretical proofs. However, including nonlinear functions of the covariates or the pixel coordinates in a meaningful way is a very challenging task, but it can potentially improve performance. Let $\mathcal{N}$ denotes a node of the ORT which is nothing but a subset of $[0,1]^p$, and consider its two children $\mathcal{N}_L = \{\boldsymbol{z} : \boldsymbol{\alpha}^T\boldsymbol{z} \leq c\}$ and $\mathcal{N}_R = \{\boldsymbol{z} : \boldsymbol{\alpha}^T\boldsymbol{z} > c\}$. Note that $\mathcal{N}_L$ and $\mathcal{N}_R$ cannot be empty, and satisfy $\mathcal{N}_L \cup \mathcal{N}_R = \mathcal{N}$. The central idea behind the tree construction is hierarchically partitioning the data in a greedy manner through a recursive binary splitting rule. Similar to the CART, a daughter node is treated as a parent node in the next step. Then, a parent node $\mathcal{N}$ at any depth is divided into two daughter nodes $\mathcal{N}_L$ and $\mathcal{N}_R$ by maximizing the impurity gain

$$\widehat{\Delta}(\mathcal{N}, \boldsymbol{\alpha}, c) = \frac{1}{|\mathcal{N}|}\left[\sum_{\boldsymbol{z}\in\mathcal{N}}\big(w(\boldsymbol{z}) - \bar{w}\big)^2 - \sum_{\boldsymbol{z}\in\mathcal{N}_L}\big(w(\boldsymbol{z}) - \bar{w}_L\big)^2 - \sum_{\boldsymbol{z}\in\mathcal{N}_R}\big(w(\boldsymbol{z}) - \bar{w}_R\big)^2\right] \quad (3.3)$$

with respect to $(\boldsymbol{\alpha}, c)$. Here, $\bar{w}$, $\bar{w}_L$, and $\bar{w}_R$ are defined as the sample averages of the response variable w_i at the node $\mathcal{N}$, $\mathcal{N}_L$, and $\mathcal{N}_R$, respectively. The impurity gain is nothing but the decrease in the error sum of squares (SSE) by increasing the depth of the tree by one step.

Suppose $(\widehat{\boldsymbol{\alpha}}, \widehat{c})$ maximizes the equation (3.3). We then split the node $\mathcal{N}$ if

$$\widehat{\Delta}(\mathcal{N}) = \widehat{\Delta}(\mathcal{N}, \widehat{\boldsymbol{\alpha}}, \widehat{c}) > r_n,$$

where r_n is the splitting threshold for the recursive tree partitioning. Since there is no pruning step in the proposed algorithm, we choose r_n in such a manner that ensures a

shallow tree construction. To this end, refinement of $\mathcal{N}$ produces two daughter nodes $\mathcal{N}_L = \{\boldsymbol{z} : \widehat{\boldsymbol{\alpha}}^T \boldsymbol{z} \leq \widehat{c}\}$ and $\mathcal{N}_R = \{\boldsymbol{z} : \widehat{\boldsymbol{\alpha}}^T \boldsymbol{z} > \widehat{c}\}$. These child nodes act as a parent node for the next level of the tree construction, and further recursive refinement carries forward in a similar fashion. At any level of the tree refinement, a node is considered a leaf node if $\widehat{\Delta}(\mathcal{N}, \boldsymbol{\alpha}, c) \leq r_n$, and we stop the recursion when all nodes become leaf nodes. Finally, we have the leaf nodes that partition the design space, where each partition is nothing but a convex polytope in $[0,1]^p$. Note that the decision tree construction is a filtering step for the proposed jump-preserving smoothing. The intuition behind this is to partition the data in a manner such that the data points on the same side of the JLCs always lie in the same estimated partition. Lemma 11 in Section 3.3 supports our intuition. A pseudo code outlining the tree construction method is presented in Algorithm 1.

Selection of r_n for numerical implementation: We choose r_n to be small enough to ensure that there are sufficient partitions to adequately represent the entire image. Note that if we choose this parameter too small, then it will only increase computation time without any meaningful improvement in the performance of the proposed method. Moreover, r_n should be proportional to the variance of the noise in the image, as that will cause the loss function we use to be proportionally large. Based on numerical simulations, we recommend using $r_n = 0.01\sigma^2$ for numerical implementations, and when σ^2 is unknown, estimate it by any standard estimator. For this choice of r_n, the algorithm exhibits an average runtime of 1.6 seconds * across images of sizes 523×523 and 628×628.

3.2.2 Proposed jump preserving surface estimator

Using the proposed algorithm, we divide the design space $[0,1]^p$ in leaf nodes, say $\{\mathcal{L}_n^i | 1 \leq i \leq K_n\}$, where K_n denotes the number of leaf nodes, and $\bigcup_{1 \leq i \leq K_n} \mathcal{L}_n^i = [0,1]^p$. For a given point $x \in [0,1]^p$, let $\mathcal{L}_n^{\boldsymbol{x}}$ denote the leaf node where it lies. Then, the proposed estimator is given by:

$$\widehat{f}_n(\boldsymbol{x}) = \frac{\sum_{z \in \mathcal{L}_n^{\boldsymbol{x}} \bigcap \mathcal{D}_n} w(\boldsymbol{z})}{|\mathcal{L}_n^{\boldsymbol{x}} \bigcap \mathcal{D}_n|}. \tag{3.4}$$

*All experiments were conducted on a Windows 11 system (via Windows Subsystem for Linux) equipped with an Intel Core i7-8700K CPU @ 3.70 GHz and 24 GB RAM. The algorithm was implemented in RcppArmadillo and executed in a multithreaded CPU setting using 12 threads, without GPU acceleration.

From now on, we denote $\mathcal{L}_n \bigcap \mathcal{D}_n$ by $\mathcal{L}_n$ for simplicity. Since the observations in $\mathcal{L}$ lie on one side of a JLC, the proposed estimator preserves the discontinuity well. In the application of image denoising, we slightly modify the proposed estimator. Refer to Eqn. 3.10 for details.

Algorithm 1 Recursive Tree Construction

Input: Given dataset $\mathcal{D}_n$ and r_n is the specified cutoff.
ℓ is the list of nodes and ℓ_1 is an empty list.
while ℓ is not empty, **do**
    Pick a node $\mathcal{N}$ from ℓ,
    Calculate $\tilde{\Delta}(\mathcal{N})$.
    if $\tilde{\Delta}(\mathcal{N}) \leq r_n$ **then**
        Remove $\mathcal{N}$ from ℓ
        Mark it as a leaf node by adding it to ℓ_1.
    else
        Calculate $(\widehat{\boldsymbol{\alpha}}_{opt}, \widehat{c}_{opt}) = \arg\max_{(\boldsymbol{\alpha},c)} \widehat{\Delta}(\mathcal{N}, \boldsymbol{\alpha}, c)$
        Split $\mathcal{N}$ into $\mathcal{N}_R$ and $\mathcal{N}_L$ using the line $\{\widehat{\boldsymbol{\alpha}}_{\boldsymbol{opt}}^{\boldsymbol{T}}\boldsymbol{z} = \widehat{c}\}$.
        Remove $\mathcal{N}$ from ℓ
        Add $\mathcal{N}_R$ and $\mathcal{N}_L$ to ℓ.
    end if
end while
Output: ℓ_1 is the list of leaf nodes.

3.3 Statistical Properties

In this section, we investigate certain statistical properties to substantiate our arguments in the proposed methodology. We derive novel theoretical results to establish the consistency of the oblique decision tree within the context of jump regression. Before going into the details, we introduce relevant notations and assumptions below.

Notations & Assumptions: We define a sequence of trees by $\{\mathcal{T}_n\}$, and the sequence of leaf nodes of $\mathcal{T}_n$ by $\{\mathcal{L}_n\}$, where $\mathcal{L}_{(n+1)} \subset \mathcal{L}_n$. We define $|\mathcal{L}_n|$ as the number of datapoints in $\mathcal{D}_n$ which are inside $\mathcal{L}_n$, and define μ as the Lebesgue measure on $[0,1]^p$. We now make the following assumptions:

(i) f is piecewise Lipschitz.

(ii) f is bounded. Without any loss of generality, we assume that it is bounded inside $[0,1]^p$.

(iii) The data-points in $\overline{\Lambda_\ell}/int(\Lambda_\ell)$ are called jump points, and the set of jump points has Lebesgue measure zero.

(iv) The set of jump location points is a piecewise lower-dimensional hyperplane. For notational convenience, each such hyperplane is considered to be a different JLC.

(v) There exists at most countably many JLCs.

(vi) If f is continuous at a jump point, then that point is called a singular point.

(vii) The pointwise noise ε are independently distributed and follows $\mathbf{N}(0,\sigma^2)$.

Next, we discuss some simplifications of the impurity gain measure:

Property 4.

$$\hat{\Delta}(\mathcal{N},\boldsymbol{\alpha},c)=\frac{|\mathcal{N}_L||\mathcal{N}_R|}{|\mathcal{N}|^2}\Big(\bar{w}_L-\bar{w}_R\Big)^2. \tag{3.5}$$

Proof: From Eq. (3.5), we have:

$$\begin{aligned}\widehat{\Delta}(\mathcal{N},\boldsymbol{\alpha},c)&=\frac{1}{|\mathcal{N}|}\Big(\sum_{\boldsymbol{z}\in\mathcal{N}}w(\boldsymbol{z})^2-\sum_{\boldsymbol{z}\in\mathcal{N}_L}w(\boldsymbol{z})^2-\sum_{\boldsymbol{z}\in\mathcal{N}_R}w(\boldsymbol{z})^2-|\mathcal{N}|\bar{w}^2+|\mathcal{N}_R|\bar{w}_R^2+|\mathcal{N}_L|\bar{w}_L^2\Big)\\&=\frac{1}{|\mathcal{N}|}\Big(|\mathcal{N}_R|\bar{w}_R^2+|\mathcal{N}_L|\bar{w}_L^2-|\mathcal{N}|(\frac{|\mathcal{N}_R|\bar{w}_R}{|\mathcal{N}|}+\frac{|\mathcal{N}_L|\bar{w}_L}{|\mathcal{N}|})^2\Big)\\&=\frac{1}{|\mathcal{N}|^2}\Big((|\mathcal{N}||\mathcal{N}_R|-|\mathcal{N}_R|^2)\bar{w}_R^2+(|\mathcal{N}||\mathcal{N}_L|-|\mathcal{N}_L|^2)\bar{w}_L^2-2|\mathcal{N}_R||\mathcal{N}_L|\bar{w}_L\bar{w}_R\Big)\\&=\frac{|\mathcal{N}_L||\mathcal{N}_R|}{|\mathcal{N}|^2}\Big(\bar{w}_L-\bar{w}_R\Big)^2.\end{aligned}$$

Theorem 5. $|\mathcal{L}_n|\to\infty$ *almost surely.*

Proof: Since $\mathcal{L}_n$ are leaf nodes, we have:

$$\widehat{\Delta}(\mathcal{L}_n)\le r_n.$$

Now, consider any $(\boldsymbol{\alpha}, c)$ that splits $\mathcal{L}_n$ in half. Then, using Eqn. (3.5) for that split, we have:

$$r_n \geq \frac{|\mathcal{L}_n^L||\mathcal{L}_n^R|}{|\mathcal{L}_n|^2}\left(\bar{w}_L - \bar{w}_R\right)^2.$$

We know $\frac{|\mathcal{L}_n^L||\mathcal{L}_n^R|}{|\mathcal{L}_n|}\left(\bar{w}_L - \bar{w}_R\right)^2$ follows noncentral $\chi_1^2(k)$ with a non-centrality parameter which we call k. Therefore, we have:

$$r_n \geq \frac{\chi_1^2(k)}{|\mathcal{L}_n|} \implies |\mathcal{L}_n| \geq \frac{\chi_1^2(k)}{r_n}.$$

As $\chi_1^2(k)$ is stocahastically larger than χ_1^2, we can say:

$$\mathbb{P}(|\mathcal{L}_n| \leq M) \leq \mathbb{P}(\chi_1^2 \leq Mr_n).$$

As $r_n \to 0$, we can conclude that $|\mathcal{L}_n| \to \infty$ in probability. If we choose r_n such that the RHS of the above inequality becomes summable, then we will have almost sure convergence as well. In that case, we use a bound given by Ghosh (2021):

$$\begin{aligned}\mathbb{P}(\chi_1^2 \leq Mr_n) &\leq exp\left[\frac{1}{2}\left(1 - Mr_n + log(Mr_n)\right)\right] \\ &= \sqrt{Mr_n}exp\left(\frac{1 - Mr_n}{2}\right) \leq \sqrt{eMr_n}.\end{aligned}$$

Therefore, if $\{\sqrt{r_n}\}$ is summable, we have $|\mathcal{L}_n| \to \infty$ almost surely.

Remark 6. *Observe that $\mu(\mathcal{L}_n)$ is non-increasing as $\mathcal{L}_{n+1}$ is a subset of $\mathcal{L}_n$. Since $\mu(\mathcal{L}_n)$ is non-increasing and non-negative, it must have a limit.*

Theorem 7. *If $\lim_{n\to\infty} \mu(\mathcal{L}_n) \neq 0$, then f is a.e. constant in $\bigcap_n \mathcal{L}_n$.*

Proof: Suppose, $\mu(\mathcal{L}_n) \to \ell$. Since $|\mathcal{L}_n| \approx n^p\mu(\mathcal{L}_n)$, for large n, we also have $|\mathcal{L}_n| \to \infty$. Assume $\bigcap_n \mathcal{L}_n = \mathcal{L}$. As $\mu(\mathcal{L}) = \ell > 0$, we have $\mathcal{L}_n \to \mathcal{L}$ which is non empty. Now consider a split along the direction $\boldsymbol{\alpha}$ and constant c, which splits $\mathcal{L}$ into $\mathcal{L}^L$ and $\mathcal{L}^R$. The same hyperplane also splits $\mathcal{L}_n$ in $\mathcal{L}_n^L$ and $\mathcal{L}_n^R$. Then, we must have $\mathcal{L}_n^L \to \mathcal{L}^L$ and $\mathcal{L}_n^R \to \mathcal{L}^R$.

As $\mathcal{L}_n$ are leaf nodes, we must have $\widehat{\Delta}(\mathcal{L}_n, \boldsymbol{\alpha}, c) \leq r_n$. Therefore,

$$\begin{aligned}
&\frac{1}{|\mathcal{L}_n|}\Big[\sum_{\boldsymbol{z}\in\mathcal{L}_n}\big(w(x,y)-\bar{w}\big)^2-\sum_{\boldsymbol{z}\in\mathcal{L}_n^R}\big(w(x,y)-\bar{w}_R\big)^2-\sum_{\boldsymbol{z}\in\mathcal{L}_n^L}\big(w(x,y)-\bar{w}_L\big)^2\Big]\leq r_n\\
&\implies \frac{|\mathcal{L}_n^L||\mathcal{L}_n^R|}{|\mathcal{L}_n|^2}\Big(\bar{w}_L-\bar{w}_R\Big)^2\leq r_n.\\
&\implies \frac{\mu(\mathcal{L}_n^L)\mu(\mathcal{L}_n^R)}{\mu(\mathcal{L}_n)^2}\Big(\bar{w}_L-\bar{w}_R\Big)^2\leq r_n.
\end{aligned}$$

Note that $\bar{w}_R$ and $\bar{w}_L$ are the averages of the right and left child, respectively. Now observe:

$$\begin{aligned}
\bar{w}_L &= \sum_{\boldsymbol{z}\in\mathcal{L}_n^L} w(\boldsymbol{z})/|\mathcal{L}_n^L|\\
&= \sum_{\boldsymbol{z}\in\mathcal{L}_n^L} (f(\boldsymbol{z})+\epsilon(\boldsymbol{z}))/|\mathcal{L}_n^L|\\
&= \frac{\frac{1}{n^p}\sum\limits_{\boldsymbol{z}\in\mathcal{L}_n^L} f(\boldsymbol{z})}{\frac{|\mathcal{L}_n^L|}{n^p}} + \frac{\sum\limits_{\boldsymbol{z}\in\mathcal{L}_n^L}\epsilon(\boldsymbol{z})}{|\mathcal{L}_n^L|}\\
&\to \Big(\int_{\mathcal{L}^L} f(\boldsymbol{z})d\mu\Big)/\mu(\mathcal{L}^L).
\end{aligned}$$

As the first term is the Riemann sum of the integral of $f(\boldsymbol{z})\mathbb{I}_{\mathcal{L}_n^L}(\boldsymbol{z})$, which is bounded by f, we can use Dominated Convergence Theorem (DCT) to get that limit. The second term converges to 0, due to the Strong Law of Large Numbers (SLLN). As $r_n \to 0$, combining the above two results, we have:

$$\frac{\mu(\mathcal{L}^L)\mu(\mathcal{L}^R)}{\mu(\mathcal{L})^2}\left(\frac{\int_{\mathcal{L}^L} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L}^L)}-\frac{\int_{\mathcal{L}^R} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L}^R)}\right)^2=0. \tag{3.6}$$

As $\mu(\mathcal{L})\neq 0$, and the choices of α and c were arbitrary, we note that:

$$\begin{aligned}
&\frac{\int_{\mathcal{L}^L} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L}^L)}=\frac{\int_{\mathcal{L}^R} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L}^R)}\\
\implies &\frac{\int_{\mathcal{L}^L} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L}^L)}=\frac{\int_{\mathcal{L}} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L})}\\
\implies &\frac{\int_{\boldsymbol{\alpha}^T z\leq c} f(\boldsymbol{z})d\mu}{\mu(\{\boldsymbol{\alpha}^T z\leq c\}\bigcap\mathcal{L})}=\frac{\int_{\mathcal{L}} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L})}
\end{aligned} \tag{3.7}$$

for all possible splits. Define $h(\boldsymbol{z}) = \frac{f(\boldsymbol{z})}{\int_{\mathcal{L}} f(\boldsymbol{z})d\mu}$. Consider a random variable $\boldsymbol{X}$ which takes values inside $\mathcal{L}$ with density $h(z)$. From Eqn. (3.7), we see that:

$$\int_{\boldsymbol{\alpha}^T z \leq c} h(\boldsymbol{z})d\mu = \frac{\mu(\{\boldsymbol{\alpha}^T z \leq c\} \bigcap \mathcal{L})}{\mu(\mathcal{L})} = \mathbb{P}(\boldsymbol{\alpha}^T \boldsymbol{X} \leq c).$$

Therefore, all linear combinations of $\boldsymbol{X}$ have the same distribution as linear combinations of a uniform random variable on $\mathcal{L}$. Hence, by Cramér-Wold theorem, we can say $\boldsymbol{X}$ has uniform distribution on $\mathcal{L}$, i.e., $h(z) = \frac{1}{\mu(\mathcal{L})}$ *a.e.* $\implies$ $f(\boldsymbol{z}) = \frac{\int_{\mathcal{L}} f(\boldsymbol{z})d\mu}{\mu(\mathcal{L})}$ *a.e.* Hence $f(\boldsymbol{z})$ is *a.e.* constant on $\mathcal{L}$.

Observation: As $\mathcal{L}_n$ are convex, so is $\mathcal{L}$. Therefore, if $\mu(\mathcal{L}) = 0$, then it is a convex set of dimension less than p.

Suppose, the dimension of $\mathcal{L}$ is $k < p$. Consider μ_k to be the Lebesgue measure corresponding to $\mathbb{R}^k$ on $\mathcal{L}$. Hence, we have $\mu_k(\mathcal{L}) \neq 0$.

