High-Order Sparsity Exploiting Methods with Applications in Imaging and PDEs

by

Dennis Denker

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2016 by the
Graduate Supervisory Committee:

Anne Gelb, Chair
Richard Archibald
Dieter Armbruster
Albert Boggess
Rodrigo Platte
Toby Sanders

ARIZONA STATE UNIVERSITY

May 2016

ABSTRACT

High-order methods are known for their accuracy and computational performance when applied to solving partial differential equations and have widespread use in representing images compactly. Nonetheless, high-order methods have difficulty representing functions containing discontinuities or functions having slow spectral decay in the chosen basis. Certain sensing techniques such as MRI and SAR provide data in terms of Fourier coefficients, and thus prescribe a natural high-order basis. The field of compressed sensing has introduced a set of techniques based on ℓ^1 regularization that promote sparsity and facilitate working with functions having discontinuities. In this dissertation, high-order methods and ℓ^1 regularization are used to address three problems: reconstructing piecewise smooth functions from sparse and and noisy Fourier data, recovering edge locations in piecewise smooth functions from sparse and noisy Fourier data, and reducing time-stepping constraints when numerically solving certain time-dependent hyperbolic partial differential equations.

DEDICATION

To my parents, for their unending support and love

ACKNOWLEDGMENTS

I would like to thank Prof. Anne Gelb for her dedication to my success in completing this work and for helping with my development as a mathematician. I am deeply grateful to her.

I would like to thank Prof. Rodrigo Platte and Prof. Douglas Cochran for their advice over the years, as various ideas developed.

I would like to thank the other committee members Dr. Rick Archibald, Prof. Dieter Armbruster, Prof. Al Boggess, and Dr. Toby Sanders for the time they have dedicated to seeing this work through. Their comments were extremely valuable in developing these ideas.

I would like to thank Albert Leffler and the Leffler family for their constant support and help in reviewing this paper.

I would like to thank Cara Gerard for her support and motivation.

I would like to thank my friends Andee, Candy, Debbie, Jorly, Laura, Paloma, Scott, and Sean for supporting this endeavor from the beginning.

I would like to thank Joan Matuska for inspiring me.

Finally, I would like to thank Jingjing Fan, Evelyn Karis, and Ryan Mead. Helping you with your preliminary investigation, gave me insight into many of the details of these problems. Your energy and enthusiasm was inspirational.

This work is supported in part by grants NSF-DMS 1216559, AFOSR FA9550-12-1-0393, NSF-DMS 1521600, NSF-DMS 1502640, and AFOSR FA9550-15-1-0152.

TABLE OF CONTENTS

Page

LIST OF TABLES vii

LIST OF FIGURES ix

1 INTRODUCTION 1

2 FUNCTION RECOVERY THROUGH REGULARIZATION AND WAVELET REPROJECTION 4

2.1 Introduction 4

2.1.1 Reconstruction from Non-uniform Fourier Data 5

2.1.2 Reconstruction Using Regularization 6

2.1.3 Promoting Edge Sparsity and Piecewise Smooth Solutions through Regularization of Solutions Represented in a Wavelet Basis 8

2.2 Design of Regularization Based on Wavelet Reprojection 9

2.3 The Choice of Wavelet Basis 11

2.4 Sample One-dimensional Results 14

2.5 Two-dimensional Implementation 17

2.6 Kronecker Product Representation 20

2.7 The Vectorized Split Bregman Implementation 21

2.8 A Memory Efficient Implementation 22

2.9 Two Dimensional Results 23

2.9.1 Filtering 35

2.10 Analysis 43

2.10.1 Edge Response 44

2.10.2 Gibbs Oscillation Suppression and Implicit Filtering 45

2.10.3 Edge Detection Response for High Order Functions 48

CHAPTER Page

2.10.4 Concluding Remarks 48

3 EDGE DETECTION OF PIECEWISE SMOOTH FUNCTIONS FROM UNDER-SAMPLED FOURIER DATA 51

3.1 Preliminaries 53

3.1.1 The Concentration Factor Edge Detection Method 55

3.1.2 Sparsity Promoting Regularization 57

3.1.3 Polynomial Annihilation Edge Detection As a Sparsifying Operator 59

3.1.4 Sample Sets 61

3.2 Edge Detection for Under-sampled Fourier Data 62

3.2.1 The Concentration Factor Edge Detection Method 62

3.2.2 The Sparsity Enforcing Edge Detection Method 64

3.2.3 Generating an Edge Map Using Thresholding 65

3.2.4 Variance of Jump Function Responses 67

3.2.5 Post Processing Edge Detection Using Variance 70

3.2.6 Determining Edges from Regularized Reconstruction 74

3.3 Determining the Two-dimensional Edge Map Using Variance 77

3.3.1 Variance of Jump Function Responses 82

3.3.2 Determining Edges by Regularized Reconstruction 83

3.4 Concluding Remarks 89

4 AN ADAPTIVE FOURIER FILTER FOR RELAXING TIME STEPPING CONSTRAINTS FOR EXPLICIT SOLVERS 92

4.1 Background 93

4.2 The Proposed Filter 94

CHAPTER Page

4.3 Numerical Results 96
4.4 Conclusion 102
5 CONCLUSION 103
REFERENCES 105
APPENDIX
A THE WAVELET BASIS 109
A.1 Motivation 110
A.2 The Discrete Wavelet Basis 111
A.2.1 Multi-resolution Analysis 111
A.2.2 Wavelet Basis Formation 112
A.3 The Daubechies Wavelets 112
B THE SPLIT BREGMAN ALGORITHM 115
B.1 Introduction 116
B.2 Splitting 116
B.2.1 The Bregman Distance 117
B.2.2 The Iterative Solution 118
B.3 Adding Feedback to Simplify the Iteration 121
B.4 Solving the Sub-problems 123
B.4.1 The Fidelity Term 123
B.4.2 The Basis Pursuit Problem 123
C PERMISSION TO USE PUBLISHED PAPERS 127

LIST OF TABLES

Table Page

2.1 Parameters Selected for Each Test Case. Unless Otherwise Specified, We also Choose $N = 128$. 16

2.2 Parameters Selected for Each Two-dimensional Test Case. Unless otherwise Specified We Let $N = 128$, $\zeta = \frac{1}{40}$, and the Gaussian Sampling Pattern Will Be Used. 28

2.3 Fraction of Cells in which the Reconstruction Accuracy of $\mathbf{F}_{wav}$ Is Better Than That of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. 38

2.4 Sum of log_{10} Ratio of Reconstruction Errors with $\mathbf{F}_{wav}$ Compared to $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. 38

2.5 Fraction of Cells in which the Reconstruction Accuracy of $\mathbf{F}_{wav}$ is Better Than That of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ Away From Edges. 39

2.6 Fraction of Cells in which the Reconstruction Accuracy is Improved by Filtering for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ 40

2.7 Fraction of Cells in which the Reconstruction Accuracy is Improved by Filtering for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ in Smooth Regions. 41

2.8 Parameters Selected for Each Two-dimensional Spiral Sampling Test Case. 41

2.9 Fraction of Cells in which the Reconstruction Accuracy Is Improved by Using Spiral Sampling for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ 41

2.10 Fraction of Cells in which the Reconstruction Accuracy Is Improved by Using Spiral Sampling in Smooth Regions for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. 42

3.1 Sample Concentration Factors 56

3.2 Parameters Selected for Each Test Case. Unless otherwise Specified, We also Choose $N = 64$, $Q = 30$ and $\zeta = .5$. 62

Table Page

4.1 Parameter Values and Results at High Resolution. 97

4.2 Parameter Values and Results at Low Resolution. 99

4.3 Comparison of Results Using Different Resolutions at Filter Power, $p = 12$. 101

4.4 Comparison of Results Using Different Resolutions at Filter Power, $p = 8$. 102

LIST OF FIGURES

Figure Page

2.1 The Location and Scale of the Wavelet Basis Vectors and the Wavelet Coefficients Associated with Decompositions of the Monomials of Increasing Order. 14

2.2 B_{high} Wavelet Components for D2...D5 When Approximating a Discontinuity. 15

2.3 The Projection of f_1 onto the B_{low} Basis and the log_{10} Pointwise Error in That Projection with $N = 128$. 15

2.4 $E_2(f_1)$ and $E_3(f_1)$ with $N = 128$. 15

2.5 f_1,$\mathcal{F}^{-1}\hat{\mathbf{V}}$, $\vec{f}_{wav}$, and $\vec{f}_{nowav}$, on the Domain $[-4., .6]$ to Show Detail. 18

2.6 The Absolute Pointwise Reconstruction Errors in $\vec{f}_{wav}$ and $\vec{f}_{nowav}$. 19

2.7 $\vec{Q}$ for Each Test Case. 20

2.8 The Test Function, f_2, Shown As a Surface Plot, a Contour Plot, and a One-dimensional Cross-section along the Line $y = \frac{1}{8}$. 24

2.9 K-space Sampling Distributions with $\gamma = 0.3$. 25

2.10 The Fourier Reconstruction of f_2 Associated with the Test Cases. 27

2.11 Test Case 1 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$. 28

2.12 Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 1. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate. 29

2.13 The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 1 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$. 30

2.14 Test Case 2 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$. 30

2.15 Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 2. White Areas Indicate $\mathbf{F}_{wav}$ is More Accurate. 31

Figure Page

2.16 The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 2 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$. 32

2.17 Test Case 3 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$. ... 33

2.18 Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 3. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate. 33

2.19 The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 3 and log_{10} Reconstruction Errors Along the Cross-section $y = \frac{1}{8}$. 34

2.20 Test Case 4 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$. ... 35

2.21 Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 4. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate. 35

2.22 The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 4 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$. 36