Now as $\mathcal{L}$ is of dimension k, there must exist $(n-k)$ axes which are not parallel to the k-dimensional affine subspace $\bar{L}$ which contains $\mathcal{L}$. Call them $e_1, e_2, \ldots e_{n-k}$. Let $\mathcal{R}_{n-k}$ be the subspace generated by $e_1, e_2, \ldots e_{n-k}$. Now consider $\mathcal{R}_x = \boldsymbol{z_x} + \mathcal{R}_{n-k}$, where z_x is the closest lattice point to $\boldsymbol{x}$, and define a new function:

$$J_n(\boldsymbol{x}) = \frac{\sum\limits_{y \in \mathcal{R}_x \bigcap \mathcal{L}_n} w(\boldsymbol{y})}{|\mathcal{R}_x \bigcap \mathcal{L}_n|} \quad \forall \boldsymbol{x} \in \bar{L}.$$

We are basically superimposing the whole image onto $\bar{L}$. Now, if we define $\Delta_J(\mathcal{L}_n)$ in a similar fashion to Δ, it is easy to observe that

$$\Delta_J(\mathcal{L}_n) \leq \Delta(\mathcal{L}_n) \leq r_n.$$

Proceeding similarly to the above theorem, we get:

$$\frac{\mu_k(\bar{\mathcal{L}}_n^L)\mu_k(\bar{\mathcal{L}}_n^R)}{\mu(\bar{\mathcal{L}}_n)^2}\left(\bar{J}_n^L - \bar{L}_n^R\right)^2 \leq r_n,$$

where

$$\begin{aligned}\bar{J}_n^L &= \sum_{\mathbf{x}\in\bar{\mathcal{L}}_n^L} J_n(\boldsymbol{x})/|\bar{\mathcal{L}}_n^L| \\ &= \int_{\bar{\mathcal{L}}_n^L} J_n(\boldsymbol{x}) d\mu_k x/(|\bar{\mathcal{L}}_n^L|/n^k),\end{aligned}$$

and $\bar{J}_n^R$ is defined in a similar fashion. Observe that if f is continuous on $\mathcal{L}$, $J_n(\boldsymbol{x})$ converges point-wise to $f(x)$. Using the dominated convergence theorem (DCT) on the integral, we get:

$$\int_{\bar{\mathcal{L}}_n^L} J_n(\boldsymbol{x}) d\mu_k x \to \int_{\mathcal{L}^L} f(\boldsymbol{x}) d\mu_k x.$$

Then, proceeding similarly to the proof of Theorem 7, we get f is a.e. constant in $\mathcal{L}$. Remember that for small enough $\mu(\mathcal{L}_n)$, we assume only one JLC is present in $\mathcal{L}_n$, Hence, if f is not continuous on a measure set with respect to μ_k on $\mathcal{L}$, the same result holds. Otherwise, because of the assumption of a JLC being approximately a lower-dimensional plane, and $\mathcal{L}$ being convex, it must mean that the JLC contains $\mathcal{L}$ entirely or $\mathcal{L}$ contains the JLC entirely. Therefore, we have the following result:

Corollary 8. *If* $\lim_{n\to\infty} \mu(\mathcal{L}_n) = 0$ *and* $\bigcap_n \mathcal{L}_n = \mathcal{L}$ *lies on a* $k(< p)$ *dimensional affine subspace of* $\mathbb{R}^p$ *with* $\mu_k(\mathcal{L}) \neq 0$*, then one of these is true :*

(i) $\mathcal{L}$ *contains an entire JLC.*

(ii) $\mathcal{L}$ *is contained entirely inside a JLC.*

(iii) f *is a.e. constant on* $\mathcal{L}$*.*

Using $k = 1$ in the above corollary, we get:

Corollary 9. *Let* $\gamma(A)$ *denote the maximum distance between two points in the set* A*. If* $\lim_{n\to\infty} \mu(\mathcal{L}_n) = 0$ *and* $\lim_{n\to\infty} \gamma(\mathcal{L}_n) \neq 0$*, then one of the following is true:*

(i) $\bigcap_n \mathcal{L}_n$ *contains an entire JLC.*

(ii) $\bigcap_n \mathcal{L}_n$ *is contained entirely inside a JLC.*

(iii) f is a.e. constant on $\bigcap_n \mathcal{L}_n$.

Property 10. *If $\widehat{\Gamma} = \bigcup_x \left\{ \mathcal{L}^{\boldsymbol{x}} | \mathcal{L}^{\boldsymbol{x}}$ contains an entire JLC and $\mu(\mathcal{L}^{\boldsymbol{x}}) = 0 \right\}$, then $\mu(\widehat{\Gamma}) = 0$.*

The above property holds true because at most countably many sets of unions are distinctly non-empty, as there are at most countably many JLCs.

Next, we show that the average of all continuity points of f over a leaf node converges to the true value. Observe that if $\boldsymbol{x} \in \mathcal{L}_n$ is the point of interest and there is no JLC in $\mathcal{L}_n$, then as $\mu(\mathcal{L}_n) \to 0$, $\gamma(\mathcal{L}_n) \to 0$, and f is piecewise Lipschitz, the average of the observed intensity values over the leaf node should converge to $f(\boldsymbol{x})$. However, if $\mathcal{L}_n$ contains a JLC, we need to show that all the points on the leaf node are similar, and do not have any jumps between them, which brings us to the next result.

Lemma 11. *Let $\boldsymbol{x}$ be a non-singular continuity point of f. If $\lim_{n\to\infty} \gamma(\mathcal{L}_n^{\boldsymbol{x}}) = 0$, then $\exists\ N$ such that $\forall\ m > N$, $\mathcal{L}_m^{\boldsymbol{x}}$ does not contain any discontinuity point.*

Proof: Suppose that for all n, $\mathcal{L}_n^{\boldsymbol{x}}$ contains at least one discontinuity point. Consider $\mathcal{N}_\epsilon(\boldsymbol{x})$ to be an ϵ ball around $\boldsymbol{x}$. As $\gamma(\mathcal{L}_n^{\boldsymbol{x}}) \to 0$, we can say that for every ϵ, there exists an $n_\epsilon > 0$ such that $\forall\, m > n_\epsilon$, $\gamma(\mathcal{L}_n^{\boldsymbol{x}}) < \epsilon$, which implies $\mathcal{L}_m^{\boldsymbol{x}} \subseteq \mathcal{N}_\epsilon(\boldsymbol{x})$. However, as $\mathcal{L}_m^{\boldsymbol{x}}$ contains at least one discontinuity point, say $\boldsymbol{y_n}$. Since $\{\boldsymbol{y_n}\}$ is a bounded sequence, there exists a convergent subsequence. Assume that the convergent subsequence converges to $\boldsymbol{y}$. As $\boldsymbol{y} \in \overline{\bigcap_n \mathcal{L}_n^{\boldsymbol{x}}}$, we have $\|\boldsymbol{y} - \boldsymbol{x}\| = 0 \implies \boldsymbol{y} = \boldsymbol{x}$. Since the set of jump points is closed by definition, either $\boldsymbol{y}$ is a discontinuity point or a singular point. Both options contradict our assumption that $\boldsymbol{x}$ is a non-singular continuity point. Therefore, $\mathcal{L}_n^{\boldsymbol{x}}$ cannot contain a discontinuity point $\forall\ n$. Since $\mathcal{L}_n^{\boldsymbol{x}}$ is a non-increasing sequence of sets, we conclude that after some N, it does not contain any discontinuity point.

Theorem 12. $\lim_{n\to\infty} \widehat{f}_n(\boldsymbol{x}) = f(\boldsymbol{x})$ *a.e., almost surely.*

Proof: We prove this only on continuity points for which $\mathcal{L}^{\boldsymbol{x}}$ *does not contain an entire JLC* as the remaining points have measure zero. We divide this proof into three different cases:

Case 1: f is not *a.e.* constant in $\bigcap_n \mathcal{L}_n^{\boldsymbol{x}}$. Hence we also have $\lim_{n\to\infty} \mu(\mathcal{L}_n^{\boldsymbol{x}}) = 0$.
As per the assumptions, $\bigcap_n \mathcal{L}_n^{\boldsymbol{x}}$ cannot lie entirely on a JLC. Hence, we have $\lim_{n\to\infty} \gamma(\mathcal{L}_n^{\boldsymbol{x}}) = 0$.

Using Lemma 11, $\exists\ N$ such that $\mathcal{L}_n^{\boldsymbol{x}}$ does not contain any JLC $\forall\ n > N$. In this case, $\forall \boldsymbol{z} \in \mathcal{L}_n^{\boldsymbol{x}}$, we have:

$$
\begin{aligned}
& |w(\boldsymbol{z}) - f(\boldsymbol{x})| \leq |f(\boldsymbol{z}) - f(\boldsymbol{x})| + |\varepsilon(\boldsymbol{z})| \leq C_\ell \gamma((\mathcal{L}_n^{\boldsymbol{x}})) + |\varepsilon(\boldsymbol{z})| \\
\implies & \left| \sum_{z \in \mathcal{L}_n^{\boldsymbol{x}}} w(\boldsymbol{z}) - f(x) \right| \leq |\mathcal{L}_n^{\boldsymbol{x}}| C_\ell \gamma(\mathcal{L}_n^{\boldsymbol{x}}) + \left| \sum_{z \in \mathcal{L}_n^{\boldsymbol{x}}} \varepsilon(\boldsymbol{z}) \right| \\
\implies & \left| \widehat{f}_n(\boldsymbol{x}) - f(x) \right| \leq C_\ell \gamma(\mathcal{L}_n^{\boldsymbol{x}}) + \left| \frac{\sum\limits_{z \in \mathcal{L}_n^{\boldsymbol{x}}} \varepsilon(\boldsymbol{z})}{|\mathcal{L}_N^{\boldsymbol{x}}|} \right| \qquad (3.8) \\
\implies & \left| \widehat{f}_n(\boldsymbol{x}) - f(x) \right| \xrightarrow{n \to \infty} 0, \text{ using SLLN and Corollary 8.}
\end{aligned}
$$

Case 2: f is *a.e.* constant in $\bigcap\limits_n \mathcal{L}_n^{\boldsymbol{x}}$ and $\lim\limits_{n \to \infty} \mu(\mathcal{L}_n^{\boldsymbol{x}}) \neq 0$.

In this case, we know that f is *a.e.* constant in $\bigcap\limits_n \mathcal{L}_n^{\boldsymbol{x}}$. So we have:

$$
\begin{aligned}
\left| \widehat{f}_n(\boldsymbol{x}) - f(x) \right| &= \frac{\sum\limits_{z \in \mathcal{L}_n^{\boldsymbol{x}}} (f(\boldsymbol{z}) - f(\boldsymbol{x}))}{|\mathcal{L}_n^{\boldsymbol{x}}|} + \frac{\sum\limits_{z \in \mathcal{L}_n^{\boldsymbol{x}}} \varepsilon(\boldsymbol{z})}{|\mathcal{L}_n^{\boldsymbol{x}}|} \\
&= \frac{\frac{1}{n^p} \sum\limits_{z \in \mathcal{L}_n^{\boldsymbol{x}}} (f(\boldsymbol{z}) - f(\boldsymbol{x}))}{|\mathcal{L}_n^{\boldsymbol{x}}|/n^p} + \frac{\sum\limits_{z \in \mathcal{L}_n^{\boldsymbol{x}}} \varepsilon(\boldsymbol{z})}{|\mathcal{L}_n^{\boldsymbol{x}}|}.
\end{aligned}
$$

The numerator of the first term converges to $\int\limits_{\bigcap\limits_n \mathcal{L}_n^{\boldsymbol{x}}} (f(\boldsymbol{z}) - f(\boldsymbol{x}))$ which is 0 as f is almost surely *a.e.* constant there, and the denominator converges to $\lim\limits_{n \to \infty} \mu(\mathcal{L}_n^{\boldsymbol{x}})$ which is non-zero. Therefore, we have $\lim\limits_{n \to \infty} \widehat{f}_n(\boldsymbol{x}) = f(\boldsymbol{x})$ almost surely.

Case 3: f is *a.e.* constant on $\bigcap\limits_n \mathcal{L}_n^{\boldsymbol{x}}$ and $\lim\limits_{n \to \infty} \mu(\mathcal{L}_n^{\boldsymbol{x}}) = 0$.
Observe that if $\exists\ k$ such that $\lim\limits_{n \to \infty} \mu_k(\mathcal{L}_n^{\boldsymbol{x}}) \neq 0$, we can proceed similar to Case 2. Therefore, without any loss of generality, we can assume $\gamma(\mathcal{L}_n^{\boldsymbol{x}}) \to 0$, and then proceed similarly to Case 1.

From above three cases, we conclude that $\lim\limits_{n \to \infty} \widehat{f}_n(\boldsymbol{x}) = f(\boldsymbol{x})$ *a.e.*, almost surely.

3.4 An Application of the Proposed Estimation Method: Image Denoising

Image denoising is a fundamental problem in the field of image processing. It serves as a crucial pre-processing step in many applications to make further analysis meaningful and reliable. For example, in sequential image surveillance, denoising plays a central role. In addition to removing noise, a key requirement for effective denoising methods is the preservation of edge structures within the image. In the JRA literature [Qiu (2005)], a 2D grayscale image is expressed as a discontinuous regression surface, where the edges of objects in the image are treated as points of discontinuity or "jump points." In this section, we introduce a new image denoising method based on an oblique-axis regression tree, designed to preserve these jumps in the regression surface. Although the algorithm is versatile and applicable to various types of images, we focus on 2D grayscale images here for simplicity. The proposed 2D grayscale image denoising technique directly applies the algorithm from Section 2 with $p = 2$. Consequently, the 2D JRA model can be described similarly as follows:

$$w_{ij} = f(x_i, y_j) + \varepsilon_{ij}\ , \quad \text{for} \quad i, j = 1, 2, \ldots, n, \tag{3.9}$$

where $\mathcal{D}_n = \{(x_i, y_j) : i, j = 1, 2, ..., n\}$ are equally spaced design points (or pixels) in the design space $\Omega = [0, 1] \times [0, 1]$, $f(x, y)$ is the unknown image intensity function at (x, y), and ε_{ij} are independent and identically distributed (i.i.d.) random errors with mean 0 and variance $\sigma^2 > 0$. To denoise an image, we perform recursive tree splitting until all nodes become leaf nodes, for a specified cut-off r_n depending on n. Using the tree partitioning method outlined in Algorithm 1, we partition the design space $[0, 1]^2$ in K_n number of convex polygons, represented as $\{\mathcal{L}_n^i | 1 \leq i \leq K_n\}$. The resulting estimator is obtained by averaging the image intensity values within the leaf nodes. However, in practice, large leaf nodes may cause the loss of intricate details in the image surface. To address this issue, we consider *leaf-wise local* averaging instead of averaging over the whole leaf node. For this adjustment, we consider a square neighborhood at each pixel (x, y) as $B(x, y; h_n)$ where $2h_n > 0$ is the length of each side of the square. The modified estimator at $f(x, y)$

is thus defined as:

$$\widehat{f}_n^M(x,y) = \frac{\sum\limits_{(u_i,v_j)\in\mathcal{L}_n^{(x,y)}\bigcap B(x,y;h_n)} w(u_i,v_j)SS_{(x,y)}(u_i,v_j)}{\sum\limits_{(u_i,v_j)\in\mathcal{L}_n^{(x,y)}\bigcap B(x,y;h_n)} SS_{(x,y)}(u_i,v_j)}. \tag{3.10}$$

Here, $\mathcal{L}_n^{(x,y)}$ represents the leaf node containing the test pixel coordinate (x,y), (u_i,v_j) denotes pixel coordinates within $\mathcal{L}_n^{(x,y)}$, and $SS_{(x,y)}(u_i,v_j)$ is a similarity score between the pixels (x,y) and (u_i,v_j) expressed as follows:

$$SS_{(x,y)}(u_i,v_j) = \exp\left(-\frac{\kappa_n||B(u_i,v_j;h_n) - B(x,y;h_n)||^2}{(nh_n)^2}\right).$$

In the above expression, the distance between the neighborhoods represents the L_2-distance between the pixel intensities. Regarding the choice of h_n, selecting a value too large could blur fine details, while a very small value may result in a noisy estimator $\widehat{f}_n^M(x,y)$. Based on numerical experience, we recommend choosing $h_n \in [\frac{2}{n}, \frac{4}{n}]$. Figure 3.2 demonstrates that the proposed method can partition the pixels of a given image appropriately, and thus can remove noise while preserving the edge details.

Figure 3.2: Demonstration of partitioning of the pixels in the test image. Images from left to right are the noisy image, the denoised image by the proposed method, and the estimated partitions, respectively. The partitions are represented by different shades. Note that the estimated partitions are subsets of the true partitions.

3.5 Numerical Studies

In this section, we evaluate the performance of the proposed method through numerical analysis, comparing it with several state-of-the-art methods from the literature. Since jump preserving surface estimation is common in image analysis, we conduct our numerical experiments with various simulated images and real images. Including the noise removal, another major aspect in the context of image denoising is edge preservation. We demonstrate the performance of the proposed method both in terms of noise removal and edge preservation. To evaluate the noise removal performance of the proposed method in comparison with the state-of-the-art methods, we consider root-expected mean square error (REMSE), defined as

$$REMSE(\widehat{f}, f) = \sqrt{\mathbb{E}\Big[\sum_{(u,v)\in\Omega}\Big(\widehat{f}(u,v) - f(u,v)\Big)^2\Big]}.$$

On the other hand, the performance of edge preserving can be quantified by a similarity measure between the sets of detected edge pixels of the denoised image and the true image. One such similarity measurement between the denoised image and the true image can be expressed as

$$d_{KQ}(\widehat{\Gamma}_{\widehat{f}}, \widehat{\Gamma}_f) = \frac{0.5}{|\widehat{\Gamma}_{\widehat{f}}|}\sum_{(u,v)\in\widehat{\Gamma}_{\widehat{f}}} d_E((u,v), \widehat{\Gamma}_f) + \frac{0.5}{|\widehat{\Gamma}_f|}\sum_{(u,v)\in\widehat{\Gamma}_f} d_E((u,v), \Gamma_{\widehat{f}}),$$

where $\widehat{\Gamma}_{\widehat{f}}$ and $\widehat{\Gamma}_{\widehat{f}}$ are the detected edge pixels for the denoised image and the true image, respectively. It is to be noted that d_{KQ} does not hold the properties of a metric. One popular metric to compute the distance between two point sets is *Hausdorff* distance. However, *Hausdorff* metric is very sensitive to individual points. Therefore, we select d_{KQ} to compute the similarity measurement between the point-sets of the detected edge pixels of the two images. For a good image denoising procedure, it is expected that both the values of REMSE and d_{KQ} would be small.

3.5.1 A brief description of the competing methods

To evaluate the numerical performance of the proposed method in terms of REMSE and jump preservation, we select the following three popular state-of-the-art methods: (i) image denoising using usual random forest technique Breiman (2001), (ii) edge preserving image denoising method proposed by Qiu (2009), and (iii) image denoising method by non-local means proposed by Buades et al. (2011).

(i) Image denoising using random forest (RF): Random forest is the most popular tree-based ensemble learning algorithm in the context of nonparametric regression. Since it is an ensemble method, it is capable of accommodating the complex shapes in the data. One of the major advantages of the random forest over a single tree-based method (CART) is that it can avoid the problem of overfitting, unlike CART. Therefore, random forest could be used as a potential image denoising method. However, in the case of image smoothing, to accommodate the complex structure of the image object, random forest splits more, leading to deeper trees, which aggravates the problem of overfitting. The proposed method also involves a decision tree, and therefore it is comparable with random forest-based methods.

(ii) Jump preserving local linear kernel smoothing (JPLLK): The method first divides the local neighborhood into two parts using gradient information. Then, three different estimates are calculated using local linear kernel regression using the left, right, and whole neighborhood, respectively. The best of these three locally fitted surfaces is chosen, depending on their mean squared error. Then, we follow the same procedure on the fitted surface again. But this time, one of those three estimates is chosen as the final estimate based on their estimated variance. This method is very good at preserving jumps, although its image denoising performance falls short when compared to our proposed method.

(iii) Image denoising based on non-local means (NLM): In this method, the true image intensity value at a given pixel is estimated using a weighted average on a large portion of the entire image. To calculate the weights, each pixel is scored according to its similarity with the given pixel. Then, these scores are used as weights for the estimation. For convenience, we choose an implementation proposed by Froment (2014), as it auto-

matically calculates the parameters for the denoising method. Among these competing methods, the non-local means method usually performs at best at image denoising in the sense of visual appearances, at the cost of blurring the image.

3.5.2 Simulations

In the simulation study, we consider a triangle image as the test image. See the extreme left of the upper panel in Figure 3.3 for the true triangle image. The image intensity function of the test image can be expressed as

$$f(u, v) = \mathbb{I}[(u, v) \in \Omega_1], \quad (u, v) \in [0, 1] \times [0, 1], \tag{3.11}$$

where $\mathbb{I}$ is the indicator function, and Ω_1 denotes the region enclosed by the triangle. Throughout this section, we use Gaussian additive noise to generate the noisy image surface. See the 3-rd image from the left of the upper panel in Figure 3.3 for the noisy test image.

We carry out the simulation study under the following settings: (i) to demonstrate the noise removal property, we use three different noise levels with $\sigma = 0.1, 0.2,$ and 0.3. (ii) to demonstrate the consistency of the proposed estimation method, we perform further simulations with two different image resolutions: 200×200, i.e., $n = 200$, and 400×400, i.e., $n = 400$. Table 3.1 shows the REMSE values for various cases and methods. Moreover, the summary of the performances of the concerned methods with respect to edge preservation is provided in Table 3.2. Based on 100 replications, the results are shown in Table 3.1.

		Proposed Algorithm	Competing Methods		
Image Resolution	Noise Levels	OLT	JPLLK	NLM	RF
200×200	0.3	33.8 (4.697)	54.1 (1.196)	51.1 (1.559)	108.3 (1.095)
	0.2	25.5 (4.237)	40.0 (0.659)	29.6 (1.136)	86.5 (0.728)
	0.1	18.1 (4.752)	27.4 (0.382)	14.4 (0.511)	70.6 (0.717)
400×400	0.3	22.3 (4.024)	49.5 (0.539)	40.0 (0.821)	91.6 (0.852)
	0.2	15.4 (3.822)	33.9 (0.318)	24.9 (0.584)	69.6 (0.522)
	0.1	7.9 (3.349)	19.8 (0.203)	12.5 (0.255)	53.0 (0.470)

Table 3.1: Comparisons of various methods on the simulated test image using (REMSE$\times 10^3$) values based on 100 independent replications with standard error within the parentheses. Note that the contrasts of the true images are 1.

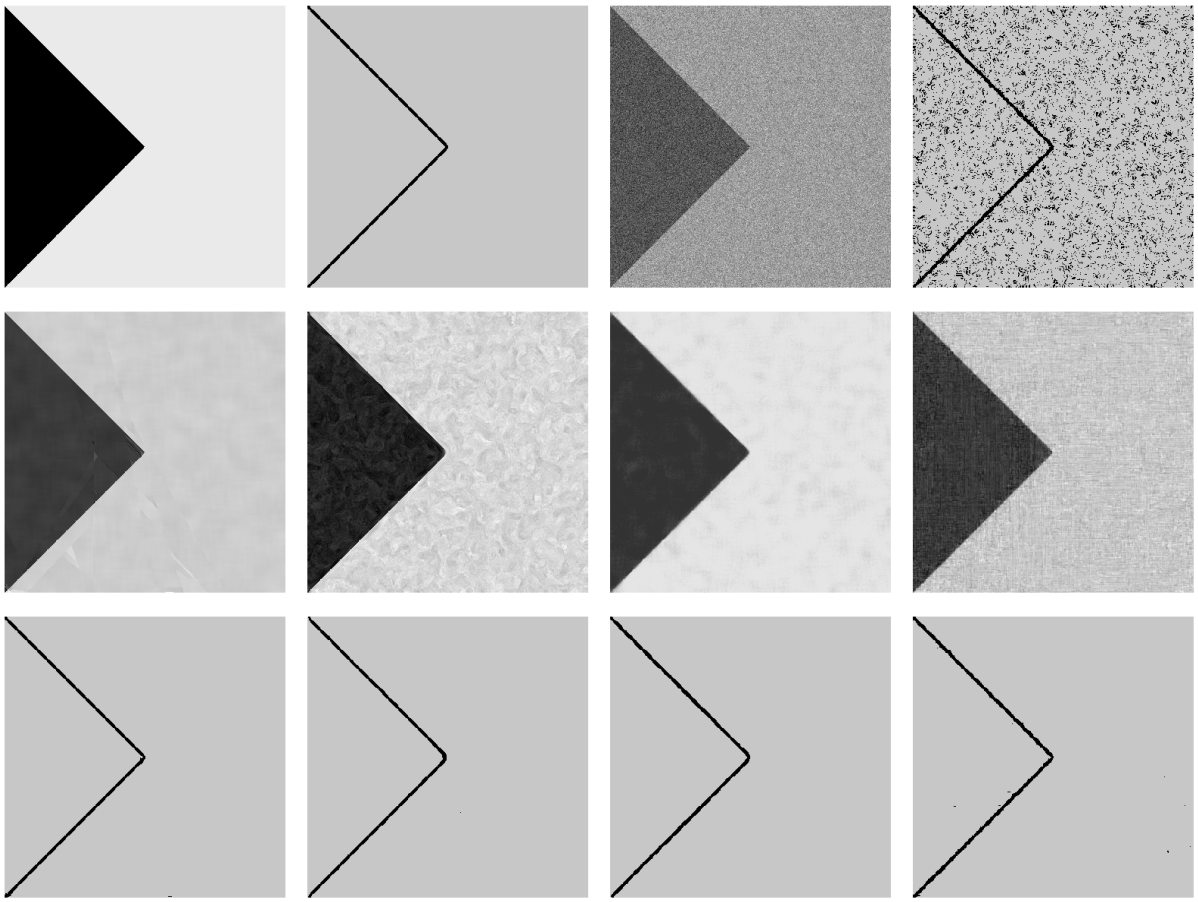


Figure 3.3: Performance comparison on the simulated test image with point-wise Gaussian noise with $\sigma = 0.3$. The first row shows the original image, original edges, noisy image, and detected edges on the noisy image, respectively. The second row shows denoised images produced by the proposed method, JPLLK, NLM, and RF. The third row shows the detected edges of the corresponding denoised images.

From Tables 3.1 and 3.2, we can see that the proposed ORT-based method performs better when $n = 400$ rather than 200, in terms of both REMSE and edge preservation. As the proposed method is asymptotically consistent, these results are intuitively reasonable. The proposed method is uniformly better than RF and JPLLK both in noise removal and edge preservation. Except for the case when $n = 200$ and $\sigma = 0.1$, the proposed method outperforms NLM in all other cases. The denoised images are shown in the middle panel of Figure 3.3. From the visual impression, it appears that the denoising performances of JPLLK and RF are relatively poor compared to the proposed method. NLM performs relatively better; however, several noisy patchy regions are present in the denoised image. We demonstrate the edge detection performances in the lower panel of Figure 3.3. From this simulation study, we see that the proposed method outperforms

		Proposed Algorithm	Competing Methods		
Image Resolution	Noise Levels	ORT	JPLLK	NLM	RF
200×200	0.3	0.427	0.649	0.722	2.059
	0.2	0.115	0.265	0.151	0.470
	0.1	0.110	0.266	0.000	0.391
400×400	0.3	0.086	0.570	0.386	1.67
	0.2	0.086	0.087	0.059	0.275
	0.1	0.005	0.068	0.003	0.177

Table 3.2: Comparisons of various methods regarding jump preservation on the simulated test image using $(d_{KQ} \times 10^3)$.

the competing methods in almost all scenarios considered above.

3.5.3 Choice of hyperparameter r_n

In this subsection, we discuss the effect of the choice of r_n on denoising performance. Decreasing its value can lead to finer partitioning and potentially better local adaptation; however, it may also introduce artifacts in the denoised images, as the resulting leaf nodes become too small. On the other hand, larger values of r_n promote smoothing over broader regions, which can reduce noise effectively but may lead to oversmoothing and loss of important image features. From Table 3.3, we see that the performance of the proposed method remains stable on a wide range of r_n/σ^2 values unless it is chosen very small.

$100r_n/\sigma^2$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
0.01	0.0505 (0.0024)	0.0787 (0.0060)	0.1082 (0.0081)
0.1	0.0349 (0.0015)	0.0464 (0.0056)	0.0549 (0.0094)
1	0.0241 (0.0015)	0.0323 (0.0003)	0.0423 (0.0002)
10	0.0224 (0.0003)	0.0323 (0.0002)	0.0423 (0.0002)

Table 3.3: Mean REMSE with standard deviation (in parentheses) for different values of $100r_n/\sigma^2$ and noise levels σ for the plane image.

3.5.4 Denoising performance on images of low resolution

The following table demonstrates the performance of our algorithm on the simulated triangle image in lower resolutions. We observe that both the average and standard deviations of the estimated REMSEs are considerably higher than those reported for higher-resolution images. This can be attributed to the formation of smaller leaf nodes,

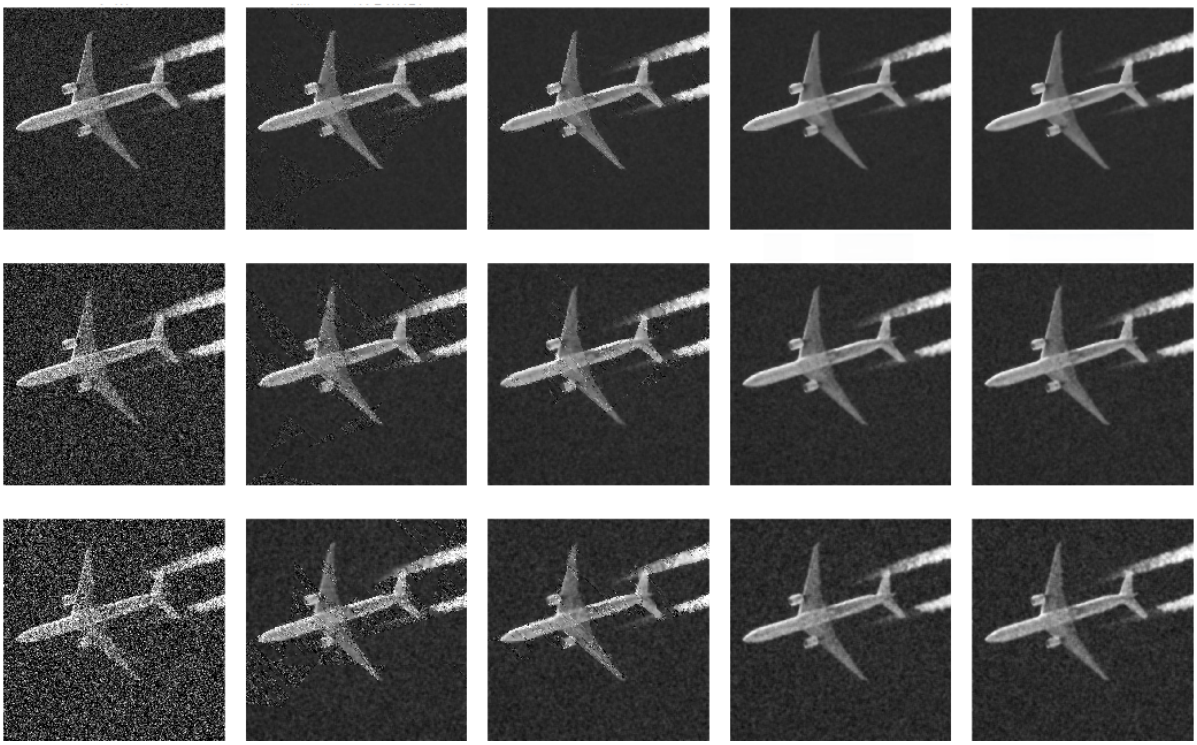


Figure 3.4: Denoising performance with various values of the hyperparameter r_n for the plane image.

as well as convergence difficulties when determining optimal splits in lower-resolution settings.

Image Size	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
32×32	0.086 ± 0.0163	0.160 ± 0.019	0.245 ± 0.021
64×64	0.049 ± 0.010	0.100 ± 0.014	0.181 ± 0.025
128×128	0.024 ± 0.006	0.053 ± 0.008	0.099 ± 0.014

Table 3.4: Denoising performance of the proposed ORT-based method for various image sizes using the triangle image. The values in the table indicate average REMSE values over 500 independent replications. Note that the contrasts of the true images are 1.

3.5.5 Applications on real images

We apply the proposed ORT-based method on complicated real images and perform a comparative analysis with the state-of-the-art competing methods. In this regard, we consider two different real images. The image in the extreme left of the upper panel of Figure 3.5 shows high-rise buildings with resolution 523×523. The image in the extreme

left of the upper panel of Figure 3.6 shows a moving aero-plane with resolution 628×628.

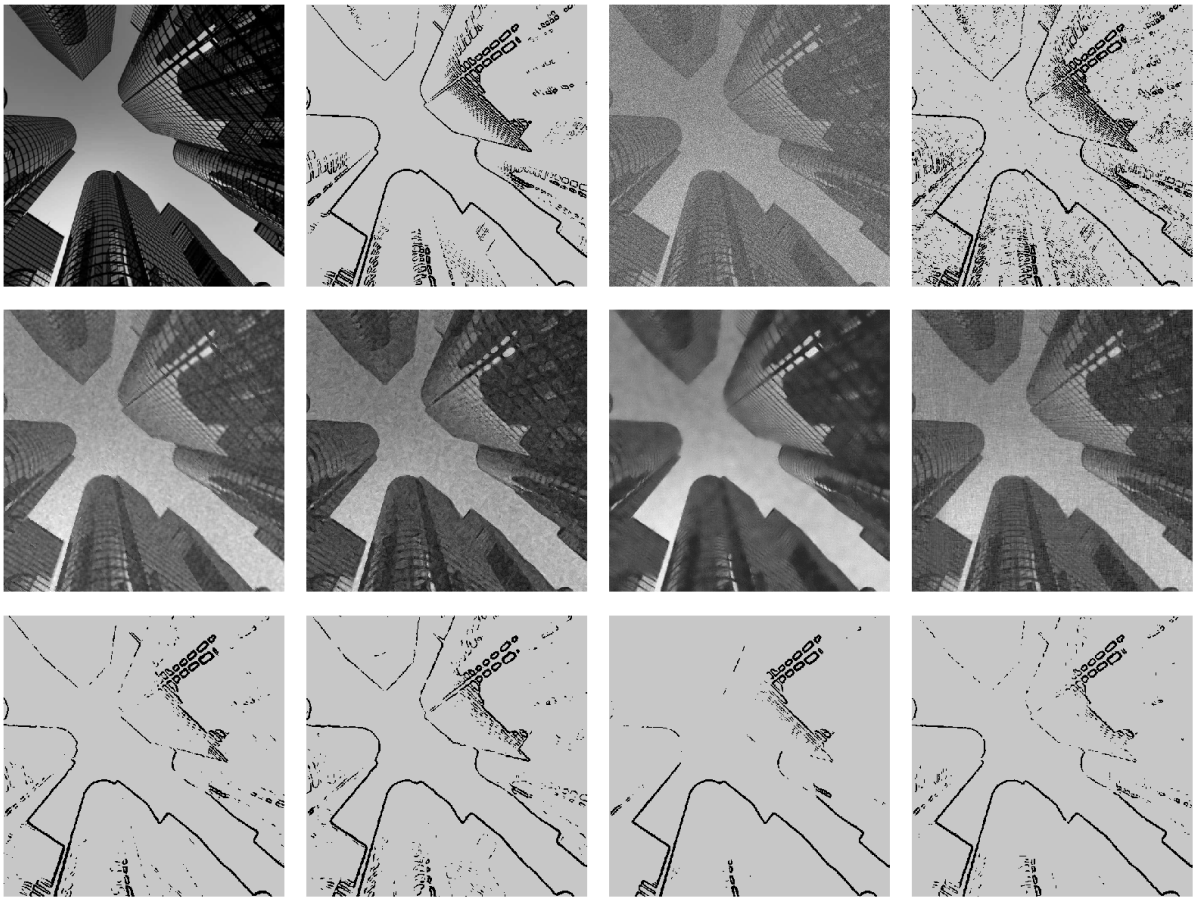


Figure 3.5: Performance comparison on the building image with point-wise Gaussian noise with $\sigma = 0.2$. The first row shows the original image, original edges, noisy image, and detected edges on the noisy image, respectively. The second row shows denoised images produced by the proposed method, JPLLK, NLM, and RF. The third row shows the detected edges of the corresponding denoised images.

The numerical performances of the concerned methods regarding noise removal and edge preservation are provided in Tables 3.5 and 3.6. From Table 3.5, we see that for the building image, the proposed method uniformly outperforms JPLLK and RF in terms of REMSE. In this regard, the NLM method is comparatively better than the proposed method except for larger noise levels, when its performance is similar to the proposed method. In terms of edge preservation, the proposed method is much better than NLM in most cases. Although edge-preservation measures are good for RF and JPLLK, their performance on noise removal is rather poor, especially when the noise level is high.

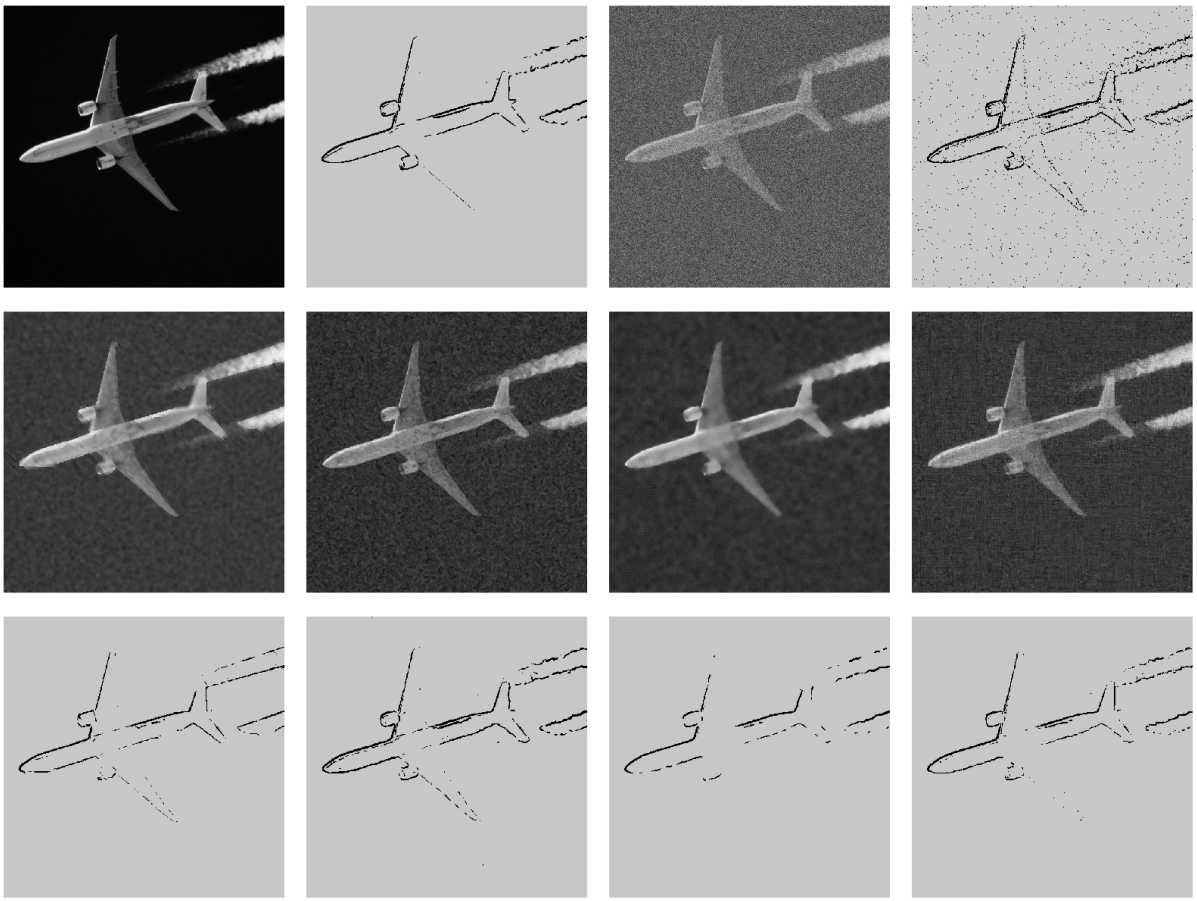


Figure 3.6: Performance comparison on the aero-plane image with point-wise Gaussian noise with $\sigma = 0.2$. The first row shows the original image, original edges, noisy image, and detected edges on the noisy image, respectively. The second row shows denoised images produced by the proposed method, JPLLK, NLM, and RF. The third row shows the detected edges of the corresponding denoised images.

In summary, we can conclude that the performance of the proposed method shows a reasonable trade-off between noise removal and edge preservation abilities. The proposed method has good image denoising capability along with a good edge preservation property.

3.5.6 Comparison with a state-of-the-art deep learning model

We now compare the proposed algorithm with one of the best-performing state-of-the-art deep learning-based models. Note that this is not an apple-to-apple comparison, as the deep learning-based models are trained on multiple images of various types, while the proposed method is for a single image only. Table 3.7 shows the performance of the proposed algorithm when compared to the pre-trained model *Restormer* under the

		Proposed Algorithm	Competing Methods		
Image Type	Noise Levels	ORT	JPLLK	NLM	RF
Building	0.3	95.8 (0.873)	102.2 (0.324)	96.2 (0.343)	106.1 (0.497)
	0.2	82.1 (0.402)	100.2 (0.221)	77.0 (0.248)	84.8 (0.415)
	0.1	58.8 (3.22)	80.3 (0.153)	48.6 (0.142)	64.1 (0.269)
Plane	0.3	45.4 (0.741)	89.0 (0.505)	44.6 (0.301)	82.1 (0.829)
	0.2	37.4 (0.621)	60.9 (0.183)	33.2 (0.191)	56.1 (0.507)
	0.1	27.7 (0.704)	33.9 (0.099)	16.7 (0.082)	34.8 (0.299)

Table 3.5: Comparisons of various methods on the real images using (REMSE$\times 10^3$) values based on 100 independent replications with standard error within the parentheses. Note that the contrasts of the true images are 1.

		Proposed Algorithm	Competing Methods		
Image Type	Noise Levels	ORT	JPLLK	NLM	RF
Building	0.3	13.2	4.14	65.16	8.53
	0.2	6.86	4.14	25.26	6.67
	0.1	1.59	1.37	2.38	5.63
Plane	0.3	8.34	15.25	31.45	3.46
	0.2	4.56	2.33	6.51	1.67
	0.1	1.63	1.10	0.75	2.29

Table 3.6: Comparisons of various methods regarding jump preservation on real images using ($d_{KQ} \times 10^3$).

same setup as before. This is reasonable because the images on which *Restormer* was originally trained are possibly different from the test images we apply to. From Table

		Proposed Method	Competing Method
Image	Noise	ORT	Restormer
Building	0.3	95.8 (0.873)	165.7(0.5)
	0.2	82.1 (0.402)	100.2(0.3)
	0.1	58.8 (3.22)	52.7(0.1)
Plane	0.3	45.4 (0.741)	154.1(0.4)
	0.2	37.4 (0.621)	75.2(0.4)
	0.1	27.7 (0.704)	35.4(0.1)

Table 3.7: Comparisons with *Restormer* on the real images using (REMSE$\times 10^3$) values based on 100 independent replications with standard error within the parenthesis.

3.7, we can see that the proposed algorithm works much better in high noise scenarios when compared to *Restormer*. Note that the performance of *Restormer* may be affected by the high noise used in the demonstrated cases, and it may not have been trained using noise of this magnitude. For implementation of *Restormer*, we utilize the official codes and pre-trained weights provided by Zamir et al. (2022).