2.23 Comparison of Surface Reconstructions for Test Case 4. 37

2.24 Filter Powers of Various Orders 39

2.25 The Effect of a Fourth Order Filter on the Fourier Reconstruction without Regularization Applied to Test Case 2. 40

2.26 Comparison of Radial Sampling Distributions for Gaussian and Spiral Sampling Patterns. 42

2.27 The 2-norm of the Reconstruction Errors as a Function of $log_{10}\frac{\mu}{\lambda}$ for Test Case 2. 43

2.28 The 2-norm of the Reconstruction Errors as a Function of $log_{10}\frac{\mu}{\lambda}$ for Test Case 4. 43

2.29 Artifacts Present in the $\mathbf{F}_{wav}$ Reconstruction. 44

Figure Page

2.30 The B_{low} Reconstruction of $r(x)$ as Compared to the B_{low} Reconstruction of $r(x-\frac{2}{N})$ and the Associated Edge Detector Responses. 45
2.31 The Suppression of Gibbs Phenomenon As a Consequence of Projecting $r(x)$ onto the B_{low} Basis. 47
2.32 A Comparison of the Raised Cosine Filter and the Wavelet Projection.. 47
2.33 The Effect of the B_{low} Projection on High-order Smooth Functions..... 49
3.1 $f_1(x)$. 55
3.2 Sample Jump Response Behavior for Various Concentration Factors with $N = 32$. 58
3.3 Concentration Factor Edge Detection Method, $S_N^{\sigma_G}[f_1](x)$. 63
3.4 $S_N^{MM}[f_1](x)$ in (3.21)..... 64
3.5 Edge Map of $f_1(x)$ Resulting from (3.22). 65
3.6 Algorithm 1 Using $S_N^{\sigma_G}[f_1](x)$, $S_N^{MM}[f_1](x)$, and $S_N^{SE}[f_1](x)$ and Threshold $c = 7/8$. 66
3.7 $\vec{v}(\mathbf{E_1})(x_j)$ in (3.23) for $r(x_j)$ in (3.8), $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24a) Is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$..... 69
3.8 $\vec{v}(\mathbf{E_1})(x_j)$ in (3.23) for $f_1(x)$ and $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24a) Is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$..... 69
3.9 $\vec{v}(\mathbf{E_2})(x_j)$ in (3.23) for $f_1(x)$ and $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24b) is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$..... 70
3.10 Edge Map Generated Using Algorithm 3 for $f_1(x)$. Here $\delta = 5$, $c = \frac{7}{8}$... 71
3.11 Sample Reconstructions and Pointwise Reconstruction Errors from Applying (3.25) to the Test Cases in Table 3.2. 75
3.12 $\vec{v}(\mathbf{Q})$ for f_1 Using the Parameters in Table 3.2..... 76

Figure Page

3.13 Algorithms 2 and 3 Applied to $\vec{v}(\mathbf{Q})$ for f_1 Using the Parameters in Table 3.2. Here $\delta = 5$ and $c = \frac{7}{8}$. 77

3.14 $f_2(x, y)$. 78

3.15 $S_{C,N}^{\sigma_G}[\mathbf{F_2}]_{m,n}$ and a Cross-section at $S_{C,N}^{\sigma_G}[\mathbf{F_2}]_{m,N+1}$. 80

3.16 The Results of Applying Algorithm 4 to $S_{C,N}^{\sigma_G}[\mathbf{F_2}]_{m,n}$ with Thresholding at $c = 15/16$. 81

3.17 $\mathbf{V}(\mathbf{E})_{m,n}$ Using $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$ and a Cross-section $\mathbf{V}(\mathbf{E})_{m,N+1}$. . . 82

3.18 The Results of Applying Algorithm 5 to $\mathbf{V}(\mathbf{E})_{m,n}$ Using $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$, Maximum Distance between Peaks, $\delta = 7$, Threshold $c = 15/16$. 83

3.19 $\mathbf{V}(\mathbf{Q})_{m,n}$ Using $Q = 40$ and a Cross-section $\mathbf{V}(\mathbf{Q})_{m,N+1}$. 84

3.20 The Results of Applying Algorithm 5 to $\mathbf{V}(\mathbf{Q})_{m,n}$ with $Q = 30$, Maximum Distance between Peaks, $\delta = 7$, threshold $c = 15/16$. 84

3.21 Comparison of Algorithm 5 Results Using N=128, SNR=12.5dB, $\gamma =$ .75, $\beta = 0.3$, $\delta = 7$, Threshold $c = 31/32$. 85

3.22 Comparison of Algorithm 5 Results Using N=128, SNR=12.5dB, $\gamma = \frac{9}{16}$, $\beta = 0.09$, $\delta = 7$, Threshold $c = 31/32$. 86

3.23 Comparison of Algorithm 5 Results Using N=128, SNR=8dB, $\gamma = \frac{1}{5}$, $\beta = 0.09$ $\delta = 7$, Threshold $c = 31/32$. 87

3.24 Comparison of Methods Using Algorithm 4 on $S_{C,N}^{\sigma_G}[\mathbf{F_2}]$ and $S_{C,N}^{MM}[\mathbf{F_2}]$ and Algorithm 5 on $\mathbf{V}(\mathbf{E})$ and $\mathbf{V}(\mathbf{Q})$ for the Test Cases in Table 3.2 and Figures 3.21, 3.22, and 3.23. 88

4.1 The Effect of Filtering on Growth Factors. 94

4.2 The Effect of the Adaptive Filter on Accuracy Using the Parameters in Table 4.1. 98

Figure Page

4.3 The Effective Solution Support for Example 1 as a Function of Time. . . 99
4.4 The Effect of the Adaptive Filter on Solution Accuracy Using the Parameters in Table 4.2. 100
B.1 The Subdifferentials at Differentiable and Non-differentiable Points in One Dimension. 119
B.2 A Simple Basis Pursuit Problem Showing the Combined Functional As Well As the 1-norm and 2-norm Components. 124
B.3 When $|2\lambda bc| > 1$ and $bc > 0$, the Minimum Occurs in the Differentiable Region $0 < u < \infty$. 125
B.4 When $|2\lambda bc| < 1$, the Minimum Occurs at a Point Lacking Differentiability. 126

Chapter 1

INTRODUCTION

High-order methods are known for their accuracy and computational performance when applied to solving partial differential equations. They are also in widespread use in representing images compactly. Also, certain sensing techniques such as MRI and SAR provide data in terms of Fourier coefficients, which naturally fit with spectral methods. The success of these methods is predicated on the ability of the associated basis to represent the function being approximated with few basis elements or with a spectrum of basis elements having rapidly decaying coefficients. Many real-world applications work with data that is best represented by functions having discontinuities. The expression of such functions in the Fourier basis or in bases formed by orthogonal polynomials results in an infinite number of basis elements, with slow coefficient decay. Additionally, when spectral methods are applied to time-dependent, non-linear, partial differential equations, solutions initially having a compact representation in the basis can develop shock or a "spectral cascade" that requires an ever increasing number of basis elements for accurate representation. In this dissertation, investigations are made regarding the application of high-order methods and sparsity-driven regularization to problems having discontinuities or numerical solutions resulting in a large spectrum in the chosen basis.

Chapter 2 contains an investigation of a technique based on sparsity-promoting regularization in combination with a wavelet basis as a means to reconstruct functions from incomplete or noisy Fourier data. The wavelets are constructed to have compact support, both in the physical and frequency domains. The wavelets also have properties that allow for a local representation of low-order polynomials with few ba-

sis elements. This combination of properties would seem to enhance regularizations that reconstruct piecewise smooth functions. In this chapter, a new regularization is developed that produces solutions formed from a wavelet basis as opposed to the Euclidean basis. The accuracy of the resulting solutions is investigated both in smooth regions and near edges under a variety of test cases. This chapter also provides a detailed discussion of the inherent issues associated with Fourier measurements made from real-world objects having a piecewise smooth representation and the process of applying sparsity-promoting regularization techniques. These discussions provide some of the necessary background for Chapter 3.

Chapter 3 investigates recovering edge information from incomplete or noisy Fourier data using new techniques based on analyzing the variance in results, when samples are subject to a set of treatments. Recovering edge information is of equal practical importance to reconstructing functions. Concentration factor based techniques are extremely successful at recovering edge information, but these methods can often detect additional false edges. Some strategies have been developed to combat this issue, but in the presence of noise or sub-sampling, false edge detections persist. In this chapter, a new strategy is developed that uses regularized function reconstructions to help eliminate these false edges.

In Chapter 4, techniques used in ℓ^1 regularization are applied to numerical solutions of partial differential equations. Explicit time-stepping methods place restrictions on time step size in order to maintain stability. One stabilization technique is to apply filtering, which can often lead to diffusive behavior. Some time-dependent partial differential equations have variable spectral support or slow spectral growth. Time step restrictions that are necessary when the numerical solution has a full spectrum are also applied when spectral support is small. This is necessary to avoid instabilities induced by numerical noise. This chapter investigates applying regularization-inspired

techniques when spectral support is small to avoid either a restrictive time-step size or diffusive filtering.

Concluding remarks are provided in Chapter 5.

Chapter 2

FUNCTION RECOVERY THROUGH REGULARIZATION AND WAVELET REPROJECTION

2.1 Introduction

Several important applications, such as MRI and SAR acquire data by way of Fourier sampling, Yan (2002); Cheney and Borden (2009); Richards *et al.* (2010). A common feature of these data acquisition methods is the imposition of basis functions over physical space. The resulting complex measurements represent integrals of the product of these basis functions with a density function. For all practical purposes, these measurements are true integrals and not discrete sums. Additionally, real-world scenes contain abrupt transitions from region to region, so the associated response density function has discontinuities.

Another common feature of these applications is that data elements may be sub-sampled or collected at non-uniform frequencies. In the case of MRI, the process of imposing the basis function involves activating a magnetic field gradient for a certain period of time. To achieve coverage of basis functions, the gradient fields are often switched on and off. To avoid peripheral nerve stimulation as a consequence of Faraday's law, the rate of magnetic field change must be restricted. Thus the acquisition of a complete set of data on a uniform grid is necessarily slow. Newer MRI imaging methods reduce the number of samples taken and the number of abrupt magnetic gradient changes to accelerate the imaging process, Li *et al.* (2015); Pipe (1999a). The resulting samples tend to have a higher density at low frequencies, but are more sparse in the higher frequencies. In the case of SAR, the frequency sampling

pattern is partially determined by the physical trajectory of the transmitter-receiver pair, Gor (2010). Additionally the trajectory has limited physical extent, and at certain times the transceiver may be re-tasked leading to gaps in the data. Another cause of missing data is interference from external radio frequency transmissions.

Noise originates from a variety of sources. Both MRI and SAR suffer from electromagnetic interactions between elements in the field of view such as reflections, shading, and field inhomogeneities from non-uniform permeability and permittivity. SAR is very dependent on accurate determination of relative antenna positions, so noise can be introduced by physical vibrations. SAR also has the disadvantage of an uncontrollable environment between the transceiver and scene. Of course, both methods are subject to electronic noise in the imaging apparatus.