3.5.7 Performance comparison on a large-scale dataset

We further evaluate the proposed method using the SIDD+NTIRE challenge dataset (Abdelhamed et al., 2020). For this analysis, the validation dataset is employed, as it provides publicly available ground-truth images. The dataset comprises 32 base images, each with 32 cropped variants, resulting in a total of 1024 images with a spatial resolution of 256×256. In the NTIRE challenge, 22 participating teams submitted 24 distinct algorithms, all trained on the validation data and subsequently assessed on the test set. Our proposed algorithm attains a peak signal-to-noise ratio, i.e., PSNR of 33.291 on the sRGB benchmark. This result is noteworthy, as it situates the proposed algorithm within the competitive range of the leading methods, including some of the best-performing deep learning approaches reported in the original challenge. Considering that the proposed method does not rely on training with large datasets, it also demonstrates broader applicability and potential for generalization.

3.6 Concluding Remarks

In this chapter, we propose a framework using Oblique-axis Regression Tree (ORT) to estimate the underlying discontinuous function on a lattice and show its point-wise convergence under only a few mild assumptions. We also show that similar points are often grouped in the same leaf node of the decision tree we obtain from the proposed algorithm, which is very desirable since it facilitates further analysis. Then, we proceed to apply this algorithm to noisy images for removing noise, and show its superior performance compared to many other state-of-the-art methods. However, there are scopes for improvement. In this chapter, we only show that the algorithm converges when the number of lattice points goes to infinity. Instead, one can consider a different approach and keep the number of lattice points fixed while letting the number of samples go to infinity. In this way, one can extend the use of the proposed algorithm to various real-life problems, such as image monitoring. Instead of working with equally spaced design points, one can expand its capability to work with general tabular data, where the design points are not necessarily evenly distributed. Another future direction of research is to generate multiple sample images using bootstrapping techniques and build a random forest using the proposed al-

gorithm to facilitate better estimates. Apart from these, the proposed algorithm can also be applied to other problems, e.g., image segmentation, denoising of an image sequence, and so on.

Chapter 4

Monitoring of Drift Patterns in Image Data using Oblique-axis Regression Tree

4.1 Introduction

With the widespread availability of advanced image acquisition technologies, large volumes of image data are now routinely collected across various scientific domains. Consequently, images have become a commonly used data format in numerous sectors. A prominent example of such a data source is the Landsat project, jointly led by the U.S. Geological Survey (USGS) and NASA. Since its inception in 1972, this program has launched eight satellites to continuously capture scientifically valuable images of the Earth's surface. The resulting 53-year archive of Landsat images has emerged as a crucial resource for a wide range of scientific disciplines, including forest science, climate science, agricultural forecasting, ecological and ecosystem monitoring, water resource management, biodiversity conservation, and many others. The current Landsat satellite is capable of capturing images of the same geographic region approximately every 16 days. This regular acquisition makes it essential to assess whether the observed images are temporally stable, that is, to sequentially monitor the images for potential temporal changes. Importantly, the need for such monitoring is not limited to satellite imagery; it also extends to various domains

such as industrial quality control and medical diagnostics. Sequential image monitoring poses significant challenges, primarily due to the complex structure of the image intensity function, which often includes discontinuities. Moreover, changes in a sequence of images tend to occur gradually over time, a phenomenon referred to as *drift*. In many applications, the goal extends beyond detecting abrupt changes or step shifts to the more complex task of monitoring and characterizing drift patterns. In the context of change-point detection, modeling drift in sequential image data is considerably more challenging. This chapter specifically addresses this issue by proposing a novel method for monitoring drift patterns in sequential images. Despite its practical importance, this problem remains largely underexplored in the existing image monitoring literature. To fill this gap, we introduce an algorithm based on decision trees, more precisely, an oblique-axis regression tree (ORT). The rationale for adopting a decision tree-based approach lies in its capacity to effectively model discontinuities in both the spatial and temporal dimensions. The ORT framework offers the flexibility needed to capture complex edge structures in image intensity functions and to accurately monitor gradual temporal changes and their patterns.

4.1.1 Data description and motivation

The methodology developed in this chapter is motivated by a sequence of satellite images capturing the Aral Sea region. The Aral Sea, formerly the third-largest lake in the world, is an endorheic lake situated between Kazakhstan and Uzbekistan. Over the past several decades, it has undergone dramatic shrinkage primarily due to unsustainable irrigation practices. This decline has resulted in severe environmental and socio-economic consequences, including increased salinity, frequent dust storms, and significant alterations in the regional climate. It is widely regarded as one of the most devastating environmental crises globally. In response, several initiatives have been undertaken by Kazakhstan and Uzbekistan to slow down the rate of shrinkage. The sequence of satellite images of the Aral Sea presents a compelling dataset for evaluating potential changes in the rate or pattern of this shrinkage over time. In addition to introducing the proposed methodology, this chapter applies the algorithm to analyze satellite images of the Aral Sea from 2013 onward. Beyond this specific case, the issue of water body shrinkage is of broader concern:

rapid urbanization, increasing water demand, and high evaporation rates, particularly in arid and semi-arid regions, have led to the gradual depletion of several other lakes and reservoirs worldwide. In such contexts, the proposed algorithm offers a valuable tool for monitoring drift patterns and long-term environmental change.

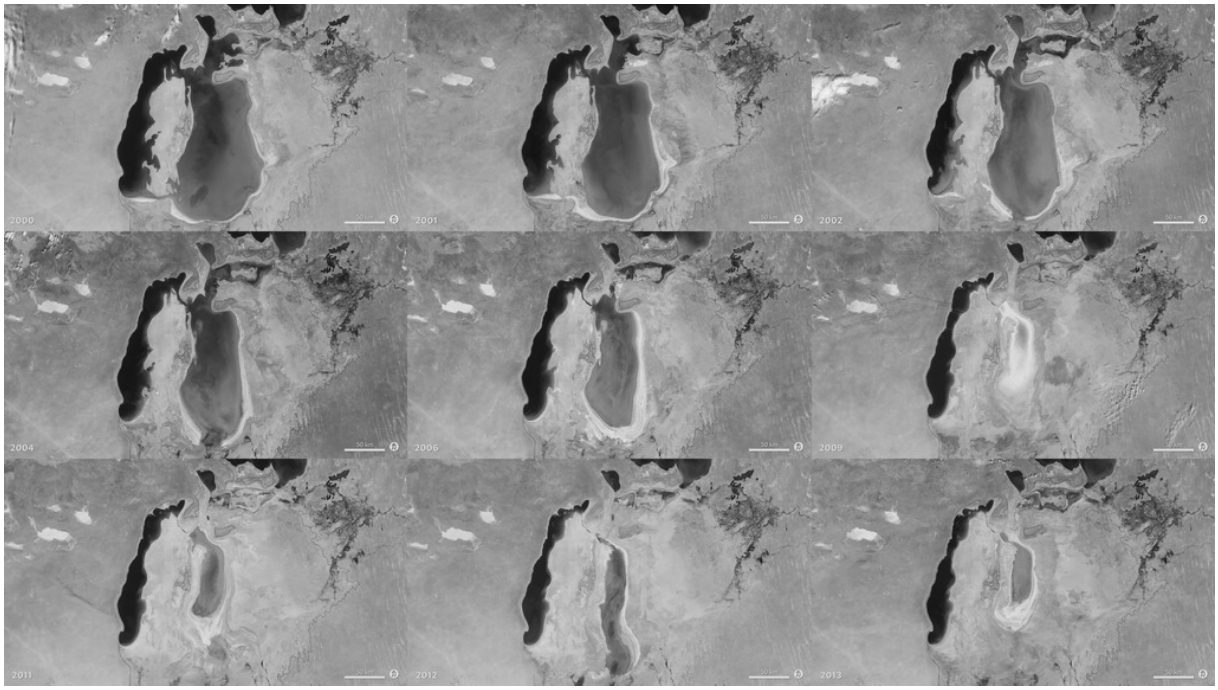


Figure 4.1: Gradual shrinking of the Aral Sea over time. (Image source: NASA). (a) Images in the upper panel were captured in 2000, 2001, and 2002. (b)Images in the middle panel are captured in 2004, 2006, and 2009. (c) Images in the lower panel were captured in 2011, 2012, and 2013.

The proposed drift monitoring algorithm demonstrates broad applicability across a range of scientific domains. In medical imaging, for example, it can be employed to monitor temporal changes such as tumor progression or regression, thereby supporting the evaluation of treatment efficacy over time. More generally, the methodology holds promise for integration into artificial intelligence frameworks aimed at automated monitoring and decision-making. By enabling the detection and characterization of gradual structural changes and their patterns in complex data, the proposed approach contributes to the advancement of data-driven monitoring systems in both scientific research and industrial applications.

4.1.2 Novelty and contribution

Despite extensive literature on image-based quality control, most of the methods available in the literature are not primarily designed for drift pattern surveillance. Although there are various control charts for both univariate and multivariate processes in the SPC literature Yi and Qiu (2022); Su et al. (2011), they are not directly useful for drift pattern monitoring in the image data.

In this chapter, we introduce a novel image monitoring framework designed to detect changes in the drift pattern within a sequence of multi-temporal images. A shift in the drift pattern refers specifically to a change in the manner in which the jump location curves (JLCs) evolve gradually over time. The proposed method operates as follows: at each time-point, the drift pattern is captured by (i) fitting an oblique-axis regression tree (ORT) to a sequence of preceding images, and (ii) estimating a denoised version of the current image based on the predicted splitting rules derived from the historical sequence. A CUSUM statistic is then constructed by quantifying the overall discrepancy between the denoised image and the observed image at the current time point. The core intuition is that if the current image follows the same drift pattern as the preceding sequence, the denoised image, generated using the inferred splitting rules, should provide a close approximation to the actual image. Consequently, in the absence of a change in the drift pattern, this discrepancy is expected to remain small. The key advantage of the proposed approach lies in its ability to preserve the structural features of the JLCs while effectively tracking their gradual temporal evolution. Although such pattern monitoring of gradual changes in JLCs is prevalent in many practical settings, it has received limited attention in the image monitoring literature. A recent effort by Yi and Qiu (2023) addresses a related problem in the context of satellite imagery; however, their approach focuses on calculating the area of image objects and employs a univariate likelihood ratio-based framework for drift detection. In contrast, the method proposed in this chapter is tailored to handle more general image types and is particularly well-suited for monitoring drift patterns in JLCs, offering a more flexible and structurally informed alternative.

The main contributions of this chapter and key highlights of the proposed methodology are summarized below:

- The core novelty of the proposed method lies in the introduction of a CUSUM-based framework for detecting changes in the pattern of gradual evolution within a sequence of images.
- Leveraging the ORT-based construction, the method effectively preserves jump discontinuities in the image intensity surface, which are often critical features in many monitoring applications.
- By focusing primarily on the evolution of jump location curves (JLCs), the method inherently suppresses insignificant background anomalies, which is an especially desirable property in satellite imaging where such anomalies are common.
- Although the methodology is designed to detect changes in drift patterns, it is also capable of signaling abrupt step shifts in the absence of any drift, thereby offering a high degree of adaptability.
- The proposed framework is broadly applicable to a wide range of image monitoring tasks. Its practical utility is demonstrated through extensive simulation studies, which highlight its superior performance, and a real-world application.

4.1.3 Organization

The remainder of the chapter is organized as follows. Section 4.2 introduces the proposed methodology in detail. Section 4.3 discusses theoretical properties of the proposed control chart. Section 4.4 presents numerical studies, including both simulation experiments and a real-world application. Finally, a few remarks in Section 4.5 conclude this chapter.

4.2 Proposed Methodology

This chapter aims to track the pattern of the drift in the sequence of image data. While the current methodology is primarily designed for a sequence of 2D grayscale images, it is also applicable to image sequences in other dimensions, e.g., a sequence of 3D images. In this section, we present the proposed methodology in a step-by-step manner.

4.2.1 Image pre-processing

The initial step in image monitoring is the image pre-processing, to convert the sequence of images into a usable format. In satellite images, the images are often not geometrically aligned. For reliable monitoring of the images, they should be geometrically aligned. In the literature, this alignment process is commonly referred to as *image registration* Qiu and Xing (2013a); Xing and Qiu (2011). In this chapter, we assume that the image sequence is already registered. In case it's not, we apply an image registration algorithm before proceeding with monitoring. Pre-processing also involves tasks such as resizing the images to the same resolution, scaling the image intensities to a given range, or removing irrelevant background regions.

4.2.2 Oblique-axis regression tree

Tree-based methods are widely used in modern statistics and machine learning. In this chapter, we develop a monitoring scheme where the control statistic is constructed using an approach based on a special type of decision tree called the oblique-axis regression tree (ORT). The choice of ORT is mainly motivated by the discontinuous nature of the image intensity function, which often displays an image object with a complex shape. Traditional axis-aligned CART algorithms, such as Breiman et al. (1984) create splits based on individual covariates while the search space is restricted to the standard axis directions. In contrast, oblique CART permits splits based on linear combinations of covariates, thus expanding the search space. The key distinction between standard CART and oblique CART lies in how they partition the predictor space. While the former creates rectangular regions, ORT produces convex polytopes. Since the image intensity functions are often discontinuous and the locations of the discontinuities may not align with the coordinate axes, ORT offers a more suitable approach for accurately capturing the object boundaries or the edges of the images. For a comprehensive discussion on ORT, readers are referred to Zhan et al. (2026); Cattaneo et al. (2024). A recent research focusing on the effectiveness of ORT in preserving jumps in discontinuous regression surfaces can be found in Basak et al. (2026).

4.2.3 Jump regression on a sequence of images

Under jump regression analysis (JRA) Qiu (2005), a sequence of 2D $n \times n$ grayscale images can be considered to follow the model:

$$\mathcal{I}_k(i,j) = w(x_i, y_j, t_k) = f(x_i, y_j, t_k) + \varepsilon(x_i, y_j, t_k) \quad \text{for} \quad i,j = 1,2,\ldots,n;\ k = 1,2,\ldots, \tag{4.1}$$

where $\mathcal{I}_k$ is the observed image at k-th time point, $w(x_i, y_j, t_k)$ is the observed image intensity at (i,j)-th pixel coordinate at k-th time point t_k, $f(x_i, y_j, t_k)$ is the corresponding unknown true image intensity value and $\varepsilon(x_i, y_j, t_k)$ are the independent and identically distributed (i.i.d.) pointwise random noise with mean 0 and variance σ^2. For the image data, the pixel coordinates lie on an equally spaced lattice defined on the design space $\Omega = [0,1] \times [0,1]$, namely, $\{(x_i, y_j) = (i/n, j/n) : i,j = 1,2,\ldots,n\}$. The observed time points are also assumed to be equally spaced. Under the JRA model, the true image intensity function is continuous in the design space Ω for each t_k, except on the edges in the boundary of the image object. In this chapter, we assume that f is piecewise constant for simplicity. Therefore, f can also be written as:

$$f(x,y,t) = \sum_{\ell=1}^{J} c_\ell \mathbb{1}_{\Gamma_\ell}(x,y,t),$$

where $\{\Gamma_\ell : \ell = 1,2,\ldots,J\}$ is a partition of Ω. The boundaries of these regions are the discontinuity points of the image. These are also called jump location curves or JLCs in the JRA literature. In this chapter, we also assume that the JLCs are piecewise linear except for finitely many points. For ease of notation, we will be assuming that JLCs are linear, i.e, we will be considering each linear part of the piecewise linear JLCs to be different JLCs. In this context, we consider the sequence of images as a three-dimensional image, so any change or discontinuity along the time dimension will also be reflected in the JLCs.

Definition 1: We say that the point (x_i, y_j) is drifting in direction (β_1, β_2) if:

$$f(x_i + \beta_1 * t, y_j + \beta_2 * t, t) = f(x_i, y_j, 0).$$

From the above definition, some confusion arises because the image intensity function is locally constant in certain regions. Moreover, the above definition does not emphasize the fact that all points in a local neighborhood should drift in the same direction. Therefore, it is more appropriate to discuss drifts on image objects instead of points. As JLCs usually surround an object, we now discuss drifts in a sequence of images in terms of the drifts of JLCs.

Definition 2: Suppose that $\mathcal{I}_k$ is the observed image captured at time t_k, and Γ is a JLC on it, then we say that Γ is drifting if all points on it are drifting in the same direction.

Any linear JLC in $\mathcal{I}_k$ can be expressed in the form $\alpha_1 x + \alpha_2 y < c, (x, y) \in S$ where S is a subset of Ω. If it is drifting, then we can rewrite this as $\alpha_1(x + \beta_1 t) + \alpha_2(y + \beta_2 t) < c \implies \alpha_1 x + \alpha_2 y + \alpha_3 t < c$, where $\alpha_3 = \alpha_1\beta_1 + \alpha_2\beta_2$. If (β_1, β_2) changes, say to (β_1', β_2'), then $\alpha_3' = \alpha_3 \implies \alpha_1\beta_1 + \alpha_2\beta_2 = \alpha_1\beta_1' + \alpha_2\beta_2' \implies \frac{\beta_1' - \beta_1}{\beta_2' - \beta_2} = -\frac{\alpha_2}{\alpha 1}$. Therefore, if this relationship does not hold, it means that α_3 has changed.

Now this situation implies that the JLC is moving along the resultant of the previous direction and a direction parallel to the JLC itself. So, for piecewise constant images, it should be visually indistinguishable from the starting image unless there is another JLC present in the image, which is not parallel to the first JLC. So changes of this nature will be reflected in the drift of some other JLC.

In the oblique regression tree method, since we split according to similar rules, it should be able to capture drifting JLCs properly, and it should only detect a change with respect to time if the slope $(\alpha_1, \alpha_2, \alpha_3)$ changes abruptly at some point. One thing to note here is that the proposed method cannot determine (β_1, β_2) exactly, although we will be able to detect any change in that direction.

4.2.4 Construction of the ORT-based decision tree

In this subsection, we discuss the recursive tree splitting algorithm, a pivotal step for the proposed drift pattern monitoring method. For any given node $\mathcal{N}$, direction α and constant c, we define $\mathcal{N}_L = \{z : \alpha^T z \leq c\}$ and $\mathcal{N}_R = \{z : \alpha^T z > c\}$ as the children nodes, where $z = (x, y, t)$. The key idea behind the tree construction is to hierarchically

partition the data by greedy binary splitting algorithm. Now for a given parent node $\mathcal{N}$ and the splitting rule denoted by the pair (α, c), we define:

$$\widehat{\Delta}(\mathcal{N}, \alpha, c) = \frac{1}{|\mathcal{N}|}\Big[\sum_{z\in\mathcal{N}}\big(w(z)-\bar{w}\big)^2 - \sum_{z\in\mathcal{N}_L}\big(w(z)-\bar{w}_L\big)^2 - \sum_{z\in\mathcal{N}_R}\big(w(z)-\bar{w}_R\big)^2\Big], \quad (4.2)$$

where $\bar{w}$,$\bar{w}_L$, and $\bar{w}_r$ are the average image intensities over the respective nodes $\mathcal{N}$, $\mathcal{N}_L$, and $\mathcal{N}_R$. Note that $\widehat{\Delta}(\mathcal{N}, \alpha, c)$ is the reduction in the sum of squares (SSE) if we divide the node using the rule (α, c). We denote the maximum value of $\widehat{\Delta}(\mathcal{N}, \alpha, c)$ over (α, c) by $\widehat{\Delta}(\mathcal{N})$. We split the node $\mathcal{N}$ if the value of $\widehat{\Delta}(\mathcal{N})$ is greater than a pre-determined threshold. Now, to construct the regression tree, we start by considering all observed intensity values across the images and time as a node, label them with their co-ordinates respectively, and then follow Algorithm 2.

Algorithm 2 Recursive Tree Construction

Input: A sequence of images labeled with respect to time.
Start: We have one node $\mathcal{N}$, the root, in the tree.
Calculate $\widehat{\Delta}(\mathcal{N})$.
if $\widehat{\Delta}(\mathcal{N}) \leq$ cutoff **then**
  Stop and return the tree you already have.
else
  Calculate $(\widehat{\alpha}_{opt}, \widehat{c}_{opt}) = \arg\max_{(\alpha,c)} \widehat{\Delta}(\mathcal{N}, \alpha, c)$.
  Split $\mathcal{N}$ into $\mathcal{N}_R$ and $\mathcal{N}_L$ using the line $\{\widehat{\alpha}_{opt}^T z = \widehat{c}_{opt}\}$.
end if
Calculate trees T_1 and T_2 recursively, using $\mathcal{N}_R$ and $\mathcal{N}_L$ as root nodes.
Join trees T_1 and T_2 with $\mathcal{N}$ to get the final tree T.
Output: T is the estimated tree.

Consider a sequence of images of length m with labeled time points, denoted as $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_m\}$ following the jump regression model expressed in Eqn. (4.1). Now that after implementing the binary splitting algorithm we have the estimated tree T, consider the corresponding set of decision rules denoted as T_{rule}, which consists of all decision rules while constructing the tree. Suppose, we are given an image $\mathcal{I} \notin \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_m\}$ labeled with time $t_{\mathcal{I}}$. Then, by applying the rule T_{rule} on that image, we get:

$$T_{rule}(\mathcal{I}, t_{\mathcal{I}}) = \{\mathcal{L}^i_{T,\mathcal{I}} | i \in \{1, 2, \ldots, K^m_n(\mathcal{I})\}\},$$

where $K^m_n(\mathcal{I})$ is the number of leaf nodes that depends on the image, its resolution and

the length of the image sequence, and $\mathcal{L}^i_{T,\mathcal{I}}$ are the leaf nodes in the image $\mathcal{I}$ after applying the decision rules T_{rule}. Typically, for a given sequence, m and n are kept fixed in the proposed algorithm. Therefore, to make the notation less cumbersome, we will denote it with $K(\mathcal{I})$ later in the chapter. Intuitively, the intensity values in a leaf node only contain those from similar pixel coordinates. Then, the proposed denoised estimator of $t_{\mathcal{I}}$ at the point (x, y) is based on the leaf-only averaging, which is given by

$$\widehat{f}_{t_{\mathcal{I}}}(x, y) = \frac{1}{|\mathcal{L}_{T,\mathcal{I}}(x, y, t_{\mathcal{I}})|} \sum_{(x_p, y_q, t_I) \in \mathcal{L}_{T,\mathcal{I}}(x,y,t_I)} \mathcal{I}(p, q), \tag{4.3}$$

where $\mathcal{I}(p, q)$ denotes the intensity of the image at the pixel (x_p, y_q) and $\mathcal{L}_{T,\mathcal{I}}(x, y, t_{\mathcal{I}})$ corresponds to that leaf node in $T_{rule}(\mathcal{I}, t_{\mathcal{I}})$ which contains the point $(x, y, t_{\mathcal{I}})$.

Remarks: Note that the above estimator based on the leaf-averaging is a jump preserving estimator of the image intensity function accommodating the temporal pattern or the drift. If $\mathcal{I} \in \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_m\}$, then the estimated image is the denoised image based on the drift information provided by the known sequence. Hence, this method can also be used as an image denoising method for a sequence of images. The proposed image monitoring method under a drift can, therefore, accommodate the jumps in the discontinuous image intensity function. For certain applications, to accommodate the intricate details of the image, the leaf-only-averaging may not work well. In those applications, one can modify the proposed method by introducing local-leaf-only averaging. Theoretical consistency of the proposed algorithm can be shown similarly as provided in Basak et al. (2026).

4.2.5 Phase II monitoring using ORT

In traditional SPC literature, process monitoring is typically divided into two phases: Phase I and Phase II. In this chapter, we propose a CUSUM control chart for Phase II monitoring of the temporal drift pattern in the image sequence based on the ORT framework. The proposed approach consists of two major components: Firstly, building a predictive model that captures the drift pattern; and secondly, formulating a CUSUM statistic to detect changes in the drift pattern. In the Phase II stage, the observed image at the k-th time-point can be expressed by following the same model as given in Eqn.

(4.1). To enable online monitoring of the sequential process, we assume that an initial in-control (IC) dataset $\Psi_{M,IC}^{(0)} = \{\mathcal{I}_{-M+1}, \mathcal{I}_{-M+2}, \ldots, \mathcal{I}_0\}$ of size M to be available. As the change occurs gradually, such prior data are required to capture the underlying drift pattern and to start the online monitoring. The time period of the initial IC dataset is then set as a baseline time interval. The primary objective of online monitoring is to detect any substantial deviation in the future drift pattern of the process from its regular drift pattern occurring in the baseline time interval as promptly as possible. Now at any time point t_{k-1}, a regression tree is fitted using the available image data from the past. Since involving all prior images is computationally challenging, we adopt a moving window approach of length m_0, and fit the ORT on the most recent m_0 number of images only. In this chapter, we refer to m_0 as the window width. We choose the window width m_0 to be strictly smaller than the total number of available observations in the baseline time interval so that the remaining $(M - m_0)$ images can be used to estimate the average prediction error, acceptable drift, and control limits, and so forth when these values are unknown. Additionally, it is a natural assumption that the choice of m_0 should cover the entire cycle of a full drift pattern. Now at time t_{k-1}, we have the partition based on the regression tree with images from time $t_{(k-m_0)}$ to t_{k-1}, and we use this to predict the partition of the image captured at time t_k. To this end, we compute the denoised image at the time-point t_k based on the predicted tree partition. The predicted partition can be calculated by putting $t = t_k$ at the constructed decision rule based on the previous m_0 number of images. Mathematically, based on the image sequence $\Psi_{m_0}^{(k-1)} = \{\mathcal{I}_{(k-m_0)}, \ldots, \mathcal{I}_{(k-1)}\}$, consider the data with dimension $n \times n \times m_0$, and fit a decision tree on it using the greedy algorithm mentioned in Section 4.2.4. Then, similar to Eqn. (4.3), the denoised image at the k-th time point with the predicted partition based on the image sequence $\Psi_{m_0}^{(k-1)}$ is defined as

$$\widehat{f}_{t_k}^{P}(x, y) = \frac{1}{|\mathcal{L}_{T,k}^{P}(x, y, t_k)|} \sum_{(i,j,t_k) \in \mathcal{L}_{T,k}(p,q,t_k)} \mathcal{I}_{t_k}(p, q),$$

where $\mathcal{L}_{T,k}^{P}(x, y, t_k)$ corresponds to the leaf node at the k-th time point based on the predicted decision rule by putting $t = t_k$ in the constructed decision rule on the image sequence $\Psi_{m_0}^{(k-1)}$. $\mathcal{I}_{t_k}(p, q)$ indicates the observed image intensity at (x_p, y_q)-th coordinate of image captured at the k-th time-point, i.e., $\mathcal{I}_k$. Then, the overall difference between

the observed image at t_k and predicted denoised image at t_k is defined as

$$\Lambda_{t_k} = \frac{1}{(n^2 - K(\mathcal{I}_{t_k}))\theta^2} \sum_{i,j}^{n} (\widehat{f}^{P}_{t_k}(x_i, y_j) - w(x_i, y_j, t_k))^2, \tag{4.4}$$

where θ^2 is the sum of noise variance and squared prediction error. In practice, we estimate θ^2 by the average MSE of observed in-control data when compared to the estimates given by the ORT algorithm. A mathematical formulation of this is given in Theorem 13. Observe that the denominator in Eqn. (4.4) is the degree of freedom of the sum, as there

Algorithm 3 The proposed drift pattern monitoring scheme for a sequence of images

Input: A sequence of $n \times n$ images labeled with respect to time, available at the k-th time-point, window width m_0 for computation of estimates, value for acceptable amount of change κ, an estimated value of noise θ^2, and a cutoff value.

Start: We start with $k = 1$.

for $k <$ *(the length of the image sequence)* **do**

Fit a tree using algorithm 4.2.4 using images in $\Psi_{m_0}^{(k-1)}$.

Calculate $T_{rule}^{t_k}$, the decision rules at time t_k, based on the tree obtained.

Using $T_{rule}^{t_k}$, calculate $\widehat{f}_{t_k}(x, y)$.

Calculate Λ_{t_k} and Q_{t_k}.

Report Q_{t_k}.

$k = k + 1$

end for

are only $K(\mathcal{I}_{t_k})$ non-empty leaf-nodes in the fitted tree. Additionally, it is important to note that, if there is no change in the drift pattern of the image, this estimate should be close enough to the observed image, and Λ_{t_k} would be small because it contains only the noise. Conversely, if a change in the drift pattern has occurred, then the predicted denoised image and the observed image should not be similar, and Λ_{t_k} would be large. Therefore, by calculating the prediction error, we can detect if a change in the drift pattern has occurred between the time-points t_k and $t_{(k+1)}$. Theorem 13 shows that after suitable centering and scaling, the mean squared error asymptotically follows a normal

distribution. Thus, a natural CUSUM charting statistic can be defined by

$$Q_{t_k} = max\left(0, Q_{t_{(k-1)}} + \frac{n}{\sqrt{2}}\left(\Lambda_{t_k} - 1\right) - \kappa\right), \quad k = 1, 2, \ldots, \tag{4.5}$$

where $Q_{t_0} = 0$, and $\kappa \geq 0$ is a pre-specified allowance parameter. To choose the allowance parameter optimally, the readers are referred to Qiu (2013). The proposed CUSUM statistic raises a signal if

$$Q_{t_k} > q_0,$$

where q_0 is a pre-determined control limit. Traditionally, the control limit is chosen in such a way that the in-control *average run length* (ARL) of the process can reach the pre-fixed nominal level of ARL_0. The in-control ARL is defined as the expected time to signal when there is no change in the drift pattern.

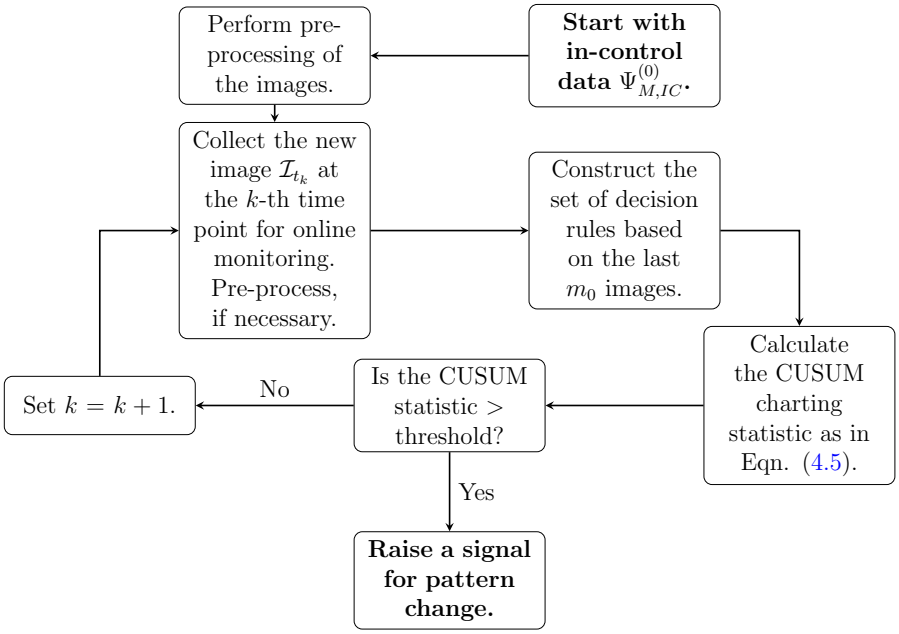


Figure 4.2: The flowchart diagram for the proposed method for monitoring drift pattern in an image sequence.

4.3 Statistical Properties

In this section, we investigate certain theoretical properties of the proposed online drift detection algorithm. Before going into the main results, we introduce the regulatory assumptions below, which are reasonable in a wide range of practical applications.

(i) f is piecewise Lipschitz.

(ii) f is bounded inside its domain $[0,1]^2$.

(iii) The points in $\overline{\Gamma_\ell} \backslash int(\Gamma_\ell)$ are called jump points, and the set of jump points has Lebesgue measure zero.

(iv) There exists at most finitely many JLCs.

(v) Each JLC is a line segment.

(vi) If f is continuous at a jump point, then that point is called a singular point.

(vii) The point-wise noise ε are independently distributed and distributed as $\mathbf{N}(0, \sigma^2)$.

Suppose that we are given a sequence of images of resolution $n \times n$ captured at time-points $\{t_1, t_2, \dots\}$. Consider the following test of hypotheses:

$$\mathcal{H}_0 : \textit{ Drift pattern of the JLCs have not changed}$$
$$\textit{vs.}$$
$$\mathcal{H}_1 : \textit{ Drift pattern of at least one JLC has changed}$$

The following Theorem provides asymptotic behavior of the charting statistic under both $\mathcal{H}_0$ and $\mathcal{H}_1$.

Theorem 13. *Suppose that* $\sup_i |t_{i+1} - t_i| \to 0$ *as* $n \to \infty$*, and the assumptions listed above hold true. Then, we have the following results:*

(i) Under $\mathcal{H}_0$, $\frac{n}{\sqrt{2}}(\Lambda_{t_k} - 1) \xrightarrow{d} \mathcal{N}(0,1)$.
(ii) Under $\mathcal{H}_1$, $\frac{n}{\sqrt{2}}\Big(\Lambda_{t_k} - 1 - O(1)\Delta^2\Big) \succ \mathcal{N}\left(0, 1 + \frac{E_{t_k} + O(1)\Delta^2}{1 + E_{t_k}}\right)$*, where* $\Delta > 0$ *is the deviation associated with the change in the drift pattern, and* E_{t_k} *corresponds to the estimation error related to ORT based image smoothing.*

The above Theorem is a direct consequence of the results provided in 3. For the reader's convenience, we will restate the results that will be used in this proof.

We first introduce the following shorter notations to facilitate subsequent discussions.

- $\mathcal{L}_n$ is a leaf node of the fitted tree when the resolution is n^p, p being the dimension of the data. For a 2-D image sequence over time, $p = 3$.
- $\mathcal{L}_n(x, y, t)$ is the leaf-node of the above tree which contains the point (x, y, t).
- μ_k is the Lebesgue measure in $\mathbb{R}^k$.

Note that the proofs of the following results are provided in Chapter 3.

Lemma. *If* $\lim_{n\to\infty} \mu_p(\mathcal{L}_n) \neq 0$, *then* f *is a.e. constant in* $\bigcap_n \mathcal{L}_n$.

Lemma. *If* $\lim_{n\to\infty} \mu_p(\mathcal{L}_n) = 0$ *and* $\bigcap_n \mathcal{L}_n = \mathcal{L}$ *lies on a* k *dimensional affine subspace of* $\mathbb{R}^p$ *with* $k < p$ *and* $\mu_k(\mathcal{L}) \neq 0$, *then one of these is true:*

(i) $\mathcal{L}$ *contains an entire JLC.*

(ii) $\mathcal{L}$ *is contained entirely inside a JLC.*

(iii) f *is a.e. constant on* $\mathcal{L}$.

Lemma. *Let* $\gamma(A)$ *denote the maximum distance between two points in the set* A. *If* $\lim_{n\to\infty} \mu_p(\mathcal{L}_n) = 0$ *and* $\lim_{n\to\infty} \gamma(\mathcal{L}_n) \neq 0$, *then one of the following is true:*

(i) $\bigcap_n \mathcal{L}_n$ *contains an entire JLC.*

(ii) $\bigcap_n \mathcal{L}_n$ *is contained entirely inside a JLC.*

(iii) f *is a.e. constant on* $\bigcap_n \mathcal{L}_n$.

Lemma. *Let* (x, y, t) *be a non-singular continuity point of* f. *If* $\lim_{n\to\infty} \gamma(\mathcal{L}_n(x, y, t)) = 0$, *then* $\exists\ N$ *such that* $\forall\ m > N$, $\mathcal{L}_m(x, y, t)$ *does not contain any discontinuity point.*

Lemma. $\lim_{n\to\infty} \widehat{f}_n(x, y, t) = f(x, y, t)$ *a.e., almost surely.*

Remarks: These lemmas provide results only when the parameter space contains the time point t, i,e., t is within the range of time points of images used in denoising. However, in the drift monitoring applications, that is not correct. In this algorithm, we predict the image observed at time t_{k+1} based on the image data up to time point t_k only. We now justify that below.

Proof of Theorem 1: Under the given assumptions, $\widehat{f}_{n,t_k}(x,y,t) \to f_n(x,y,t)$ as $n \to \infty$, $\forall t \leq t_k$ as long as (x,y,t) is not on a JLC.

Consider a point (x,y,t_k) such that it is not on any JLC, i.e., $(x,y,t_k) \notin \bigcup_\ell \left[\overline{\Gamma_\ell} \backslash int(\Gamma_\ell)\right]$. Assume that $(x,y,t_k) \in int(\Gamma_{\ell^*})$ for some ℓ^*. As it is an interior point, $|t_{k+1} - t_k| \to 0$, and under $\mathcal{H}_0$, the drifts of the JLCs remain unchanged, we also have $(x,y,t_{k+1}) \in int(\Gamma_{\ell^*})$.

Since f is continuous at (x,y,t_k), we have

$$\left| f(x,y,t_{k+1}) - f(x,y,t_k) \right| \to 0. \tag{4.6}$$

As $(x,y,t_k) \notin \bigcup_\ell \left[\overline{\Gamma_\ell} \backslash int(\Gamma_\ell)\right]$, we have:

$$\left| f(x,y,t_k) - \widehat{f}_{t_k}(x,y,t_k) \right| \to 0. \tag{4.7}$$

The above result was proven in Theorem 12. Basak et al. (2026).

Now, if $(x,y,t_{k+1}) \in \mathcal{L}_k(x,y,t_k)$ then $\widehat{f}_{t_k}(x,y,t_k) = \widehat{f}_{t_k}(x,y,t_{k+1})$. Otherwise, we have one of the following two cases:

Case 1: $\gamma(\mathcal{L}(x,y,t_{k+1})) \to 0$. Note that $\gamma(A)$ denotes the maximum distance between two points in the set A. In this case, we have

$$\left| \widehat{f}_{t_k}(x,y,t_{k+1}) - \widehat{f}_{t_k}(x,y,t_k) \right| \leq C \left| t_{k+1} - t_k + \gamma(\mathcal{L}(x,y,t_{k+1}) \right| \to 0,$$

as both $(x,y,t_k), (x,t,t_{k+1}) \in \Gamma_i$.

Case 2: $\lim_{n\to\infty} \mu_p(\mathcal{L}_n) \neq 0$ or $\gamma(\mathcal{L}(x,y,t_{k+1})) \not\to 0$. Both of these scenarios imply that f is constant in $\mathcal{L}(x,y,t_{k+1})$. Hence,

$$\left| \widehat{f}_{t_k}(x,y,t_{k+1}) - \widehat{f}_{t_k}(x,y,t_k) \right| \leq C.\ \mathrm{dist}\big((x,y,t_k), \mathcal{L}(x,y,t_{k+1})\big) \leq C \left| t_{k+1} - t_k \right| \to 0.$$

Therefore, in both cases we have:

$$\left| \widehat{f}_{t_k}(x,y,t_{k+1}) - \widehat{f}_{t_k}(x,y,t_k) \right| \to 0. \tag{4.8}$$

From the Eqns. 4.6, 4.7 and 4.8, we get $\delta_{t_{k+1}}(x,y) = \left| f(x,y,t_{k+1}) - \widehat{f}_{t_k}(x,y,t_{k+1}) \right| \to 0$.

Therefore, we have

$$\begin{aligned} z_{t_k}(x,y) &= (w(x,y,t_k) - \widehat{f}_{n,t_k}(x,y,t_k))^2 \\ &= (\varepsilon(x,y,t_k) + \delta(x,y,t_k))^2. \end{aligned}$$

Hence, $z_{t_k}(x,y) \stackrel{d}{=} \sigma^2 \chi_1^2 \left(\left[\frac{\delta_{t_k}(x,y)}{\sigma} \right]^2 \right)$. Observe that $\frac{1}{n^2} \sum_{(x,y)} z_{t_k}(x,y)$ has asymptotic mean $\sigma^2(1+E_{t_k})$, where $E_{t_k} = \frac{1}{\sigma^2} \int_0^1 \int_0^1 \delta_{t_k}(x,y)^2 dxdy$. Noe that $E_{t_k} \to 0$ as $\delta_{t_K} \to 0$ *a.e.*, and it is bounded by 2. Hence, we have

$$\begin{aligned} \theta^2 \Lambda_{t_k} \stackrel{d}{=} \frac{\sigma^2 \chi^2_{(n^2 - K(\mathcal{I}_{t_k}))} (\frac{1}{\sigma^2} \sum \delta_{t_k}(x,y)^2)}{(n^2 - K(\mathcal{I}_{t_k}))} &\approx \frac{\sigma^2 \chi^2_{n^2} (\frac{1}{\sigma^2} \sum \delta_{t_k}(x,y)^2)}{n^2}. \\ \implies n \left(\frac{\theta^2}{\sigma^2} \Lambda_{t_k} - 1 - E_{t_k} \right) &\stackrel{d}{=} \mathcal{N}\left(0, 2 + 4E_{t_k}\right). \\ \implies \frac{n}{\sqrt{2}} \left(\Lambda_{t_k} - (1+E_{t_k}) \frac{\sigma^2}{\theta^2} \right) &\stackrel{d}{=} \mathcal{N}\left(0, \frac{\sigma^2}{\theta^2}(1+2E_{t_k})\right). \end{aligned}$$

Since we consider $\theta^2 = \sigma^2(1+E_{t_k})$, we have

$$\frac{n}{\sqrt{2}}(\Lambda_{t_k} - 1) \stackrel{d}{=} \mathcal{N}\left(0, 1 + \frac{E_{t_k}}{1+E_{t_k}}\right) \stackrel{d}{\approx} \mathcal{N}(0,1) \text{ as } E_{t_k} \to 0. \tag{4.9}$$

Under $\mathcal{H}_1$, at least one JLC has not continued to the new time-point. Hence, the estimates will be incorrect at the points that were near the previous JLC. This deviation should be proportional to the jump size, say Δ. Note that in this context, we refer to the JLCs in the 3-D space, not in individual 2-D images. Since any linear JLC has approximately $O(n)$ many pixels, we can conclude that

$$\theta^2 \Lambda_{t_k} \succ \frac{1}{n^2} \sigma^2 \chi^2_{n^2} \left(\frac{1}{\sigma^2} \sum \delta_{t_k}(x,y)^2 + O(n)\Delta^2 \right).$$

Then, proceeding similarly as displayed in Eqn. (4.9), we have

$$\frac{n}{\sqrt{2}} \left(\Lambda_{t_k} - 1 - O(1)\Delta^2 \right) \succ \mathcal{N}\left(0, 1 + \frac{E_{t_k} + O(1)\Delta^2}{1+E_{t_k}}\right).$$

4.4 Numerical Studies

In this section, we first check the performance of the proposed drift monitoring method on various types of simulated data. Next, we compare these performances with those of various state-of-the-art image monitoring methods. Subsequently, we apply the proposed method to real satellite data on the Aral Sea.