2.1.1 Reconstruction from Non-uniform Fourier Data

As stated in Section 2.1, the collected data can be represented by the continuous Fourier transform of a discontinuous density function, f. We assume the field of view is the square $[-1,1] \times [-1,1]$, so an individual Fourier measurement is given by

$$\hat{f}_{\omega_x,\omega_y} = \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} f\,(x,y)\, e^{-i\pi(\omega_x x+\omega_y y)} dydx. \tag{2.1}$$

The frequencies ω_x and ω_y need not be integers. It is well known that when f is discontinuous any approximation to f constructed from a finite number of Fourier terms will be subject to the Gibbs phenomenon and the accuracy of the approximation will be reduced to first order, Hesthaven *et al.* (2007). Further, even in the absence of discontinuities the truncated Fourier series

$$S_N f\,(x,y) = \sum_{k_x=-N_x}^{N_x}\sum_{k_y=-N_y}^{N_y} \hat{f}_{k_x,k_y} e^{i\pi(k_x x+k_y y)}; \quad k_x,k_y \in \mathbb{Z} \tag{2.2}$$

has limited resolving power.

Practical measurements, $\left\{\hat{f}_{(\omega_x,\omega_y)_j}\right\}$, can occur on a set of non-integer frequencies $T = \left\{(\omega_x,\omega_y)_j\right\}$, with each frequency pair being associated with the function, $\phi_j = e^{i\pi\left(\omega_{x_j}x+\omega_{y_j}y\right)}$. These functions will not in general be orthogonal. Direct methods for reconstructing the density function from non-uniform data, such as convolutional gridding, can be poorly conditioned, especially as the sampling pattern deviates substantially from the uniform grid. Because of this, a different approach is applied that instead uses the prior information that the density function is piecewise smooth, and we seek candidate solutions such that numerical Fourier measurements approximate the provided measurements, $\left\{\hat{f}_{(\omega_x,\omega_y)_j}\right\}$.

2.1.2 Reconstruction Using Regularization

Regularization is a well-known technique for solving ill-conditioned inverse problems. Regularizations using ℓ^1 penalty terms have enjoyed great success in the fields of image processing and compressed sensing. A driving principle in many of these techniques is that some attribute of a good solution will be sparse. Total variation denoising is a popular physical space[1] technique for reconstructing smooth images from noisy ones, Rudin *et al.* (1992); Osher *et al.* (2005). The denoising formulation employs the regularization

$$\vec{u}_{clean} = \underset{\vec{u}}{\operatorname{argmin}} \frac{1}{2}\left\|\vec{u}_{original} - \vec{u}\right\|_2^2 + \lambda TV\left(\vec{u}\right), \;\; TV\left(u\right) = \sum_j |\vec{u}_{j+1} - \vec{u}_j|. \tag{2.3}$$

In this case $\vec{u}$ represents a physical-space image. The total variation, TV in (2.3), is an ℓ^1 term that penalizes oscillations at the cost of favoring piecewise constant solutions, producing the so-called "staircase" artifact[2] in the recovered image. This

[1]Here physical space refers to data derived from spatial as opposed to frequency measurements, thus there is no requirement for a Fourier transform.

[2]The "staircase" artifact is the effect of approximating a smooth function with a piecewise constant solution. This leads to abrupt transitions from constant piece to constant piece.

regularization was extended to one involving a Generalized Total Variation, which seeks piecewise polynomials, but the higher order variation quickly becomes complicated, Bredies *et al.* (2010).

As stated above, many regularizations assume that some feature of a solution is sparse and an ℓ^1 term is created to promote that sparsity. In compressed sensing ℓ^1 terms are used as convex relaxations for ℓ^0 terms, Candès *et al.* (2006). An ideal ℓ^0 penalty term would seek to minimize the count of non-zero entries in its argument, but ℓ^0 terms are not convex and a general numerical solution is impossible. In Lustig *et al.* (2007) the authors seek sparsity in the wavelet coefficients using a penalty term involving the wavelet transform leading to the formulation

$$\vec{u}_{new} = \underset{\vec{u}}{\operatorname{argmin}} \frac{1}{2} \left\| \vec{u}_{original} - \vec{u} \right\|_2^2 + \alpha TV\left(\vec{u}\right) + \lambda \left\| \mathcal{W}\vec{u} \right\|_1,$$

where $\vec{u}$ represents a physical-space image and $\mathcal{W}$ is the wavelet transform. When functions cannot be locally represented by low order polynomials, the wavelet sparsity ratio exceeds 50% for certain wavelet families, such as the Daubechies wavelets with two vanishing moments. In these cases numerical tests show the sparsity promoting term conflicts with the fidelity term, introducing significant bias error into the regularization.

An important concern is that physical space techniques do not address some of the fundamental issues described in Section 2.1.1 and numerical tests of these techniques, when applied to Fourier data often commit the so-called "inverse crime"[3], Guerquin-Kern *et al.* (2012). Additionally, when using non-uniform Fourier data, physical space regularizations are subject to the ill-conditioning associated with the pre-processing

[3]Real-world measurements come from a close approximation to the continuous Fourier transform, (2.1), of the density function. This is not equivalent to sampling in physical space and applying the discrete Fourier transform, which instead produces Fourier coefficients associated with a band-limited projection of the true density function.

step of converting the Fourier data directly to a uniform physical grid.

In Archibald *et al.* (2015), the regularization addresses these issues directly by using a fidelity term that involves the forward Fourier transform and a high-order penalty term that seeks edge sparsity, while also allowing for piecewise polynomial solutions. A simplified formulation[4] is given by

$$\vec{u}_{new} = \underset{\vec{u}}{\operatorname{argmin}} \left\| \mathbf{GF}\vec{u} - \vec{\hat{f}}_g \right\|_2^2 + \lambda \|\mathbf{E}_m \vec{u}\|_1. \tag{2.4}$$

Here $\vec{u}$ is defined on a uniform physical-space grid, $\mathbf{F}$ is the discrete Fourier transform, $\mathbf{G}$ is a matrix that selects Fourier coefficients that match the frequencies in the collected data, $\hat{f}_g$, and $\mathbf{E}_m$ is a high-order polynomial annihilation edge detector,Wasserman *et al.* (2015); Stefan *et al.* (2010). Observe that the fidelity term in (2.4) still does not address the mismatch between the discrete Fourier transform, $\mathbf{F}$, and the measurements derived from the continuous Fourier transform, but the penalty term assists in reducing the impact of Gibbs phenomenon.

2.1.3 Promoting Edge Sparsity and Piecewise Smooth Solutions through Regularization of Solutions Represented in a Wavelet Basis

Applying (2.4) to Fourier data results in rapid and accurate convergence in smooth regions even in the presence of noise or with incomplete sample sets. The method also succeeds in eliminating Gibbs oscillations away from edges, but the method still has first-order accuracy near edges. This chapter investigates representing $\vec{u}$ in (2.4) in terms of a basis that is better adapted at representing piecewise functions. In (2.4), the solution, $\vec{u}$, is represented in the Euclidean basis, which is well suited to representing edges. Nonetheless, the Euclidean basis treats all points in isolation.

[4]In the simplified formulation, the assumption is made that Fourier coefficients are collected with sub-sampling, but still at integer frequencies so the discrete Fourier transform as opposed to the NUFFT is used.

Thus, (2.4) relies solely on the fidelity term to promote smoothness between edges. This chapter introduces a new technique based on using a wavelet subspace for the representation of $\vec{u}$. The wavelets are designed to seek a compromise that allows for some global information to be incorporated into the basis, while still being able to represent edges locally. The new method does not seek sparsity in terms of wavelet coefficients as in Lustig *et al.* (2007), but instead uses the wavelet subspace as a constraint on admissible solutions.

The rest of this chapter is organized as follows: In Section 2.2, a new regularization that includes wavelet reprojection is designed. The regularization is developed into a two-dimensional algorithm in Section 2.5. Preliminary results are demonstrated in Section 2.9. Additionally, subjective observations regarding the results and the testing methodology will be discussed. Section 2.10 examines the underlying effects that result from the addition of the wavelet basis.

2.2 Design of Regularization Based on Wavelet Reprojection

Before describing the modifications made to (2.4), we briefly provide some definitions and notation that will be useful throughout this chapter. We start by defining $\mathcal{W}$ as the discrete wavelet transform associated with some wavelet basis, B, so that

$$\vec{c} = \mathcal{W}\vec{f}.$$

In this formulation, physical space has an N point discretization[5], so $\vec{f} \in \mathbb{R}^N$ is some discrete signal and $\vec{c} \in \mathbb{R}^N$ are the associated wavelet coefficients. The inverse wavelet transform, $\mathcal{W}^{-1}_{N\times N}$, is a unitary matrix. For economy of expression throughout this chapter, we will let B_{low} be the subset of B that contains the $\frac{N}{2}$ elements of B that are least localized spatially. Conversely, B_{high} will be defined to be the $\frac{N}{2}$ most localized

[5]To maximize compatibility with the wavelets, we choose N to be a power of two.

wavelet elements of B. The vector, $\vec{c}_{low}$, is the set of wavelet coefficients associated with the elements of B_{low} in the solution.

The modification to the regularization in (2.4) in one dimension is given by

$$\vec{c}_{low} = \underset{\vec{c}}{\operatorname{argmin}} \left\| \mathcal{G}\mathcal{W}^{-1}\mathcal{A}\left(\vec{c}, \vec{0}\right) - \mathcal{S}\vec{\hat{g}} \right\|_2^2 + \lambda \left\| \mathbf{E}_m \mathcal{W}^{-1}\mathcal{A}\left(\vec{c}, \vec{0}\right) \right\|_1. \tag{2.5}$$

The reconstructed solution is then given by

$$\vec{f} = \mathcal{W}^{-1}\mathcal{A}\left(\vec{c}_{low}, \vec{0}\right). \tag{2.6}$$

Here $\vec{c}$ is a vector of wavelet coefficients associated with the elements of B_{low} and the M provided Fourier samples, acquired by measurement of the true function are given by $\vec{\hat{g}}_{M\times 1}$. In practice, we will assume M to be significantly smaller than N and the associated sampling frequencies need not be integers. The operator $\mathcal{G}$ uses the numerical solution to approximate the Fourier coefficients matching the elements of $\vec{\hat{g}}$.[6] The augmentation operator, $\mathcal{A}(\vec{u}, \vec{v})$ has the effect of stacking the column vectors $\vec{u}$ and $\vec{v}$ to form a new column vector. Therefore $\mathcal{A}\left(\vec{c}_{low}, \vec{0}_{\left(\frac{N}{2}\right)\times 1}\right)$ pads $\vec{c}_{low}$ to form an $N \times 1$ vector.[7] The operator $\mathcal{S}$ is added to allow for preprocessing steps to be applied to the provided measurements.[8]

To highlight edges and suppress smooth regions, the polynomial annihilation technique, Archibald *et al.* (2005); Stefan *et al.* (2010); Wasserman *et al.* (2015); Archibald *et al.* (2015), is used to form an m^{th}-order edge detection matrix, $\mathbf{E}_m$. This edge detector uses a stencil of points to approximate the local polynomial series expansion

[6]Here the forward Fourier transform, $\mathcal{G}$, can be the matrix pair $\mathbf{GF}$ as in (2.4) or an implementation of the NUFFT restricted to the sampled frequencies.