4.4.1 Performance of the proposed method on various types of simulated data

We conduct extensive simulation studies to evaluate the performance of the proposed algorithm under various types of drift conditions. For demonstration, the images are simulated with JLCs consisting only of straight line segments, and we consider various types of drift along these segments to mimic real-world scenarios. Moreover, we add Gaussian noise with a fixed standard deviation of 0.15 to each of the simulated images having resolution 128×128. We set the window width m_0 to be 20 in all simulation examples.

Simulation 1: In the first image sequence, we consider a single JLC consisting of a vertical straight line moving horizontally as the reference. In this sequence, the line segment shifts one pixel to the right per unit of time. Figure 4.3 illustrates how the image changes over time. These changes occur linearly or at the same rate in the in-control image sequence. We try various values of the allowance parameter κ, and find the corresponding control limit q_0 such that the in-control ARL becomes 20 based on 100 independent replications for each trial value of κ. We then select that value of κ and the corresponding q_0 for which the plot of the CUSUM statistic visually exhibits minimal trend. In this case, the selected value of κ is 0.9. Using that allowance parameter, we determine the control limit q_0 for achieving the average run length (ARL) of 20, indicating that the test of hypothesis has a significance level of 0.05. Next, we generate a sequence where the drift rate of the line segment doubles from a pre-determined time of $t = 20$ in this example. We then calculate the detection delay using the same cutoff and permissible change values. The results show that the proposed algorithm can successfully detect the

drift change, with the average detection delay being very close to 1. The functional form of these two image sequences without noise is as follows:

$$f_{IC}(x, y, t) = \mathbb{1}\Big[x > \frac{t}{128}\Big],$$
$$f_{OC}(x, y, t) = \mathbb{1}\Big[x > \frac{t + t\mathbb{1}(t > 20)}{128}\Big],$$

where f_{IC} and f_{OC} represent the in-control and the out-of-control image sequence, respectively.

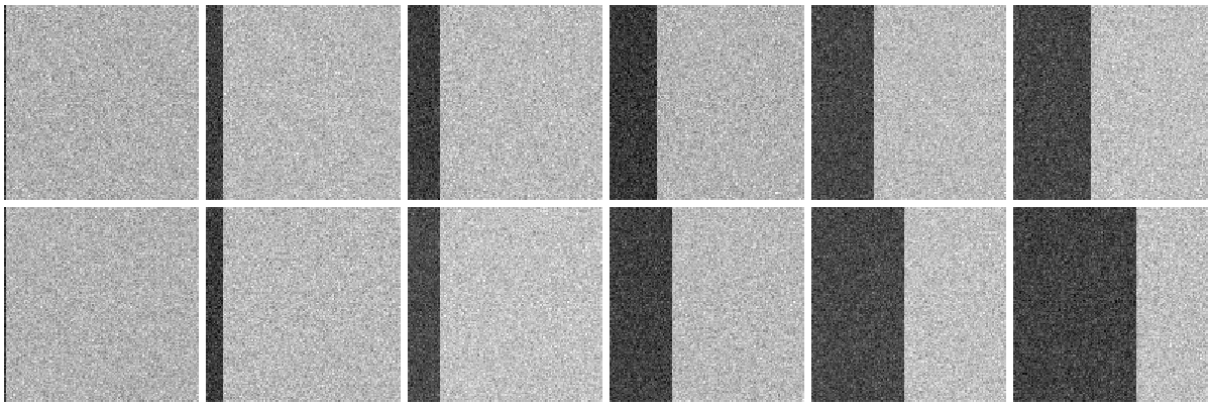


Figure 4.3: The top row shows the in-control images at time-points $t = 0$, 10, 20, 30, 40, and 50 for Simulation 1. The bottom row shows the corresponding images for the out-of-control image sequence.

Simulation 2: The second image sequence features a square with four linear JLCs as its arms. In this sequence, the JLCs move away from the center of the square at a rate of 1/4-th of a pixel length per unit of time. This sequence is designed to simulate a change in size or scale of an object. Figure 4.4 illustrates how the image changes over time. Following a similar procedure as in the previous simulation, we select the allowance parameter κ to be 2.0, and the corresponding control limit q_0 so that the in-control ARL becomes 20. Similar to the previous case, we double the rate of change after the predetermined time of $t = 20$, resulting in an image sequence where a change in the drift pattern occurs. Using the obtained control limit, we find that the average detection delay in this case is close to 4. Note that in this sequence, it takes 4 units of time to observe a change in the image for the in-control case, and since the standard deviation of the added noise is relatively large, this result is expected. The functional forms of the image

sequences without noise are as follows:

$$f_{IC}(x,y,t) = \mathbb{1}\Big[\max\big\{|x-0.5|,|y-0.5|\big\} > \frac{t+20}{512}\Big],$$
$$f_{OC}(x,y,t) = \mathbb{1}\Big[\max\big\{|x-0.5|,|y-0.5|\big\} > \frac{20+t+t\mathbb{1}(t>20)}{512}\Big].$$

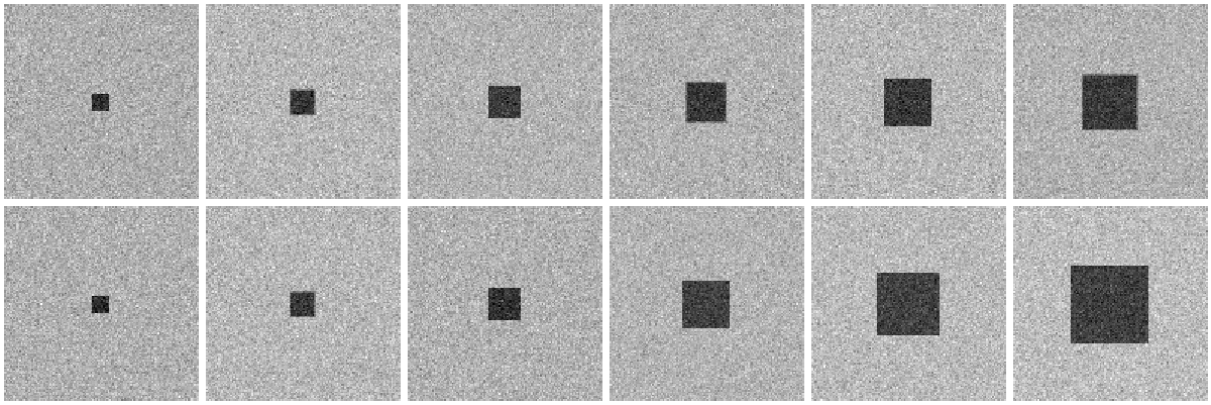


Figure 4.4: The top row shows the in-control images at time-points t = 0, 10, 20, 30, 40, and 50 for Simulation 2. The bottom row shows the corresponding images for the out-of-control image sequence.

Simulation 3: In this sequence, we analyze a square moving from the left to the right of the image frame at a speed of 1/2 of a pixel length per unit time. Following a similar procedure as in the previous two simulation cases, we find that the detection delay is approximately 2. This result is consistent with the findings from the second example. In this case, κ is selected to be 0.7. The functional form of these image sequences is as follows:

$$f_{IC}(x,y,t) = \mathbb{1}\big(max(|x-\frac{t}{256})|,|y-0.5|) > \frac{5}{128}\big),$$
$$f_{OC}(x,y,t) = \mathbb{1}\big(max(|x-\frac{t+t\mathbb{1}(t>20)}{256}|,|y-0.5|) > \frac{5}{128}\big).$$

Simulation 4: In the fourth simulation example, we demonstrate that the proposed method performs well when there is no drift in the in-control image sequence, and there occurs a change in JLC in the out-of-control image sequence. Following a similar procedure as in the previous simulation cases, we find that the average detection delay is close to 1, indicating that the proposed method can also be used to monitor images for any change

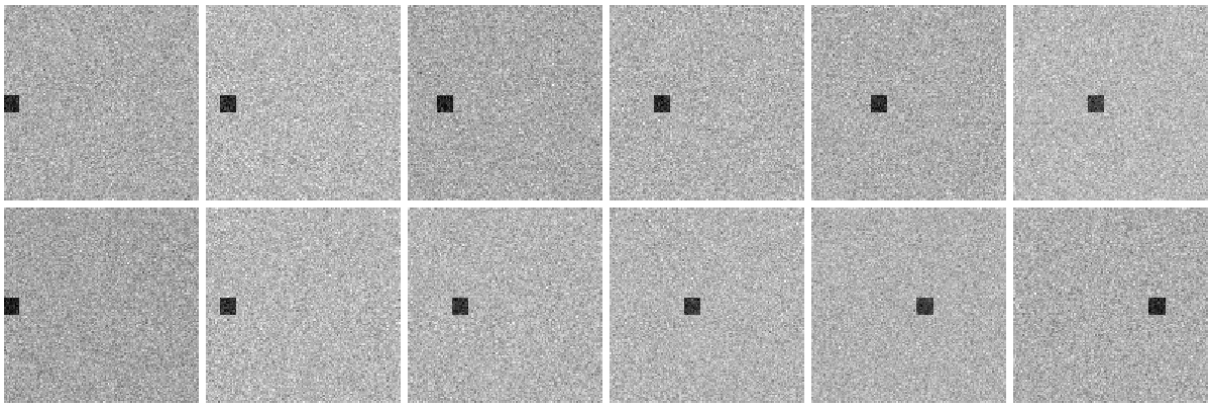


Figure 4.5: The top row shows the in-control images at time-points $t = 0$, 20, 40, 60, 80, and 100 for Simulation 3. The bottom row shows the corresponding images for the out-of-control image sequence.

in JLCs when there is no drift. In this case, κ is selected to be 2.0. The following two functions represent the in-control and out-of-control image sequences:

$$f_{IC}(x,y,t) = \mathbb{1}\Big((x,y) \notin \big[0.25, 0.75\big] \times \big[0.25, 0.75\big]\Big),$$

$$\begin{aligned} f_{OC}(x,y,t) = {} & \mathbb{1}(t \leq 20)\mathbb{1}\Big((x,y) \notin \big[0.25, 0.75\big] \times \big[0.25, 0.75\big]\Big) \\ & + \mathbb{1}(t > 20)\mathbb{1}\Big((x,y) \notin \big[0.25 - \frac{5}{128}, 0.75 + \frac{5}{128}\big] \times \big[0.25 - \frac{5}{128}, 0.75 + \frac{5}{128}\big]\Big). \end{aligned}$$

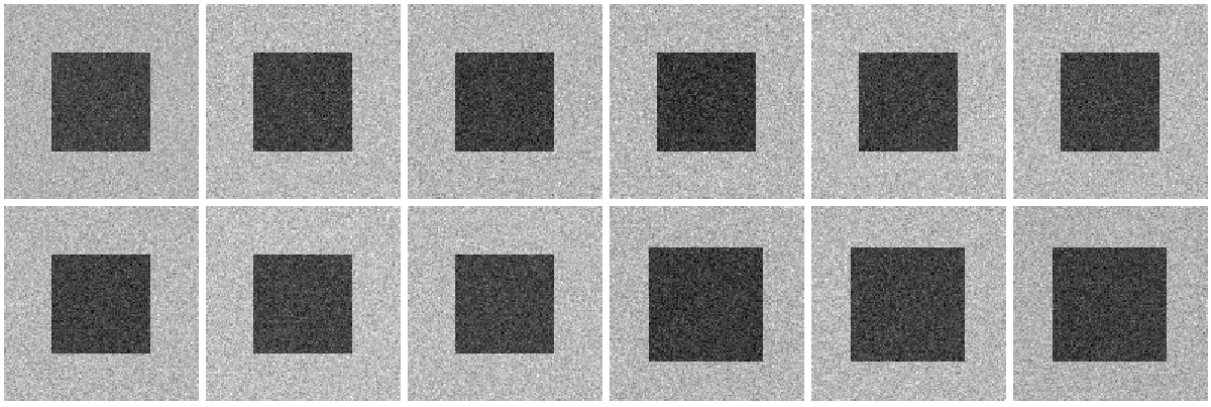


Figure 4.6: The top row shows the in-control images at time-points $t = 0$, 10, 20, 30, 40, and 50 for Simulation 4. The bottom row shows the corresponding images for the out-of-control image sequence.

Simulation 5: The last simulation differs from the previous ones as it demonstrates a scenario where the proposed method should not detect a change, despite an actual

change occurring in the image sequence. In this case, the image consists of a stationary square with an intensity value of 0 inside and 1 outside. However, instead of any spatial movement, the intensities themselves change over time. In the in-control sequence, the rate of change remains unaltered, while in the out-of-control sequence, the rate of change alters after $t = 20$. Note that the JLCs still remain unchanged in all images in both sequences. Therefore, the proposed method should not detect this type of change. Following a similar procedure as in the previous simulation cases, we find that the detection delay is approximately 20, which is close to the in-control ARL value itself. This indicates that the algorithm behaves similarly to cases where no change is present. Such a property of the proposed algorithm is advantageous in certain real-world applications, such as ignoring variations due to shadows, changes in lighting, or time-of-day effects. In this case, κ is selected to be 0.7. The following functions represent the image sequences:

$$f_{IC}(x, y, t) = t * \alpha + (\beta - t\alpha)\mathbb{1}\Big(max(x, y) > \frac{114}{128}\Big),$$
$$f_{OC}(x, y, t) = t * \alpha\big(1 + \mathbb{1}(t > 20)\big) + (\beta - t\alpha)\mathbb{1}\Big(max(x, y) > \frac{114}{128}\Big).$$

Here β is the starting intensity outside the square and α is the rate of change of intensity value. In this simulation, presented by Figure 4.7, we choose $\alpha = 0.005$, and $\beta = 1$.

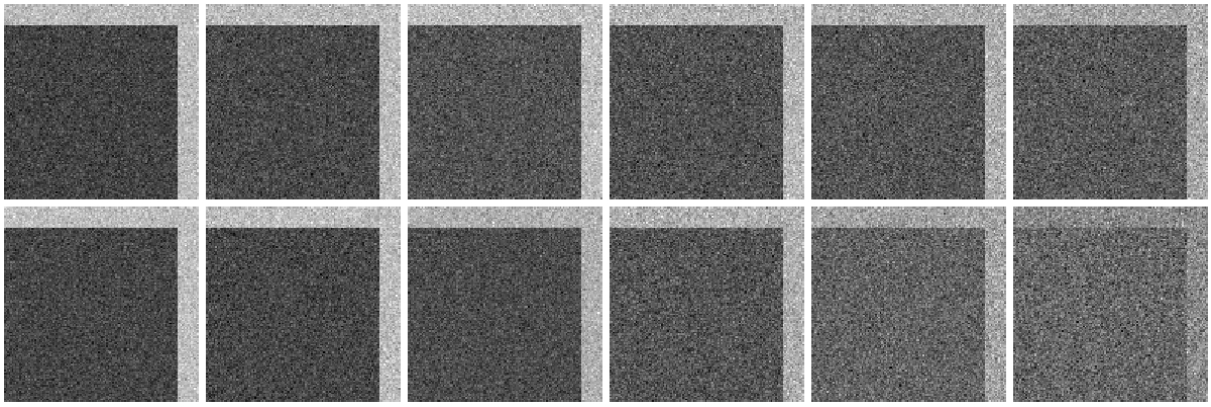


Figure 4.7: The top row shows the in-control images at time-points $t = 0$, 10, 20, 30, 40, and 50 for Simulation 5. The bottom row shows the corresponding images for the out-of-control image sequence.

4.4.2 Additional simulation studies:

We conduct two more simulation studies to assess the performance of our algorithm under rotating JLCs. The empirical results indicate that the algorithm performs reasonably well. However, the variance of the ARL estimates is noticeably higher than in the case of linearly drifting JLCs.

Simulation 6: In this simulation study, we consider an image consisting of two homogeneous regions, one white and one black, separated by a JLC. The JLC rotates over time. Under the in-control scenario, the rotational speed remains constant. In the out-of-control scenario, the speed changes after a pre-specified change point. Figure 4.8 illustrates how the images change over time in this simulation setting. The following functions characterize the resulting image sequences:

$$f_{IC}(x,y,t) = \mathbb{1}\Big(\Big(x-\tfrac{129}{2}\Big)\cos\Big(\tfrac{\pi}{40}t\Big) + \Big(y-\tfrac{129}{2}\Big)\sin\Big(\tfrac{\pi}{40}t\Big) \geq 0\Big),$$

$$f_{OC}(x,y,t) = \mathbb{1}\Big(\Big(x-\tfrac{129}{2}\Big)\cos\Big(\tfrac{\pi}{40}t\,\mathbb{1}(t\leq w) + \big(\tfrac{\pi}{40}w + \tfrac{\pi}{20}(t-w)\big)\mathbb{1}(t>w)\Big) + \Big(y-\tfrac{129}{2}\Big)\sin\Big(\tfrac{\pi}{40}t\,\mathbb{1}(t\leq w) + \big(\tfrac{\pi}{40}w + \tfrac{\pi}{20}(t-w)\big)\mathbb{1}(t>w)\Big) \geq 0\Big).$$

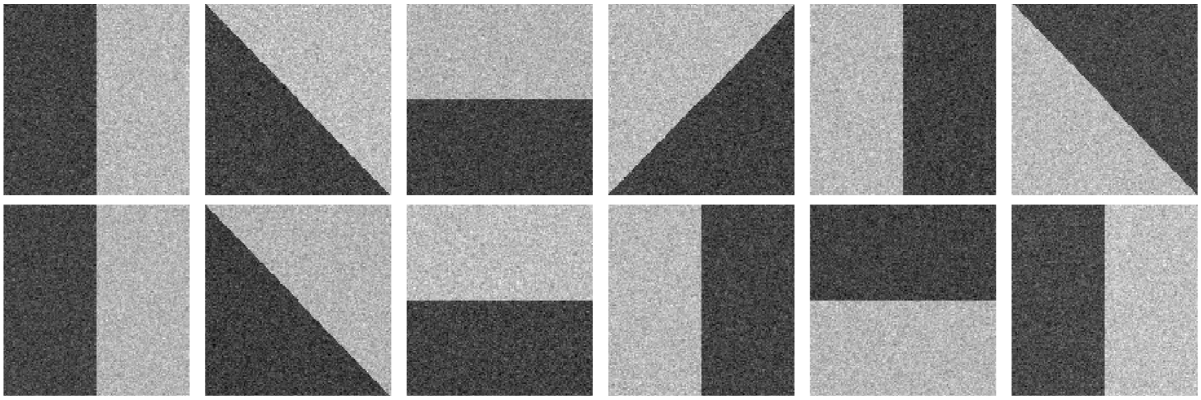


Figure 4.8: The top row shows the in-control images at time-points $t = 0$, 10, 20, 30, 40, and 50 for Simulation 6. The bottom row shows the corresponding images for the out-of-control image sequence.

Simulation 7: This simulation study follows a similar setup as before. However, instead of a rotating line, we consider an image sequence generated by a rotating square

box. Figure 4.9 illustrates how the images change over time in this simulation setting. The functional forms of the resulting image sequences are given below:

$$f_{IC}(x,y,t) = \mathbb{1}\Big(\big|\cos\big(\tfrac{\pi}{40}t\big)\Big(x-\tfrac{129}{2}\Big)+\sin\big(\tfrac{\pi}{40}t\big)\Big(y-\tfrac{129}{2}\Big)\big| \le 32\Big)$$
$$\cdot\, \mathbb{1}\Big(\big|-\sin\big(\tfrac{\pi}{40}t\big)\Big(x-\tfrac{129}{2}\Big)+\cos\big(\tfrac{\pi}{40}t\big)\Big(y-\tfrac{129}{2}\Big)\big| \le 32\Big),$$

$$f_{OC}(x,y,t) = \mathbb{1}\Big(\big|\cos\big(\theta(t)\big)\Big(x-\tfrac{129}{2}\Big)+\sin\big(\theta(t)\big)\Big(y-\tfrac{129}{2}\Big)\big| \le 32\Big)$$
$$\cdot\, \mathbb{1}\Big(\big|-\sin\big(\theta(t)\big)\Big(x-\tfrac{129}{2}\Big)+\cos\big(\theta(t)\big)\Big(y-\tfrac{129}{2}\Big)\big| \le 32\Big) \text{ where,}$$

$$\theta(t) = \tfrac{\pi}{40}t\,\mathbb{1}(t \le w) + \Big(\tfrac{\pi}{40}w + \tfrac{\pi}{20}(t-w)\Big)\mathbb{1}(t > w).$$

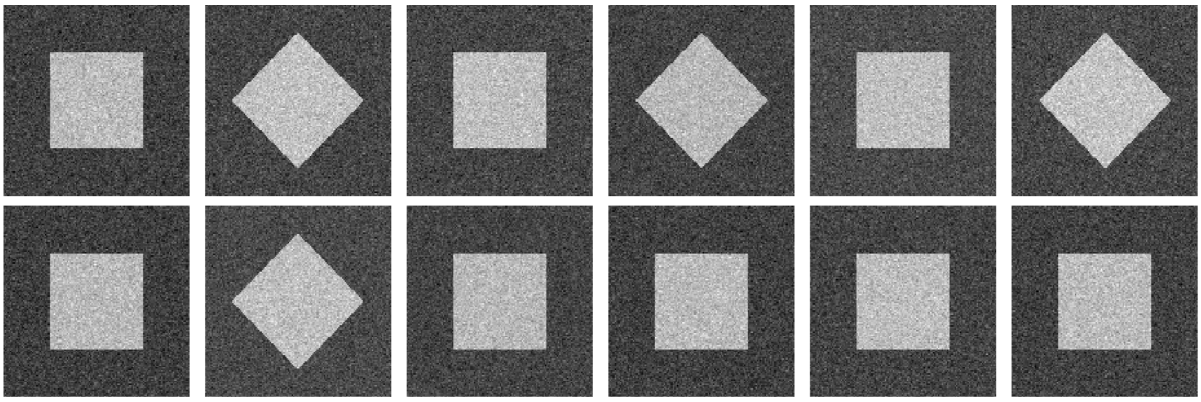


Figure 4.9: The top row shows the in-control images at time-points t = 0, 10, 20, 30, 40, and 50 for Simulation 7. The bottom row shows the corresponding images for the out-of-control image sequence.