[7]This generalized form of a padding operator is chosen to allow for future refinements of the regularization by potentially including elements of B_{high} in the reconstruction.

[8]In Archibald *et al.* (2015) an exponential filter was used to deal with noise and to help mitigate some of the consequences of the Gibbs phenomenon.

of the subject function, and terms of order less than m are eliminated. This results in low-order smooth regions having a vanishing response from the edge detector. Conversely, edges are approximated locally by high-order polynomials in the discrete setting, and thus result in a non-zero response from the edge detector. A detailed explanation of this method can be found in Section 3.1.3. The count of edges is assumed to be small and therefore the regularization uses the ℓ^1 norm of the detected edges to promote sparsity. The regularization parameter, λ, is chosen to appropriately weight the regularization term.

In summary, the technique operates by constructing solutions from a wavelet subspace such that Fourier coefficients generated from these solutions approximate the provided Fourier measurements, in the ℓ^2 sense, while promoting edge sparsity based on the response of a high-order edge detector.

2.3 The Choice of Wavelet Basis

As stated in Section 2.1.3, the goal of representing the solution in a wavelet basis is to promote piecewise smooth solutions with good edge resolution. We use the Daubechies wavelets as the wavelet basis in (2.5) based on their vanishing moments and compact support. A detailed explanation of the construction of the Daubechies wavelets is given in Appendix A.

Wavelet basis elements are characterized by their scale and location. It is convenient to induce an ordering on the basis. We let $B = \{B_j\}$ be the set of wavelet basis vectors ordered from least localized to most localized. When two basis vectors have equal locality the vectors will be ordered with centers from left to right. Thus using the notation given in Section 2.2, $B_{low} = \{B_j\}, j = 1, ..., \frac{N}{2}$, and $B_{high} = \{B_j\}, j = \frac{N}{2} + 1, ..., N$. Each basis in the Daubechies family of wavelets is characterized by the number of vanishing moments. For example, the D2 wavelets have two vanishing

moments[9]. Thus, if we let $\vec{\psi}$ be a member of the D2 basis with support in the interior of the domain and let $\vec{f}$ be a discrete signal of the form $f_j = a + bj, \quad j \in \mathbb{Z}, a, b \in \mathbb{R}$, then $\left\langle \vec{f}, \vec{\psi} \right\rangle = 0$. Another important property of the Daubechies wavelets is that for signals sampled from polynomials that are of low order, but still greater than the number of vanishing moments, inner products with the members of B_{high} are small, e.g. for the D2 wavelets the coefficients associated with the monomial x^2 in the domain $[-1, 1)$ are of the order 10^{-4}.

Figure 2.1 provides a visual representation of the D2 wavelets and shows the wavelet transform of monomials of various orders. The rows of the left half of Figure 2.1 represent the basis elements of the D2 wavelets. The first row contains the scaling function, subsequent rows have wavelets of increasing locality. For each group having the same locality, the wavelets are shown with their positions in the domain going from left to right. The right half of the figure shows inner products of these wavelets with different monomials in each column, starting with $f(x) = 1$ and going to $f(x) = x^5$. We see that the only non-vanishing inner product for the monomial, $f(x) = 1$, is the scaling function in the first row. For $f(x) = x$, the scaling function has a non-zero inner product, as well as the wavelets that have support intersecting the domain boundary, because $f(x) = x$ is not periodic. High-order monomials result in similar boundary coefficients, but also have non-zero inner products in the interior due to the D2 wavelets having only two vanishing moments; but for low-order monomials, these coefficients are small.

The members of the B_{high} subset of a Daubechies basis having P vanishing moments will have physical support $2P$, as discussed in Appendix A. This leads to a tradeoff between element support size and regularity. Consider the wavelet transform

[9]There is some inconsistency in the literature regarding the naming of the Daubechies wavelets. We use DN where N is the number of vanishing moments.

of a function consisting of piecewise polynomials of maximum order P. The wavelet coefficients will be zero except for those elements with support intersecting a discontinuity. Thus, as shown in Figure 2.2, increasing P allows for higher order polynomials in smooth regions at the expense of more wavelets being impacted by discontinuities.

In (2.5), we limit the representation of the solution to the span of B_{low}, as using the full wavelet basis in the discrete setting is identical to using the Euclidean basis, i.e., $\text{span}\left(\{\vec{e}_1, \ldots, \vec{e}_N\}\right) = \text{span}\left(B\right)$. Only when using the truncated basis does the basis promote smoothness, but using the truncated basis comes at the cost of not being able to represent edges with the resolution of the Euclidean basis. Numerical tests indicate that the $D2$ family of wavelets achieves a good compromise between locality and approximation accuracy. To demonstrate this we consider the test function

$$f_1\left(x\right) = \begin{cases} \frac{3}{2} & -\frac{3}{4} \leqslant x < -\frac{1}{2} \\ \frac{7}{4} - \frac{\pi x}{2} + \sin\left(\pi x - \frac{1}{4}\right) & -\frac{1}{4} \leqslant x < \frac{1}{8} \\ \frac{11\pi x}{4} - 5 & \frac{3}{8} \leqslant x < \frac{3}{4} \\ 0 & otherwise. \end{cases} \tag{2.7}$$

Figure 2.3 demonstrates the result of projecting (2.7) unto B_{low} of $D2$. We see that the truncated wavelet basis approximates $f(x)$ in the regions $\left(-\frac{3}{4}, -\frac{1}{2}\right)$ and $\left(\frac{3}{8}, \frac{3}{4}\right)$ at machine accuracy, but has more difficulty in the region $\left(-\frac{1}{4}, \frac{1}{8}\right)$ which is consistent with Figure 2.1. Nonetheless, we will see that the accuracy there is sufficient to allow for the reconstruction of smooth regions using (2.5). Additionally, Figure 2.3 demonstrates how the edges of (2.7) are approximated in the truncated basis. For comparison, Figure 2.4 show the results of applying the polynomial annihilation edge detectors of orders 2 and 3 to (2.7).

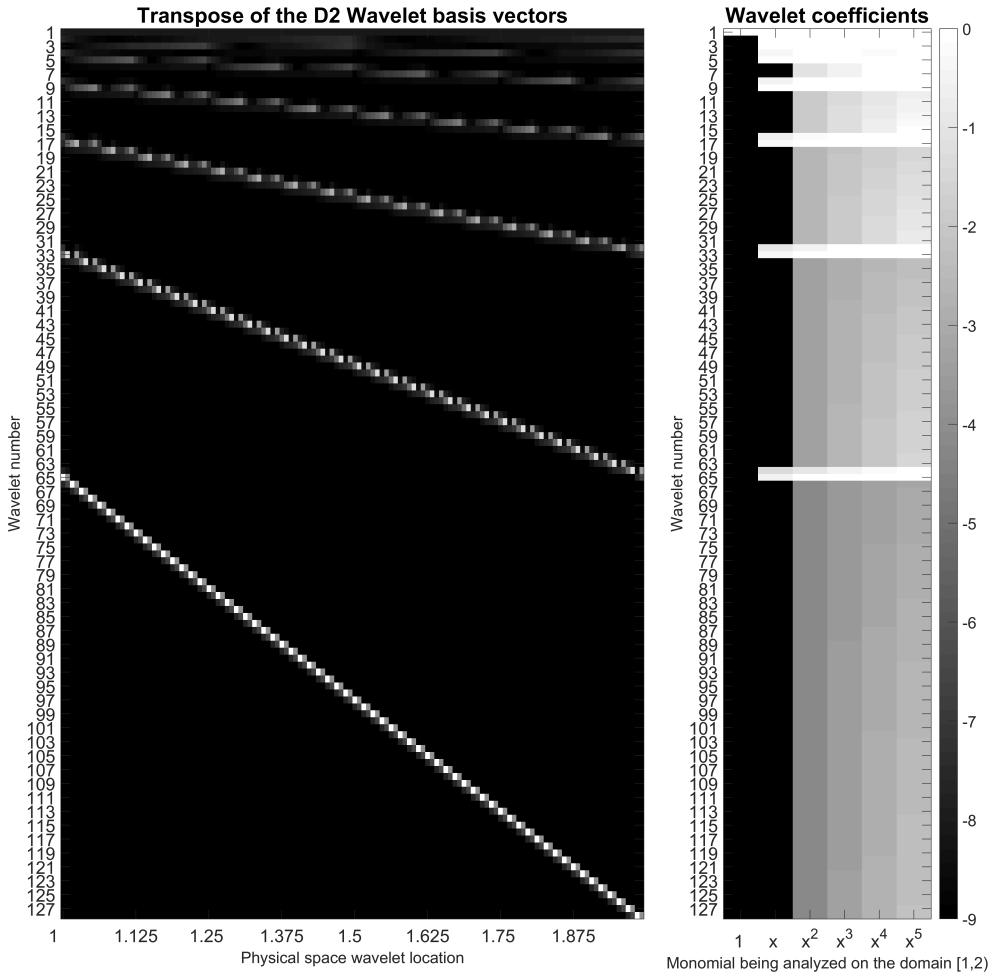


Figure 2.1: The Location and Scale of the Wavelet Basis Vectors and the Wavelet Coefficients Associated with Decompositions of the Monomials of Increasing Order.