The following graphs show the CUSUM statistics over time for each of the scenarios. These graphs correspond to one out-of-control image sequence. In Figure 4.10, time is plotted along the X-axis, the value of the CUSUM statistics is plotted along the Y-axis, and the vertical line represents the actual change point, i.e., $t = 20$. Note that in Simulation 5, the CUSUM chart starts to increase long after $t = 20$. This is because when the difference between the intensity values inside and outside the square is getting smaller, the proposed method finds it increasingly more difficult to identify the JLCs.

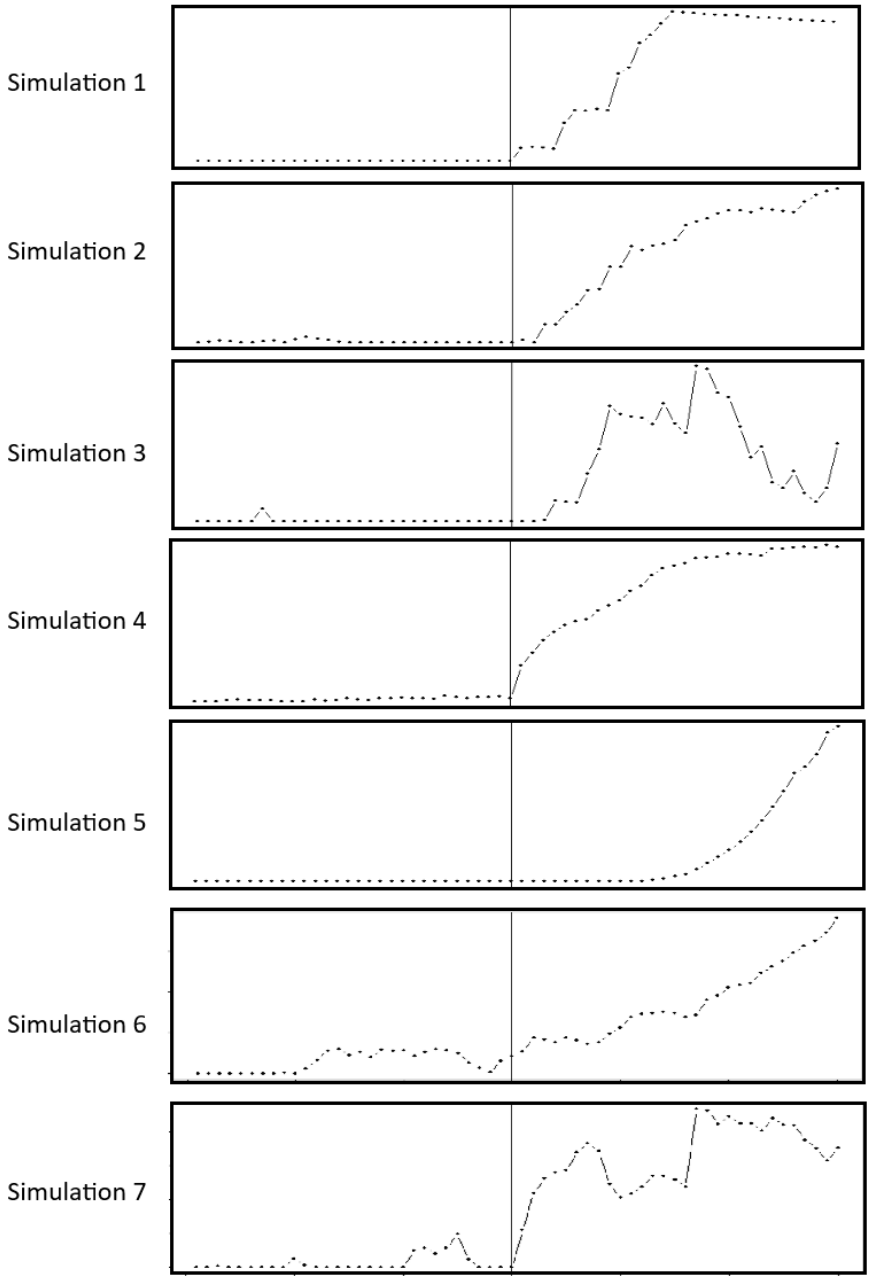


Figure 4.10: The proposed control chart corresponding to the seven simulation scenarios. The panels are ordered from top to bottom, representing Simulation 1 through Simulation 7. Time is plotted along the X-axis, and the CUSUM charting statistic is plotted along the Y-axis.

4.4.3 Comparison with state-of-the-art image monitoring methods

To investigate the numerical performance of the proposed method, we consider the following state-of-the-art image monitoring methods as competing methods.

(I) Supervised image monitoring algorithm by Bui and Apley (2018): It is an image monitoring and anomaly detection method in the framework of supervised learning. The central idea is to monitor the behavior of local residuals over time. To quantify deviations from the in-control state, Bui and Apley (2018) introduce a general spatial moving statistic (SMS) based on the sample A-D (Anderson and Darling) statistic Anderson and Darling (1954). For online monitoring, the charting statistic at the k-th time-point t_k is defined as:

$$S_{t_k} = \max_{(i,j)} \text{SMS}_{k,ij}, \tag{4.10}$$

where $\text{SMS}_{k,ij}$ denotes the spatial moving statistic at the (i,j)-th pixel at the k-th time-point. For a comprehensive discussion regarding this method, readers are referred to Bui and Apley (2018). It is important to note that this method is a Shewhart-type control chart. To implement the above procedure, we use *spc4sts* package Bui and Apley (2021) from CRAN-R.

(II) Wavelet-based image monitoring algorithm by Koosha et al. (2017): The wavelet-based method of image monitoring adopts a nonparametric profile monitoring framework where each image is decomposed into a set of 1-D profiles and monitored collectively using a GLR (generalized likelihood ratio) control chart. Koosha et al. (2017) refer to the wavelet coefficients as frequency-domain features, and use a suitable wavelet transformation to extract these features from the images. In this current context, each row of an $n \times n$ image is treated as a profile, and extracting d features per profile results in a wavelet coefficient vector of dimension $nd \times 1$ for each image. Then, the central idea of online monitoring is to construct a GLR-based control chart using the extracted $nd \times 1$ dimensional wavelet coefficient vectors.

We use the Average Run Length (ARL) and the standard deviations of the run lengths as measures of comparison of the concerned methods. For a test with 5% significance level, the ARL should be 20; and standard deviations of the run lengths should be close to 20 as well. In this chapter, we calculate these values empirically using 500 independent replications per simulation data, for each method. These values are presented in the Table 4.1. Each cell in this table displays the estimated out-of-control ARL along with the corresponding standard deviation of the run lengths in parentheses.

	Type	Proposed	Bui-Apley	Koosha-et.-al.
Simulation 1	(IC)	21.586 (21.190)	6.370 (0.499)	18.614 (8.059)
	(OC)	1.000 (0.000)	6.352 (0.478)	14.166 (5.410)
Simulation 2	(IC)	21.062 (19.870)	5.182 (12.728)	25.000 (11.427)
	(OC)	3.906 (1.190)	4.078 (9.113)	16.712 (6.381)
Simulation 3	(IC)	21.508 (19.944)	14.038 (14.265)	19.064 (8.833)
	(OC)	1.946 (0.860)	13.262 (12.76)	21.794 (24.768)
Simulation 4	(IC)	16.978 (22.200)	21.026 (20.810)	20.708 (9.208)
	(OC)	1.000 (0.000)	13.656 (7.288)	1.000 (0.000)
Simulation 5	(IC)	22.304 (24.140)	33.282 (5.252)	25.000 (1.985)
	(OC)	18.998 (16.431)	27.564 (3.577)	15.368 (1.046)

Table 4.1: Phase II performance comparisons on various simulation settings.

From Table 4.1, we observe that for all simulated cases, both the in-control ARL and the standard deviation of the run length behave as expected. We also observe that in the first out-of-control scenario, the proposed method performs very well, much better than its competitors. However, in the second scenario, the detection delay appears relatively large. This is partly due to the nature of the simulated drift: the image sequence does not change between time points 20 and 21, but rather between 21 and 22, making the minimum possible detection delay equal to 2. Additionally, the change in the drift pattern is very small, requiring slightly more than two time points for reliable detection. Similar behavior is observed in the third scenario. In the fourth scenario, we demonstrate that the proposed algorithm can detect a change in JLCs when there is no drift before the change. In the fourth scenario, the method by Koosha et al. (2017) perform similarly as well. Finally, in the fifth scenario, the in-control ARL and the out-of-control ARL do not differ significantly, which aligns with the expected behavior, because JLCs do not change in this scenario.

The method proposed by Bui and Apley (2018) appears to detect changes prematurely in most in-control scenarios, and it does not perform well in the out-of-control scenarios of Simulations 1-4. However, in the fifth simulation—where no drift is present—it behaves as expected. One of the major reasons for this kind of performance of this method is attributed to the fact that this method is a Shewhart-type control chart, and it is not designed to detect small incremental changes. However, this method can efficiently detect large pattern changes in images. For instance, pattern changes in the textured surfaces. The method developed by Koosha et al. (2017) exhibits similar limitations as the previous

method, although it performs well as expected in the fourth scenario. Therefore, these types of image monitoring methods are often not practically useful in the presence of small, gradual changes.

Performance of the proposed method under simulated rotation: Our theoretical justifications of the proposed method show that it is supposed to work when the movement of the JLCs is linear, but in short window widths and when the degree of rotation is not too large, we can approximate it with linear motion. For this reason, the proposed method works reasonably well in these types of scenarios as well. To demonstrate this behavior, we ran two different simulations. In Simulation 6, we rotated a single line separating two regions, colored black and white, at a constant, slow speed. In the out-of-control case, we double the rotation speed after a set time in Simulation 7, in a similar setup in which we rotate a black square on a white background. Corresponding simulation results are shown in Table 4.2.

Setting	Proposed	Bui-Apley	Koosha-et.-al.
Simulation 6	3.474 (2.314)	1.000 (0.000)	4.838 (10.612)
Simulation 7	3.114 (10.449)	14.266 (13.774)	5.750 (18.718)

Table 4.2: Phase II performance comparisons using out-of-control ARL.

4.4.4 Real data application

To illustrate the proposed ORT-based image monitoring methodology, we consider a multi-temporal sequence of satellite images covering the southwestern region of the Aral Sea. Further details about the dataset can be found in Section 4.1.1. We analyze the temporal pattern of the shrinking of the Aral Sea over a period of 2013 to 2024. Example images used for monitoring are presented in Figure 4.11.

Data preparation

In the real dataset, several scenes are compromised by heavy cloud cover and are therefore excluded from the analysis. Additionally, we also find missingness in the image data; certain stretches have no images at all, either due to the unavailability of the original

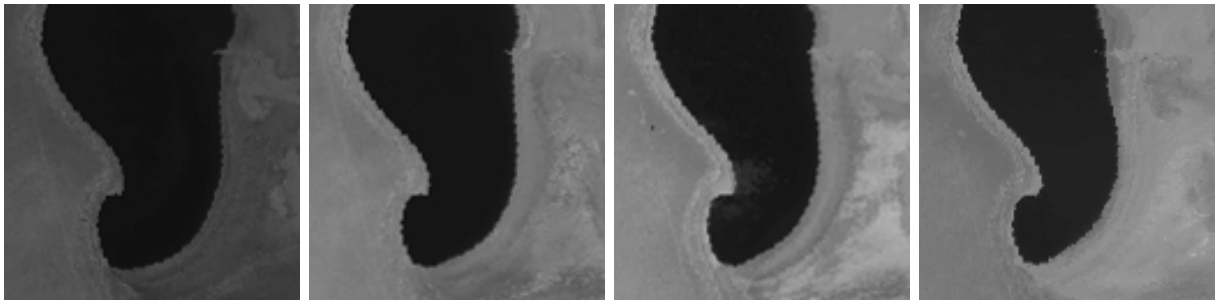


Figure 4.11: Sequence of the Aral Sea images used for real data analysis. Images are captured in 2013, 2015, 2017, and 2020. (Data source: U.S. Geological Survey.)

images or the removal of low-quality images. To minimize any impact on the proposed algorithm, we imputed these missing images by linearly interpolating the closest preceding and following images. We use the following equation to impute the missing images:

$$\widehat{w}(x,y,t) = \frac{1}{(t_{i+1}-t_i)}\Big[w(x,y,t_i)(t_{i+1}-t) + w(x,y,t_{i+1})(t-t_i)\Big], \text{ where } t \in (t_i, t_{i+1}).$$

By applying linear interpolation, we create an evenly spaced dataset covering the period from 2013 to 2024. We resize each image to the resolution 128×128. Subsequently, as a part of the pre-processing step, we perform image registration across the image dataset to remove any geometric misalignment by using *rNiftyreg* (Clayden et al., 2023) package from CRAN-R. For this analysis, we assume that the first 48 images, i.e., those from 2013 to 2016, are in-control. We set the window width to $m_0 = 24$, meaning that the preceding two years of data are used for prediction.

Parameter selection

Allowance parameter κ: To determine the value of the allowance parameter κ, we randomly select a few images from the first 48 data-points of the prepared dataset, and impute the missing images, i.e., the images not selected in the sample, by the linear interpolation method. Using this bootstrapped sample, we calculate the values of the CUSUM statistic without the allowance parameter at each of the months 25 through 48. We then run the proposed algorithm on each bootstrapped dataset with the allowance parameter set to 0, and compute the mean rate of change of the CUSUM statistic.

Suppose that $Q_{i,j}$ corresponds to month j in i-th bootstrap sample and we have a total of B bootstrap samples. We select

$$\widehat{\kappa} = \frac{1}{B}\sum_{i=1}^{B}\frac{1}{24}\sum_{j=25}^{48}\frac{Q_{i,j}}{j}$$

as the value of the allowance parameter κ.

Control limit q_0: We use the above value of the allowance parameter to apply the proposed algorithm on the bootstrapped samples again. Next, we compute the sample mean and sample standard deviation of the CUSUM statistic, denoted by $\overline{Q}$ and s_Q, respectively. Finally, we use $\left(\overline{Q}+\Phi^{-1}(0.95)s_Q\right)$ as the control limit q_0 for online drift pattern monitoring, where $\Phi^{-1}(0.95)$ is the 95-th percentile of the standard normal distribution.

Outcome of the analysis:

Using the procedure mentioned above, we construct the CUSUM chart displayed in Figure 4.12. In this chart, the time-point 0 represents June of 2015, and one unit on the X-axis represents a month. As the original data set starts in June of 2013, the first estimate of CUSUM statistics that we can get is delayed by the window width $m_0 = 24$ used in this algorithm. Figure 4.12 shows that there is a significant change in the drift pattern of the image sequence at the time-point 41 representing November, 2018. Although there is hardly any recent study on the Aral Sea shrinkage during 2013 to 2024, visual inspection suggests that a deceleration in the shrinking rate has started around the second half of 2018. Since the shrinkage of the Aral sea is a global concern, various global initiatives have been taken to mitigate the problem. We believe that additional ecological studies are required to confirm the observed deceleration in the shrinkage rate of the Aral Sea.

4.5 Concluding Remarks

In this chapter, we propose an efficient image monitoring scheme aimed at detecting shifts in the drift pattern of jump location curves (JLCs). A key innovation of the proposed approach lies in its decision tree-based framework for image surveillance, specifically em-

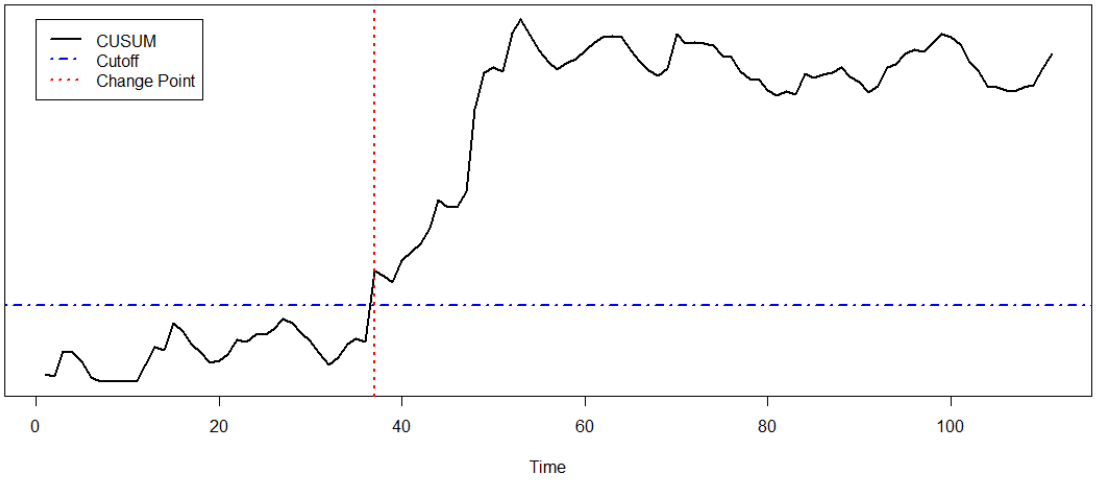


Figure 4.12: The proposed CUSUM chart for online drift pattern monitoring of the sequence of the Aral Sea images. The time-point 0 represents June of 2015, and one unit on the X-axis represents a month. The CUSUM charting statistic is plotted along the Y-axis.

ploying oblique-axis regression trees (ORTs). While JLC-based image monitoring has shown promise in various practical applications, existing methods often fail to account for drift and have limited ability to accommodate gradual changes across sequential images. This research work contributes to the image monitoring literature by introducing a JLC-based methodology that effectively captures and tracks temporal evolution patterns in image sequences. The proposed method demonstrates superior performance across a variety of settings, as evidenced by comprehensive numerical studies that include both drift and non-drift scenarios in the in-control image sequence. Several promising avenues exist for extending this line of research. One immediate direction involves the development of drift monitoring techniques that can operate effectively in the presence of spatio-temporally correlated noise. Additionally, in real-world applications, image registration is often required as a pre-processing step. Thus, a monitoring framework that can inherently handle geometric misalignment of image objects across images would be highly desirable. Finally, the flexibility of the ORT-based framework suggests its potential applicability to a broader class of process monitoring problems beyond the scope of image data.

Chapter 5

Concluding Remarks with Future Directions

In this dissertation, we present a comprehensive framework for image denoising and monitoring that addresses key challenges in preserving structural integrity while enhancing flexibility and predictive capability. The first major contribution is the development of a novel method that systematically combines multiple image denoising algorithms into a unified algorithmic structure. This integration is designed to compensate for the individual shortcomings of existing methods and to leverage their respective strengths, resulting in improved denoising performance across a wider variety of image types and noise conditions. The fusion mechanism is adaptive in nature and allows the algorithm to respond more effectively to local image characteristics, which is particularly beneficial in heterogeneous image regions.

We propose an innovative surface estimation technique based on oblique regression trees, which we tailor specifically for digital image data. Our tree-based global partitioning approach divides the image domain in a data-driven manner that respects local geometric features. This allows for accurate intensity estimation while maintaining the sharpness and integrity of edge structures, an essential requirement in applications such as medical imaging, remote sensing, and automated visual inspection.

We then extend the utility of our framework to predictive applications by developing a novel image monitoring scheme. This scheme is based on oblique regression trees

and integrates standard statistical process control techniques with the means for tracking gradual or abrupt drifts in the underlying edge structures of images over time. This dual capability allows the monitoring system to detect not only global changes in image intensity distributions but also more subtle structural variations that might signal meaningful shifts in the process generating the images. As a result, the proposed monitoring framework offers a more sensitive and interpretable tool for change detection in dynamic imaging environments.