2.4 Sample One-dimensional Results

In Fan and Mead (2015), this technique was explored in one dimension with the finding that the addition of the wavelet reprojection contributes to the robustness of the regularization with respect to the selection of the regularization parameter. It was also found that with the selection of the $D2$ wavelets, regularization results were similar for both the E_2 and E_3 polynomial annihilation edge detectors. Also, the results confirmed those found in Archibald *et al.* (2015), that the selection a polynomial annihilation edge detector of order 2 in (2.4) produces accuracy in smooth

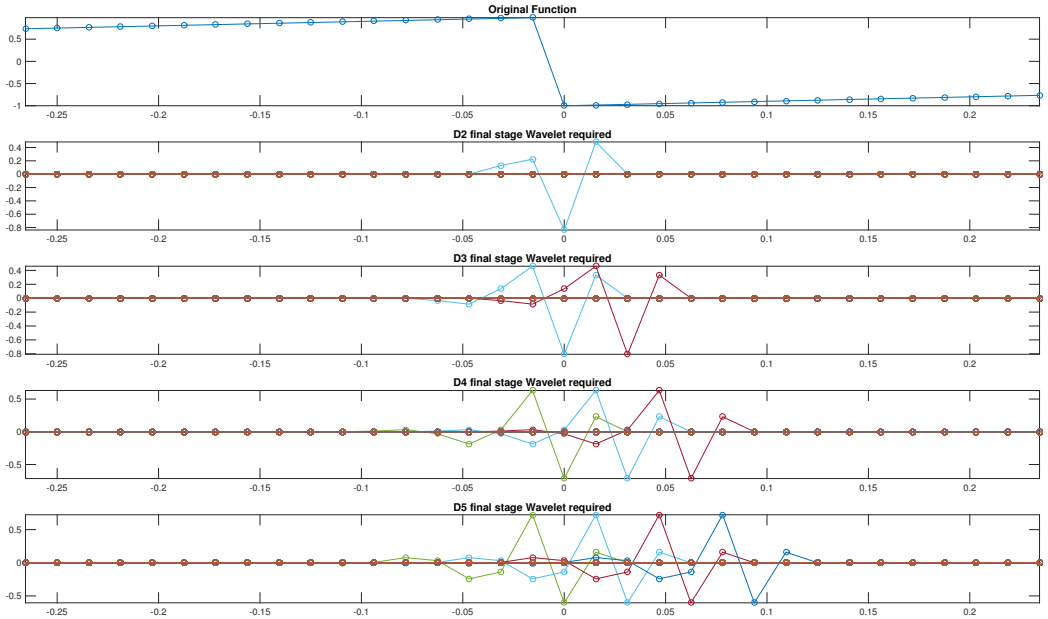
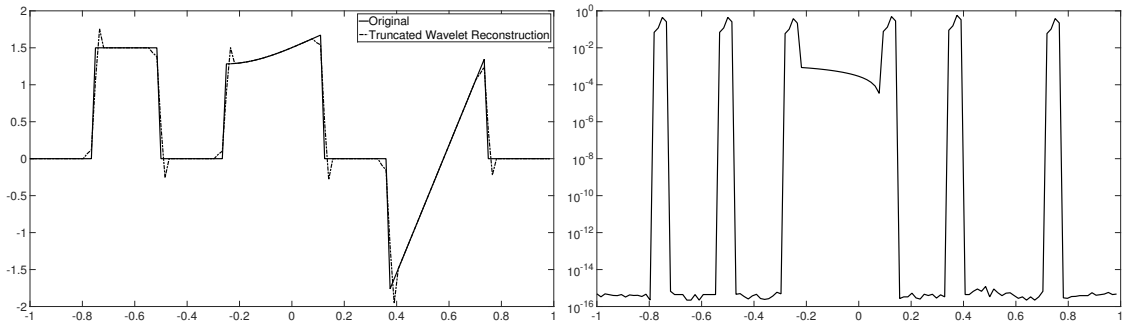


Figure 2.2: B_{high} Wavelet Components for D2...D5 When Approximating a Discontinuity.



(a) The projection of f_1 onto the B_{low} basis. (b) The pointwise error in the projection.

Figure 2.3: The Projection of f_1 onto the B_{low} Basis and the log_{10} Pointwise Error in That Projection with $N = 128$.

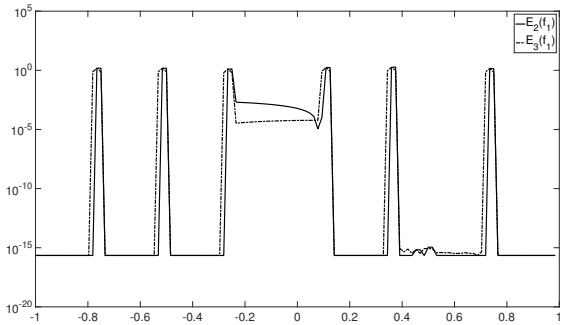


Figure 2.4: $E_2(f_1)$ and $E_3(f_1)$ with $N = 128$.

regions while keeping edge artifacts compact. Finally, the addition of the wavelet reprojection step increases accuracy in smooth regions in the presence of noise or subsampling.

We develop four test cases to demonstrate of (2.5) in one dimension. We let N be the number of physical space grid points. Each test case is characterized by two parameters. We let SNR represent the signal to noise ratio calculated as, $SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}$, with SNR $= \infty$ representing no noise. The set of sampled coefficients, $\hat{\mathbf{V}}$, consists of two subsets, $\hat{\mathbf{V}} = \hat{\mathbf{F}} \cup \hat{\mathbf{R}}$, and is controlled by the parameter $\gamma \in (0, 1]$, where $dim(\hat{\mathbf{V}}) = \gamma \cdot N, \quad \hat{\mathbf{F}} = \left\{ \hat{f}_k : -\frac{N}{16} \leq k \leq \frac{N}{16} \right\}$[10], and $\hat{\mathbf{R}}$ is a set of randomly selected coefficients from a normal distribution. The regularization parameters, $\lambda = 0.015$ for (2.5), and $\lambda = 0.08$ for (2.4) were used as determined in Fan and Mead (2015). These parameters were chosen by trial and error using various levels of noise and sub-sampling with the selected test function.

Table 2.1: Parameters Selected for Each Test Case. Unless Otherwise Specified, We also Choose $N = 128$.

Test Case	SNR	γ
1	∞	1
2	∞	0.4
3	13dB	1
4	13dB	0.4

We will apply (2.5) to a set of test cases with parameters specified in Table 2.1. We let $\vec{f}_{wav}$ represent the reconstruction formed by the application of (2.5) to $\hat{\mathbf{V}}$ and we let $\vec{f}_{nowav}$ represent the reconstruction formed by the application of (2.4)

[10]The existence of set $\hat{\mathbf{F}}$ ensures that the central frequencies that determine the basic shape of the result are included. This is consistent with the real-world sampling methods discussed in Section 2.1

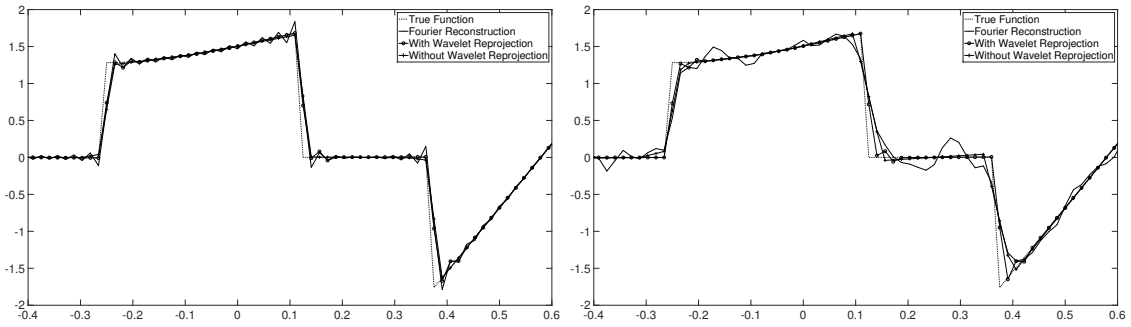
with $m = 2$ to $\hat{\mathbf{V}}$. Figure 2.5 compares the reconstructions formed by direct Fourier inversion, $\vec{f}_{wav}$, and $\vec{f}_{nowav}$. Figure 2.6 shows the absolute point-wise reconstruction errors associated with each method. To better compare the methods, we define the point-wise error ratio as

$$\vec{Q}_j = \log_{10}\left(\frac{\left|\vec{f}_{nowav_j} - \vec{f}_{1_j}\right| + \varepsilon}{\left|\vec{f}_{wav_j} - \vec{f}_{1_j}\right| + \varepsilon}\right).$$

Figure 2.7 shows $\vec{Q}$ for each of the test cases. In this figure, positive values indicate that (2.5) is more accurate than (2.4). These results indicate that in the presence of noise or with sub-sampling the wavelet reprojection method, (2.5), is in general more accurate than the regularization excluding the wavelet reprojection, (2.4). Nonetheless, the results are highly dependent on the test function chosen and the proper choice of the regularization parameter, λ; and these results do not include the application of a spectral filter in (2.4) as was done in Archibald *et al.* (2015).

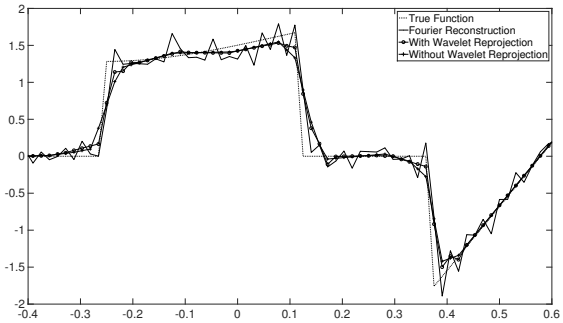
2.5 Two-dimensional Implementation

In one dimension, (2.5) and (2.4) can be solved with general purpose optimization packages, but using these solvers for two-dimensional problems with realistic values for N is prohibitively slow while requiring excessive system resources. We choose instead to use the Split-Bregman algorithm described in Goldstein and Osher (2009) to solve these optimization problems as was done in Archibald *et al.* (2015). A detailed explanation of the Algorithm can be found in Appendix B. Here we discuss the reformulation of (2.5) in two dimensions required to make it compatible with the Split-Bregman framework.

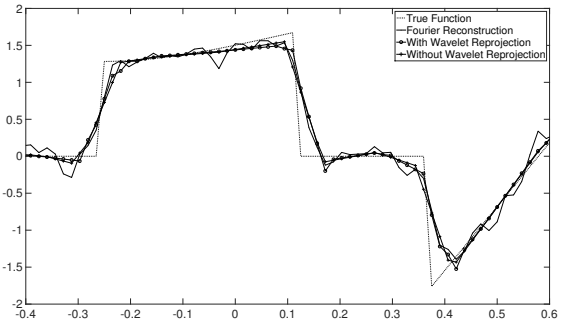


(a) test case 1

(b) test case 2



(c) test case 3



(d) test case 4

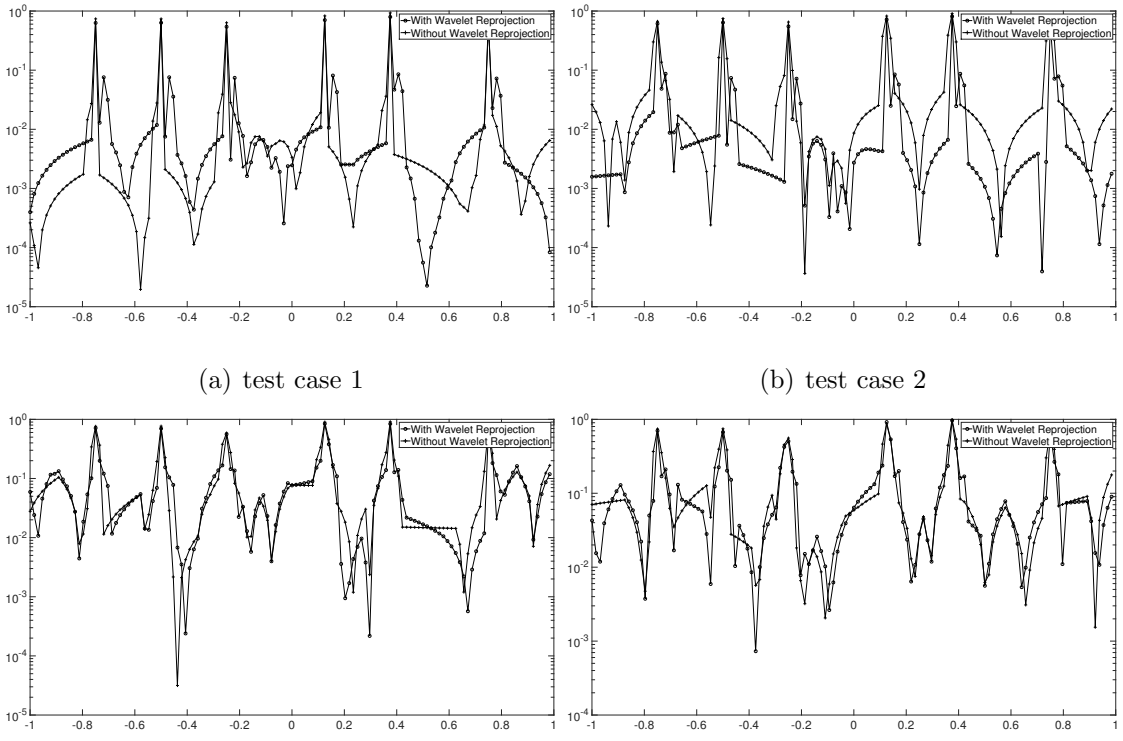
Figure 2.5: f_1,$\mathcal{F}^{-1}\hat{\mathbf{V}}$, $\vec{f}_{wav}$, and $\vec{f}_{nowav}$, on the Domain $[-4., .6]$ to Show Detail.

Consider the simplified[11] two-dimensional equivalent of (2.5):

$$\begin{aligned}\mathbf{c} = \underset{\mathbf{u}}{\operatorname{argmin}} \left\| \tilde{\mathbf{G}} \operatorname{vec}\left(\mathbf{F}\mathbf{W}^{-1}\mathbf{P}\mathbf{u}\left(\mathbf{F}\mathbf{W}^{-1}\mathbf{P}\right)^T\right) - \operatorname{vec}\left(\boldsymbol{\Sigma} \circ \hat{\mathbf{f}}_G\right)\right\|_2^2 + \\ \lambda_1 \left\| \operatorname{vec}\left(\mathbf{E}\left(\mathbf{W}^{-1}\mathbf{P}\mathbf{u}\left(\mathbf{W}^{-1}\mathbf{P}\right)^T\right)^T\right)\right\|_1 + \lambda_2 \left\| \operatorname{vec}\left(\mathbf{E}\mathbf{W}^{-1}\mathbf{P}\mathbf{u}\left(\mathbf{W}^{-1}\mathbf{P}\right)^T\right)\right\|_1 .\end{aligned} \tag{2.8}$$

The collected Fourier coefficients and the desired wavelet coefficients have been replaced with matrices. Thus, $\mathbf{u}$ is a matrix representing two-dimensional wavelet coefficients associated with the space $B_{low} \times B_{low}$. Here the "vec" operator transforms a matrix into a single column vector by concatenating the columns of the original matrix. The diagonal, binary selection matrix, $\tilde{\mathbf{G}}$, then acts to isolate the generated Fourier coefficients at the frequency pairs matching those available in $\hat{\mathbf{f}}_G$. A discrete filter is represented by $\boldsymbol{\Sigma}$. Here "$\circ$" is the element-wise Hadamard product. The aug-

[11]This formulation uses sparse sampling at integer frequencies.



(a) test case 1

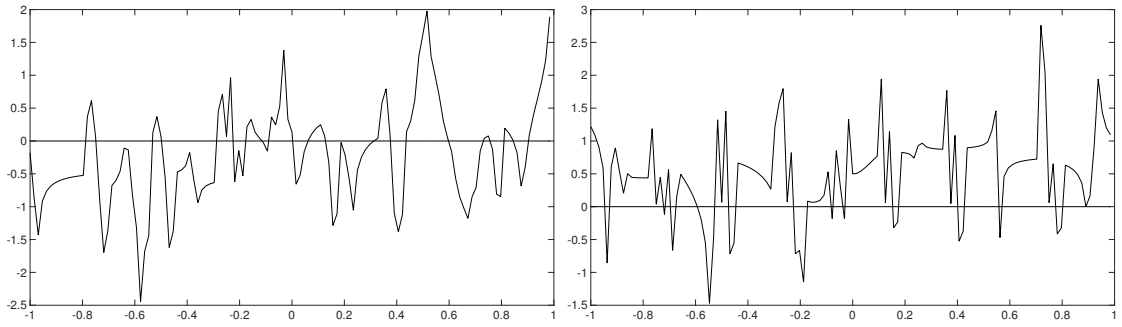
(b) test case 2

(c) test case 3

(d) test case 4

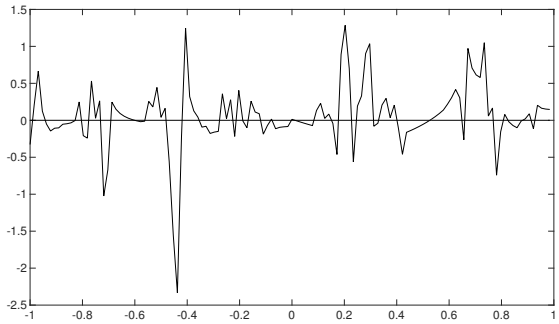
Figure 2.6: The Absolute Pointwise Reconstruction Errors in $\vec{f}_{wav}$ and $\vec{f}_{nowav}$.

mentation operator, $\mathcal{A}$, has been replaced with a padding matrix $\mathbf{P}$. The polynomial annihilation edge detector can be extended to two dimensions in a manner similar to anisotropic total variation. Instead, the edge detection regularization term has been split, with the first term detecting edges along the x direction and the second along the y direction as in Archibald *et al.* (2015). In practice, the regularization parameters, λ_1 and λ_2, will be chosen to be a single value, λ. This representation of the regularization, (2.8), is efficient in that the fast implementations of the Fourier and Wavelet transforms are naturally applied; however the bilinear forms are not easily used algebraically in the construction of iterative optimization methods.

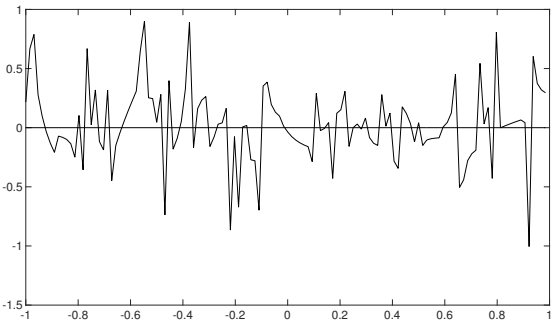


(a) test case 1

(b) test case 2



(c) test case 3



(d) test case 4

Figure 2.7: $\vec{Q}$ for Each Test Case.

2.6 Kronecker Product Representation

To create the iteration compatible with the split-Bregman algorithm, we use the following identity involving the Kronecker product, $\otimes$:

$$\operatorname{vec}(\mathbf{ACB}) = \left(\mathbf{B}^T \otimes \mathbf{A}\right) \operatorname{vec}(\mathbf{C})$$

to make the substitutions

$$\vec{u} = \operatorname{vec}(\mathbf{u}), \quad \overrightarrow{\hat{f}_G} = \operatorname{vec}\left(\hat{\mathbf{f}}_G\right), \quad \tilde{\mathbf{W}}^{-1} = \left(\mathbf{W}^{-1}\right)^T \otimes \mathbf{W}^{-1}, \quad \tilde{\mathbf{F}} = \mathbf{F}^T \otimes \mathbf{F}$$

$$\tilde{\mathbf{P}} = \mathbf{P}^T \otimes \mathbf{P}, \quad \tilde{\mathbf{E}}_y = \mathbf{I} \otimes \mathbf{E}, \quad \tilde{\mathbf{E}}_x = \mathbf{E} \otimes \mathbf{I}.$$

The vectorized equivalent of (2.8) can then be formed. The regularization parameter, λ, weights the penalty terms seeking sparsity against the fidelity term that seeks to have the solution match the provided samples. We shift this parameter from the

penalty terms to the fidelity term as $\frac{\mu}{2}$ to make the regularization compatible with Goldstein and Osher (2009); Archibald *et al.* (2015), resulting in

$$\vec{c} = \underset{\vec{u}}{\arg\min} \frac{\mu}{2} \left\| \tilde{\mathbf{G}}\tilde{\mathbf{F}}\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} - \overrightarrow{\hat{f}_G} \right\|_2^2 + \left\| \tilde{\mathbf{E}}_x\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} \right\|_1 + \left\| \tilde{\mathbf{E}}_y\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} \right\|_1 . \tag{2.9}$$

If we let $\vec{f} = \tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u}$, then (2.9) can be made to resemble the simplified version of (2.4) with the form

$$\vec{c} = \underset{\vec{u}}{\arg\min} \frac{\mu}{2} \left\| \tilde{\mathbf{G}}\tilde{\mathbf{F}}\vec{f} - \overrightarrow{\hat{f}_G} \right\|_2^2 + \left\| \tilde{\mathbf{E}}_x\vec{f} \right\|_1 + \left\| \tilde{\mathbf{E}}_y\vec{f} \right\|_1 . \tag{2.10}$$

2.7 The Vectorized Split Bregman Implementation

Having transformed (2.8) from a bilinear form to one based on left matrix multiplication, we now split (2.9) into sub-problems consistent with the Split-Bregman algorithm.