5.1 Limitations And Future Scopes

In Chapter 2, we consider only two specific denoising schemes as part of our proposed framework. While this allows for a focused theoretical analysis, it also leaves room for improvement. By following the same line of reasoning used in the theoretical justification of the scheme, future work could incorporate a broader set of denoising algorithms, possibly in combination with more advanced edge detection techniques. This would allow the framework to benefit from a larger pool of methods, enhancing both its flexibility and practical performance.

In Chapter 3, we establish the theoretical consistency of the proposed method without providing explicit estimates for its convergence rate. Addressing this gap is a key direction for future research, as deriving convergence rate bounds would not only reinforce the theoretical foundation of the method itself but also enhance the validity of all subsequent methodologies that rely on it. Moreover, the surface estimation method presented in this chapter relies on approximating the underlying jump location curve (JLC) using straight line segments or, more generally, hyperplanes. While this assumption facilitates efficient computation and analytical tractability, it may limit the applicability of the method in settings where the discontinuities exhibit curved or nonlinear structures. Future research can aim to relax this assumption by incorporating more flexible, possibly nonparametric representations of the JLC, thereby extending the applicability of the method to a wider range of complex image geometries. Another significant issue with the proposed method is the splitting rules, as the underlying optimization problem is NP-complete (Hyafil and Rivest, 1976). However, for images with resolutions below 1024×1024, we did not observe

a significant computational bottleneck in practice. The computational cost becomes more pronounced at higher resolutions. One way to alleviate this issue is to initialize the optimization procedure with informed starting values. For example, Krylov subspace methods can be used to estimate the principal local gradient direction, which can then serve as an initial candidate for the splitting direction, followed by optimization over the remaining parameters. Our current implementation uses this approach for the monitoring method discussed in Chapter 4. Specifically, we compute local gradient estimates $\beta_y = \nabla I(y)$ at each pixel y using a general Sobel filter.

We then construct the matrix Σ_Γ over a tree-node Γ:

$$\Sigma_\Gamma = \sum_{y \in \Gamma} \beta_y \beta_y^T,$$

The dominant splitting direction is taken to be the leading eigenvector of Σ_Γ. This eigenvector is efficiently computed using Krylov subspace methods such as power iteration or the Lanczos method, avoiding the need for full eigendecomposition. Under the assumption that the region Γ has only one JLC, this should provide a good estimate of the perpendicular direction of the edge. Another practical strategy is to adopt a hybrid approach by using axis-aligned regression trees in the initial stages to reduce computational overhead, and then switching to oblique splits once the leaf nodes become sufficiently small. This provides a balance between computational efficiency and modeling flexibility.

The monitoring scheme developed in Chapter 4 inherits these limitations, as it is built upon the denoising framework discussed earlier. In particular, the absence of an estimated convergence rate prevents us from establishing theoretical bounds for the prediction error of the monitoring procedure. Thus, any progress towards addressing these theoretical shortcomings, especially regarding convergence rates and structural flexibility, would directly contribute to strengthening the analytical foundations of the monitoring scheme as well. From a practical perspective, the proposed monitoring procedure is particularly useful for images containing prominent JLCs, particularly when these JLCs can be reasonably approximated by straight lines. However, for some industrial image-monitoring applications, such as rolling-process monitoring or textured-surface inspection, the current version of the procedure may not be the most suitable choice. An extended version of the

current ORT that incorporates a flexible nonlinear splitting rule, or an ensemble-based version of the ORT, may provide a better fit for such applications.

In general, any advancement in the underlying regression surface estimation techniques, as discussed in this dissertation, is expected to improve the effectiveness of the image monitoring scheme as well.

5.2 Possible Directions for Future Research

Here are some potential future applications and extensions of the methods proposed and discussed in this dissertation:

Random Forests: The proposed ORT-based denoising technique assumes that the jump location curves (JLCs) lie on some hyperplane. This assumption is crucial, as the theoretical justification of our method relies heavily on the convexity of each leaf node obtained during tree construction. This limitation can be addressed by fitting multiple trees on the image data using bootstrapping. In doing so, the JLCs can be approximated by a combination of hyperplanes, potentially leading to more accurate estimation near discontinuities and, consequently, improved surface estimation.

Bayesian ORT: A natural extension of our method is to embed it within a Bayesian framework. By placing suitable priors on the splitting rules, we can construct a Markov Chain Monte Carlo (MCMC) sampler to draw trees from the posterior distribution over tree structures conditioned on the observed image. These splitting rules can be tailored to reflect real-world characteristics and to mitigate issues such as overfitting. For instance, one can introduce a depth-dependent penalty that assigns a higher probability to not splitting as the tree depth increases, thereby discouraging overly complex trees and improving robustness. Each sampled tree can be used to generate a denoised estimate of the image, and the final estimate is obtained by averaging over these individual estimates. This approach accounts for model uncertainty and can lead to improved smoothing performance by incorporating information from a diverse ensemble of trees. Moreover, while our current model assumes that JLCs lie on a linear structure, averaging over multiple sampled trees allows for greater flexibility in capturing non-linear features, thereby addressing one

of the key limitations of the original formulation.

Correlated Noise in the Monitoring Scheme: In the current work, we have assumed that the noise is independent across pixels, which is often unrealistic in practical settings. A natural extension would be to model correlated noise, which would make the method applicable to a broader range of real-world scenarios and potentially enhance its detection capabilities.

Denoising of Image Sequence in Presence of Image Misalignment: In our monitoring application in Chapter 4, we assume that the sequence of images in the Phase I stage is already aligned, or the image registrations are performed separately as a pre-processing. This leads to a two-step procedure. A promising future direction within the ORT framework would be to develop a method that can simultaneously address the image misalignment and denoising, thereby integrating an image registration framework in the ORT-based monitoring framework.

The scope of applications of the proposed ideas is certainly not limited to these directions mentioned above. One can explore the uses of the proposed ideas in various areas of data science and statistics.

Bibliography

Abdelhamed, A., Afifi, M., Timofte, R., Brown, M. S., Cao, Y., Zhang, Z., Zuo, W., Zhang, X., Liu, J., Chen, W., Wen, C., Liu, M., Lv, S., Zhang, Y., Pan, Z., Li, B., Xi, T., Fan, Y., Yu, X., Zhang, G., Liu, J., Han, J., Ding, E., Yu, S., Park, B., Jeong, J., Liu, S., Zong, Z., Nan, N., Li, C., Yang, Z., Bao, L., Wang, S., Bai, D., Lee, J., Kim, Y., Rho, K., Shin, C., Kim, S., Tang, P., Zhao, Y., Zhou, Y., Fan, Y., Huang, T., Li, Z., Shah, N. A., Liu, W., Yan, Q., Zhao, Y., Możejko, M., Latkowski, T., Treszczotko, L., Szafraniuk, M., Trojanowski, K., Wu, Y., Michelini, P. N., Hu, F., Lu, Y., Kim, S., Kim, W., Lee, J., Choi, J.-H., Zhussip, M., Khassenov, A., Kim, J. H., Cho, H., Kansal, P., Nathan, S., Ye, Z., Lu, X., Wu, Y., Yang, J., Cao, Y., Tang, S., Cao, Y., Maggioni, M., Marras, I., Tanay, T., Slabaugh, G., Yan, Y., Kang, M., Choi, H.-S., Song, K., Xu, S., Lu, X., Wang, T., Lei, C., Liu, B., Gupta, R., and Kumar, V. (2020). Ntire 2020 challenge on real image denoising: Dataset, methods and results. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2077–2088.

Anderson, T. W. and Darling, D. A. (1954). A test of goodness of fit. *Journal of the American Statistical Association*, 49(268):765–769.

Basak, S. and Mukherjee, P. S. (2025). An efficient image denoising method integrating multi-resolution local clustering and adaptive smoothing. *Statistical Methods & Applications*, 34(4):767–785.

Basak, S., Roy, A., and Mukherjee, P. S. (2026+). Estimation of piecewise continuous regression function in finite dimension using oblique-axis regression tree with applications in image denoising. *Statistica Sinica*, 0(0):DOI: 10.5705/ss.202025.0120.

Bowman, A. W. (2006). Comparing nonparametric surfaces. *Statistical Modelling*, 6(4):279–299.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. (The Wadsworth statistics / probability series). Wadsworth International Group.

Buades, A., Coll, B., and Morel, J.-M. (2005). A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65. Ieee.

Buades, A., Coll, B., and Morel, J.-M. (2010). Image denoising methods. a new nonlocal principle. *SIAM review*, 52(1):113–147.

Buades, A., Coll, B., and Morel, J.-M. (2011). Non-local means denoising. *Image Processing On Line*, 1:208–212.

Bui, A. T. and Apley, D. W. (2018). A monitoring and diagnostic approach for stochastic textured surfaces. *Technometrics*, 60(1):1–13.

Bui, A. T. and Apley, D. W. (2021). spc4sts: Statistical process control for stochastic textured surfaces in r. *Journal of Quality Technology*, 53(3):219–242.

Burger, H. C., Schuler, C. J., and Harmeling, S. (2012). Image denoising: Can plain neural networks compete with bm3d? *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2392–2399.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698.

Cattaneo, M. D., Chandak, R., and Klusowski, J. M. (2024). Convergence rates of oblique regression trees for flexible function libraries. *The Annals of Statistics*, 52(2):466–490.

Chambolle, A. and Pock, T. (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145.

Chaudhuri, A. and Chatterjee, S. (2023). A cross-validation framework for signal denoising with applications to trend filtering, dyadic cart and beyond. *The Annals of Statistics*, 51(4):1534–1560.

Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., and Xu, C. (2021). Pre-trained image processing transformer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12299–12310.

Chu, C., Glad, I., Godtliebsen, F., and Marron, J. S. (1998). Edge-preserving smoothers for image processing. *Journal of the American Statistical Association*, 93(442):526–541.

Clayden, J., Modat, M., Presles, B., Anthopoulos, T., and Daga, P. (2023). *RNiftyReg: Image Registration Using the 'NiftyReg' Library*. R package version 2.8.1.

Criminisi, A., Robertson, D., Konukoglu, E., Shotton, J., Pathak, S., White, S., and Siddiqui, K. (2013). Regression forests for efficient anatomy detection and localization in computed tomography scans. *Medical Image Analysis*, 17(8):1293–1303. Epub 2013 Jan 27.

Das, S. and Mukherjee, P. S. (2024). Image registration for zooming using similarity matching. *Journal of Data Science*, 22(4):558–574.

Das, S., Roy, A., and Mukherjee, P. S. (2024). Image registration for zooming: A statistically consistent local feature mapping approach. *Stat*, 13(1):e664.

Dette, H. and Neumeyer, N. (2001). Nonparametric analysis of covariance. *The Annals of Statistics*, 29(5):1361–1400.

Dong, W., Shi, G., and Li, X. (2013). Nonlocal image restoration with bilateral variance estimation: A low-rank approach. *IEEE Transactions on Image Processing*, 22(2):700–711.

Donoho, D. L. (1997). Cart and best-ortho-basis: a connection. *The Annals of Statistics*, 25(5):1870–1911.

Fang, X., Paynabar, K., and Gebraeel, N. (2019). Image-based prognostics using penalized tensor regression. *Technometrics*, 61(3):369–384.

Felsberg, M., Forssen, P.-E., and Scharr, H. (2006). Channel smoothing: efficient robust smoothing of low-level signal features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):209–222.

Feng, L. and Qiu, P. (2018). Difference detection between two images for image monitoring. *Technometrics*, 60(3):345–359.

Figueiredo, M. A. and Nowak, R. D. (2003). An em algorithm for wavelet-based image restoration. *IEEE Transactions on Image Processing*, 12(8):906–916.

Fitzpatrick, M. C., Preisser, E. L., Porter, A., Elkinton, J., Waller, L. A., Carlin, B. P., and Ellison, A. M. (2010). Ecological boundary detection using bayesian areal wombling. *Ecology*, 91(12):3448–3455.

Foi, A., Katkovnik, V., and Egiazarian, K. (2007). Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE Transactions on Image Processing*, 16(5):1395–1411.

Franken, E. and Duits, R. (2009). Crossing-preserving coherence-enhancing diffusion on invertible orientation scores. *International Journal of Computer Vision*, 85(3):253–278.

Froment, J. (2014). Parameter-free fast pixelwise non-local means denoising. *Image Processing On Line*, 4:300–326.

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

Ghosh, M. (2021). Exponential tail bounds for chisquared random variables. *Journal of Statistical Theory and Practice*, 15(2):35.

Gijbels, I., Lambert, A., and Qiu, P. (2006). Edge-preserving image denoising and estimation of discontinuous surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1075–1087.

Godtliebsen, F. and Sebastiani, G. (1994). Statistical methods for noisy images with discontinuities. *Journal of Applied Statistics*, 21:459–477.

Gonzalez, R. C. and Woods, R. E. (2019). *Digital Image Processing*. Pearson Education.

Gramacy, R. B. and Lee, H. K. H. (2008). Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130.

Guhaniyogi, R. (2017). Bayesian nonparametric areal wombling for small-scale maps with an application to urinary bladder cancer data from connecticut. *Statistics in Medicine*, 36(25):4007–4027.

Halder, A., Banerjee, S., and Dey, D. K. (2024). Bayesian modeling with spatial curvature processes. *Journal of the American Statistical Association*, 119(546):1155–1167.

Hall, P. and Hart, J. D. (1990). Bootstrap test for difference between means in nonparametric regression. *Journal of the American Statistical Association*, 85(412):1039–1049.

He, Z., Zuo, L., Zhang, M., and Megahed, F. M. (2016). An image-based multivariate generalized likelihood ratio control chart for detecting and diagnosing multiple faults in manufactured products. *International Journal of Production Research*, 54(6):1771–1784.

Hyafil, L. and Rivest, R. L. (1976). Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15–17.

Kang, Y. (2020). Consistent blind image deblurring using jump-preserving extrapolation. *Journal of Computational and Graphical Statistics*, 29(2):372–382.

Kang, Y. (2022). Statistical quality control using image intelligence: A sparse learning approach. *Naval Research Logistics (NRL)*, 69(7):996–1008.

Kang, Y. (2023). *DRIP: Discontinuous Regression and Image Processing*. R package version 1.8.

Kang, Y., Mukherjee, P. S., and Qiu, P. (2018). Efficient blind image deblurring using nonparametric regression and local pixel clustering. *Technometrics*, 60(4):522–531.

Kang, Y., Roy, A., and Mukherjee, P. S. (2026). Adaptive blind image deblurring and denoising. *Scandinavian Journal of Statistics*, 53(1):413–441.

Kang, Y., Shi, Y., Jiao, Y., Li, W., and Xiang, D. (2021). Fitting jump additive models. *Computational Statistics & Data Analysis*, 162:107266.

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

King, E., Hart, J. D., and Wehrly, T. E. (1991). Testing the equality of two regression curves using linear smoothers. *Statistics & Probability Letters*, 12(3):239–247.

Konomi, B. A., Sang, H., and Mallick, B. K. (2014). Adaptive bayesian nonstationary modeling for large spatial datasets using covariance approximations. *Journal of Computational and Graphical Statistics*, 23(3):802–829.

Kontschieder, P., Fiterau, M., Criminisi, A., and Bulo, S. R. (2015). Deep neural decision forests. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Koosha, M., Noorossana, R., and Megahed, F. (2017). Statistical process monitoring via image data using wavelets. *Quality and Reliability Engineering International*, 33(8):2059–2073.

Kulasekera, K. (1995). Comparison of regression curves using quasi-residuals. *Journal of the American Statistical Association*, 90(431):1085–1093.

Lindeberg, T. (1998). Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision*, 30:117–156.

Liu, Y.-L., Wang, J., Chen, X., Guo, Y.-W., and Peng, Q.-S. (2008). A robust and fast non-local means algorithm for image denoising. *Journal of Computer Science and Technology*, 23(2):270–279.

Mao, X., Shen, C., and Yang, Y.-B. (2016). Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2802–2810.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217.

Megahed, F. M., Wells, L. J., Camelio, J. A., and Woodall, W. H. (2012). A spatiotemporal method for the monitoring of image data. *Quality and Reliability Engineering International*, 28(8):967–980.

Megahed, F. M., Woodall, W. H., and Camelio, J. A. (2011). A review and perspective on control charting with image data. *Journal of Quality Technology*, 43(2):83–98.

Mukherjee, P. S. (2017). A multi-resolution and adaptive 3-d image denoising framework with applications in medical imaging. *Signal, Image and Video Processing*, 11(7):1379–1387.

Mukherjee, P. S. and Qiu, P. (2011). 3-d image denoising by local smoothing and nonparametric regression. *Technometrics*, 53(2):196–208.

Mukherjee, P. S. and Qiu, P. (2015). Image denoising by a local clustering framework. *Journal of Computational and Graphical Statistics*, 24(1):254–273.

Okhrin, Y., Schmid, W., and Semeniuk, I. (2020). New approaches for monitoring image data. *IEEE Transactions on Image Processing*, 30:921–933.

Perona, P. and Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639.

Portilla, J., Strela, V., Wainwright, M., and Simoncelli, E. (2003). Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351.

Qiu, P. (1997). Nonparametric estimation of jump surface. *Sankhya, Series A*, 59(2):268–294.

Qiu, P. (1998). Discontinuous regression surfaces fitting. *The Annals of Statistics*, 26(6):2218–2245.

Qiu, P. (2004). The local piecewise linear kernel smoothing procedure for fitting jump regression surface. *Technometrics*, 46(1):87–98.

Qiu, P. (2005). *Image processing and jump regression analysis*. John Wiley & Sons.

Qiu, P. (2007). Jump surface estimation, edge detection, and image restoration. *Journal of the American Statistical Association*, 102(478):745–756.

Qiu, P. (2009). Jump-preserving surface reconstruction from noisy data. *Annals of the Institute of Statistical Mathematics*, 61(3):715–751.

Qiu, P. (2013). *Introduction to statistical process control*. CRC press.

Qiu, P. (2018). Jump regression, image processing, and quality control. *Quality Engineering*, 30(1):137–153.

Qiu, P. (2020). Big data? statistical process control can help! *The American Statistician*, 74(4):329–344.

Qiu, P. and Kang, Y. (2015). Blind image deblurring using jump regression analysis. *Statistica Sinica*, pages 879–899.

Qiu, P. and Mukherjee, P. S. (2010). Edge structure preserving image denoising. *Signal Processing*, 90(10):2851–2862.

Qiu, P. and Xing, C. (2013a). Feature based image registration using non-degenerate pixels. *Signal Processing*, 93(4):706–720.

Qiu, P. and Xing, C. (2013b). On nonparametric image registration. *Technometrics*, 55(2):174–188.

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

Roy, A. and Mukherjee, P. S. (2024a). A control chart for monitoring images using jump location curves. *Quality Engineering*, 36(2):439–452.

Roy, A. and Mukherjee, P. S. (2024b). Image comparison based on local pixel clustering. *Technometrics*, 66(4):495–506.

Roy, A. and Mukherjee, P. S. (2025). Upper quantile-based cusum-type control chart for detecting small changes in image data. *Journal of Applied Statistics*, 52(11):2156–2171.

Roy, A. and Mukherjee, P. S. (2026). A registration-free approach for image monitoring. *Statistics and Computing*, 36(20):1–20.

Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268.

Su, Y., Shu, L., and Tsui, K.-L. (2011). Adaptive ewma procedures for monitoring processes subject to linear drifts. *Computational statistics & data analysis*, 55(10):2819–2829.

Takeda, H. (2007). *Kernel Regression-Based Image Processing Toolbox for MATLAB*.

Takeda, H., Farsiu, S., and Milanfar, P. (2007). Kernel regression for image processing and reconstruction. *IEEE Transactions on Image Processing*, 16(2):349–366.

Wang, X.-F. and Ye, D. (2010). On nonparametric comparison of images and regression surfaces. *Journal of Statistical Planning and Inference*, 140(10):2875–2884.

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

Xing, C. and Qiu, P. (2011). Intensity-based image registration by nonparametric local smoothing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2081–2092.

Yan, H., Paynabar, K., and Shi, J. (2017). Anomaly detection in images with smooth background via smooth-sparse decomposition. *Technometrics*, 59(1):102–114.

Yang, W., Grasso, M., Colosimo, B. M., and Paynabar, K. (2023). A tensor-based hierarchical process monitoring approach for anomaly detection in additive manufacturing. *Quality and Reliability Engineering International*, 39(2):630–650.

Yi, F. and Qiu, P. (2022). An adaptive cusum chart for drift detection. *Quality and Reliability Engineering International*, 38(2):887–894.

Yi, F. and Qiu, P. (2023). Water resource surveillance for the salton sea in california by adaptive sequential monitoring of its landsat images. *Journal of Agricultural, Biological and Environmental Statistics*, pages 1–15.

You, K. (2021). *tvR: Total Variation Regularization*. R package version 0.3.2.

Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., and Yang, M.-H. (2022). Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5728–5739.

Zhan, H., Liu, Y., and Xia, Y. (2026). Consistency of oblique decision tree and its boosting and random forest. *Bernoulli*, 32(1):417–441.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155.

Zhen, Z., Paynabar, K., and Shi, J. (2023). Image-based feedback control using tensor analysis. *Technometrics*, 65(3):305–314.