We first split (2.10) into ℓ^1 problems and a Euclidean norm problem. We let $\vec{d}_x := \tilde{\mathbf{E}}_x\vec{f}$ and $\vec{d}_y := \tilde{\mathbf{E}}_y\vec{f}$, which bridge the ℓ^1 and ℓ^2 problems and include a new regularization parameter, λ. This parameter, λ, is used in the Split-Bregman algorithm to weight the original fidelity term against the new fidelity terms involving $\vec{d}$. We also introduce the variables b_x and b_y to represent the sum of residuals between iterations as is standard with the Split-Bregman algorithm. This leads to three optimization problems that must be solved:

$$\vec{u}^{n+1} = \underset{\vec{u}}{\operatorname{argmin}} \frac{\mu}{2} \left\| \tilde{\mathbf{G}}\tilde{\mathbf{F}}\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} - \overrightarrow{\hat{f}_G} \right\|_2^2 + \frac{\lambda}{2} \left\| \vec{d}_x^n - \tilde{\mathbf{E}}_x\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} - \vec{b}_x^n \right\|_2^2 + \frac{\lambda}{2} \left\| \vec{d}_y^n - \tilde{\mathbf{E}}_y\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u} - \vec{b}_y^n \right\|_2^2 \tag{2.11a}$$

$$\vec{d}_x^{n+1} = \underset{\vec{d}}{\operatorname{argmin}} \left\| \vec{d} - \tilde{\mathbf{E}}_x\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u}^{n+1} - \vec{b}_x \right\|_1 \tag{2.11b}$$

$$\vec{d}_y^{n+1} = \underset{\vec{d}}{\operatorname{argmin}} \left\| \vec{d} - \tilde{\mathbf{E}}_y\tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}\vec{u}^{n+1} - \vec{b}_y \right\|_1 . \tag{2.11c}$$

We solve (2.11a) by differentiation using the general result

$$\underset{\vec{x}}{\operatorname{argmin}} \frac{\lambda}{2} \left\| \mathbf{A}\vec{x} - \vec{b} \right\|_2^2 = \vec{x} \quad s.t. \quad \frac{\partial}{\partial\vec{x}} \frac{\lambda}{2} \left\| \mathbf{A}\vec{x} - \vec{b} \right\|_2^2 = \lambda \left(\mathbf{A}^*\mathbf{A}\vec{x} - \mathbf{A}^*b \right) = 0.$$

To conserve space we let $\tilde{\mathbf{R}} = \tilde{\mathbf{W}}^{-1}\tilde{\mathbf{P}}$. In practice, the two matrices, $\mathbf{W}^{-1}$ and $\mathbf{P}$, are combined into a rectanglar matrix, $\mathbf{R}$, containing the first half of the columns of $\mathbf{W}^{-1}$. We now have the solution to (2.11a) given by

$$\mu\left(\tilde{\mathbf{G}}\tilde{\mathbf{F}}\tilde{\mathbf{R}}\right)^*\tilde{\mathbf{G}}\tilde{\mathbf{F}}\tilde{\mathbf{R}}\vec{u} - \mu\left(\tilde{\mathbf{G}}\tilde{\mathbf{F}}\tilde{\mathbf{R}}\right)^*\overrightarrow{\hat{f}_G} + \lambda\left(\left(\tilde{\mathbf{E}}_x\tilde{\mathbf{R}}\right)^*\left(\tilde{\mathbf{E}}_x\tilde{\mathbf{R}}\vec{u} - \left(\vec{d}_x^n - \vec{b}_x^n\right)\right) + \left(\tilde{\mathbf{E}}_y\tilde{\mathbf{R}}\right)^*\left(\tilde{\mathbf{E}}_y\tilde{\mathbf{R}}\vec{u} - \left(\vec{d}_y^n - \vec{b}_y^n\right)\right)\right) = 0$$

or

$$\tilde{\mathbf{R}}^*\left(\mu\tilde{\mathbf{F}}^*\tilde{\mathbf{G}}^*\tilde{\mathbf{G}}\tilde{\mathbf{F}} + \lambda\tilde{\mathbf{E}}_x^*\tilde{\mathbf{E}}_x + \lambda\tilde{\mathbf{E}}_y^*\tilde{\mathbf{E}}_y\right)\tilde{\mathbf{R}}\vec{u} = \tilde{\mathbf{R}}^*\left(\mu\tilde{\mathbf{F}}^*\tilde{\mathbf{G}}^*\overrightarrow{\hat{f}_G} + \lambda\tilde{\mathbf{E}}_x^*\left(\vec{d}_x^n - \vec{b}_x^n\right) + \lambda\tilde{\mathbf{E}}_y^*\left(\vec{d}_y^n - \vec{b}_y^n\right)\right). \tag{2.12}$$

In (2.12), $\tilde{\mathbf{R}}^*$ is singular and cannot simply be removed.

The ℓ^1 equations, (2.11b) and (2.11c), are solved using the shrink operator as described in Appendix B. The shrink operator is defined as

$$\text{shrink}\left(\vec{x},\gamma\right)_j = \frac{x_j}{|x_j|}\max\left(0,|x_j|-\gamma\right). \tag{2.13}$$

The solutions to (2.11b) and (2.11c) are then given by

$$\vec{d}_x^{n+1} = \text{shrink}\left(\vec{d}_x - \vec{b}_x, \frac{1}{\lambda}\right) \tag{2.14a}$$

$$\vec{d}_x^{n+1} = \text{shrink}\left(\vec{d}_x - \vec{b}_x, \frac{1}{\lambda}\right). \tag{2.14b}$$

2.8 A Memory Efficient Implementation

For efficiency we convert the right hand side of (2.12) back to matrix notation. Also the diagonal, binary matrix, $\tilde{\mathbf{G}}^T$, projects $\overrightarrow{\hat{f}_G}$ back to itself, so $\tilde{\mathbf{G}}^T\overrightarrow{\hat{f}_G} = \overrightarrow{\hat{f}_G}$. The matrix form is then given by

$$rhs = \tilde{\mathbf{R}}^*\left(\mu\tilde{\mathbf{F}}^*\tilde{\mathbf{G}}^*\overrightarrow{\hat{f}_G} + \lambda\tilde{\mathbf{E}}_x^*\left(\vec{d}_x^n - \vec{b}_x^n\right) + \lambda\tilde{\mathbf{E}}_y^*\left(\vec{d}_y^n - \vec{b}_y^n\right)\right) =$$

$$\mathbf{R}^T\left(\mu\mathbf{F}^*\hat{\mathbf{f}}_G\mathbf{F}^{*T} + \lambda\left(\mathbf{d}_x^n - \mathbf{b}_x^n\right)\mathbf{E}_x + \lambda\mathbf{E}_y{}^T\left(\mathbf{d}_y^n - \mathbf{b}_y^n\right)\right)\mathbf{R}. \tag{2.15}$$

Using $\tilde{\mathbf{G}}^T\tilde{\mathbf{G}} = \tilde{\mathbf{G}}$ we derive the matrix form of the left hand side given by

$$\begin{aligned} lhs = \tilde{\mathbf{R}}^* \left(\mu\tilde{\mathbf{F}}^*\tilde{\mathbf{G}}^*\tilde{\mathbf{G}}\tilde{\mathbf{F}} + \lambda\tilde{\mathbf{E}}_x^*\tilde{\mathbf{E}}_x + \lambda\tilde{\mathbf{E}}_y^*\tilde{\mathbf{E}}_y\right)\tilde{\mathbf{R}}\vec{u} = \\ \mathbf{R}^T\left(\mu\mathbf{F}^*\left(\mathbf{G}\circ\left(\mathbf{F}\mathbf{R}\mathbf{u}\mathbf{R}^T\mathbf{F}^T\right)\right)\mathbf{F}^{*T} + \lambda\mathbf{R}\mathbf{u}\mathbf{R}^T\mathbf{E}^T\mathbf{E} + \lambda\mathbf{E}^T\mathbf{E}\mathbf{R}\mathbf{u}\mathbf{R}^T\right)\mathbf{R}. \end{aligned} \tag{2.16}$$

In spite of being complicated, the matrices involved either have fast transform equivalents or are extremely sparse. With the matrix represention, (2.15) and (2.16), (2.11a) is solved quickly by the conjugate gradient method. Also, as will be discussed in Section 3.1.3, if the domain consists of a Cartesian grid with periodic boundary conditions, then $\mathbf{E}$ is circulant and therefore represents a discrete convolution operator. The convolution theorem can then be applied to the edge detection matrices. Thus

$$\tilde{\mathbf{R}}^*\left(\mu\tilde{\mathbf{F}}^*\tilde{\mathbf{G}}^*\tilde{\mathbf{G}}\tilde{\mathbf{F}} + \lambda\tilde{\mathbf{E}}_x^*\tilde{\mathbf{E}}_x + \lambda\tilde{\mathbf{E}}_y^*\tilde{\mathbf{E}}_y\right)\tilde{\mathbf{R}}\vec{u} = \tilde{\mathbf{R}}^*\tilde{\mathbf{F}}^*\left(\mu\tilde{\mathbf{G}}^*\tilde{\mathbf{G}} + \lambda\tilde{\mathbf{F}}\tilde{\mathbf{E}}_x^*\tilde{\mathbf{E}}_x\tilde{\mathbf{F}}^* + \lambda\tilde{\mathbf{F}}\tilde{\mathbf{E}}_y^*\tilde{\mathbf{E}}_y\tilde{\mathbf{F}}^*\right)\tilde{\mathbf{F}}\tilde{\mathbf{R}}\vec{u}$$

where $\mu\tilde{\mathbf{G}}^*\tilde{\mathbf{G}} + \lambda\tilde{\mathbf{F}}\tilde{\mathbf{E}}_x^*\tilde{\mathbf{E}}_x\tilde{\mathbf{F}}^* + \lambda\tilde{\mathbf{F}}\tilde{\mathbf{E}}_y^*\tilde{\mathbf{E}}_y\tilde{\mathbf{F}}^*$ is diagonal. This allows for a fast, direct solution to (2.16).

2.9 Two Dimensional Results

To demonstrate (2.8) in two dimensions, we use the same test function as that found in Archibald *et al.* (2015). The test function consists of three pieces in the domain $[-1,1)\times[-1,1)$ using regions defined as

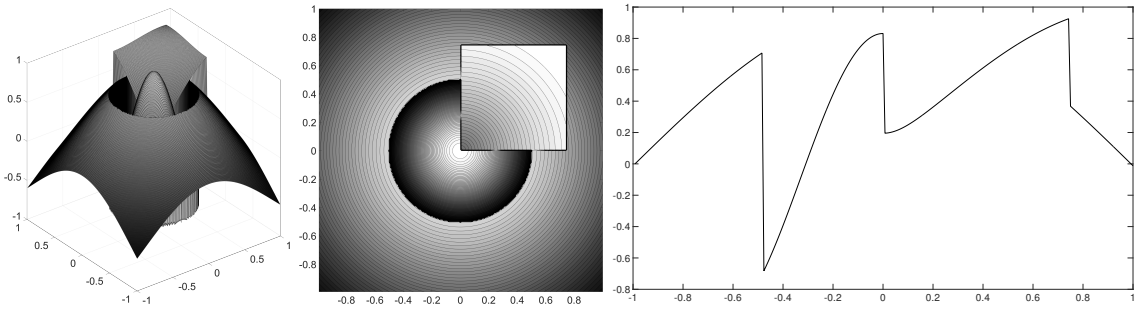
$$S = \left[0,\frac{3}{4}\right)\times\left[0,\frac{3}{4}\right)$$

$$T = \left\{(x,y) : \sqrt{x^2+y^2} <= \frac{1}{2}\right\}$$

with the function defined as

$$f_2(x,y) = \begin{cases} \sin\frac{\pi}{2}\sqrt{x^2+y^2} & (x,y)\in S \\ \cos\frac{3\pi}{2}\sqrt{x^2+y^2} & (x,y)\in\left(S^C\cap T\right) \\ \cos\frac{\pi}{2}\sqrt{x^2+y^2} & (x,y)\in\left(S^C\cap T^C\right). \end{cases}$$

We let the discretely sampled values of f_2 be $\mathbf{F}_{j,k} = f_2\left(-1 + \frac{2(j-1)}{N}, -1 + \frac{2(k-1)}{N}\right)$. Figure 2.8 shows f_2 both as a surface and as a contour plot. Also included is a cross-section of the function along the line $y = \frac{1}{8}$ that will be examined in various test cases.



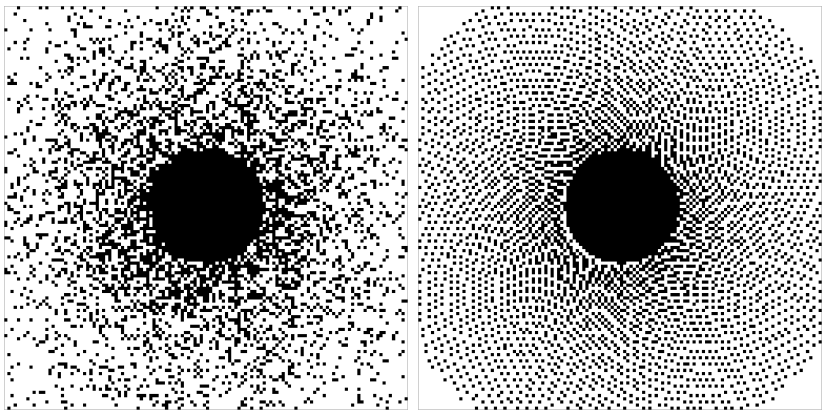
(a) Surface plot (b) Contour plot (c) Cross-section

Figure 2.8: The Test Function, f_2, Shown As a Surface Plot, a Contour Plot, and a One-dimensional Cross-section along the Line $y = \frac{1}{8}$.

We develop test cases as in one dimension, but extend the two-dimensional sampling method to form the set $\hat{\mathbf{V}} = \hat{\mathbf{F}} \cup \hat{\mathbf{R}}$. The tests approximate the sampling density found in modern MRI imaging with a higher density in the low frequencies and a roughly radially symmetric pattern, although we constrain the problem to use only integer frequencies. The provided Fourier samples are close approximations to the continuous Fourier transform of the test function.[12] It was found in prior one-dimensional test cases in Fan and Mead (2015), that when the lowest frequency samples are missing, reconstruction is impossible. Thus in all cases a central disk of frequencies is retained. This is controlled by a parameter ζ where , such that $\dim\left(\widehat{\mathbf{F}}\right) = \zeta N^2$ and $\widehat{\mathbf{F}} = \left\{\hat{f}_{j,k} : \sqrt{j^2 + k^2} \leqslant N\sqrt{\frac{\zeta}{\pi}}\right\}$. In a manner similar to the one-dimensional case, $dim(\hat{\mathbf{V}}) = \gamma N^2$. To form the set $\hat{\mathbf{R}}$ we test both the Gaussian

[12]The Fourier coefficients are generated by using the Fast Fourier Transform at the highest resolution that memory on the testing machine allows. The appropriate frequencies are then extracted and renormalized.

and spiral sampling patterns shown in Figure 2.9.



(a) Gaussian (b) Spiral

Figure 2.9: K-space Sampling Distributions with $\gamma = 0.3$.

In the Gaussian sampling pattern, the sample frequencies are drawn from a random number generator with a radial Gaussian probability distribution having mean zero and standard deviation $\frac{3}{8}N$. Frequencies are rounded to the nearest integer grid point and bounded in the square $\left[-\frac{N}{2}, \frac{N}{2}\right) \times \left[-\frac{N}{2}, \frac{N}{2}\right)$. Samples are drawn and duplicates removed until the desired sub-sampling ratio is achieved.

With the spiral sampling pattern, frequencies are selected with similar rounding, bounding and duplicate removal. We choose the increment values given by $\Delta\theta = \frac{3}{4}\sqrt{\frac{N^2(s-r)}{2}}$ and $\Delta\rho = \frac{1}{\Delta\theta N^2(s-r)}$. The form of these increments is chosen to cover the available square of Fourier modes as much as possible and to produce a spiral pattern as opposed to a radial pattern. With these increments a polar sequence can be created to cover k-space as given by

$$(\omega_x, \omega_y) = \left(j\Delta\rho\cos\left(j\Delta\theta\right), j\Delta\rho\sin\left(j\Delta\theta\right)\right); \quad j \in s\mathbb{N}^2.$$

We extend the one-dimensional reconstruction definitions by letting $\mathbf{F}_{wav}$ represent the reconstruction formed by the application of (2.8) to $\hat{\mathbf{V}}$ with polynomial annihilation order 3, $\mathbf{F}_{nowav}$ represent the reconstruction formed by the application

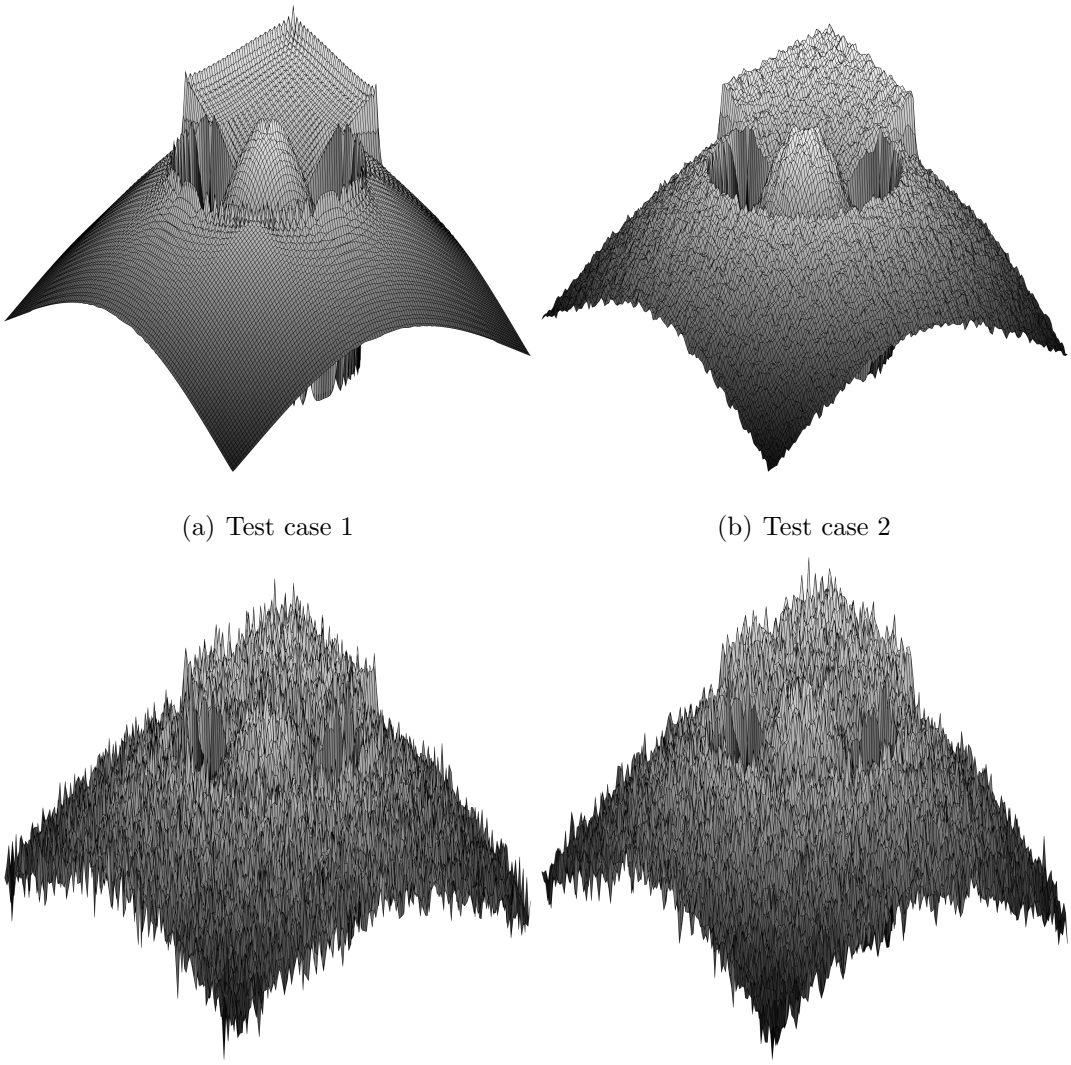
of (2.4) to $\hat{\mathbf{V}}$ with polynomial annihilation order 2, and $\mathbf{F}_{TV}$ represent the reconstruction formed by the application of (2.4) to $\hat{\mathbf{V}}$ with polynomial annihilation order 1, since polynomial annihilation of order 1 is equivalent to total variation up to a constant. Other values for the polynomial annihilation order were tested numerically, and these values gave inferior results, because a higher order polynomial annihilation operator has a larger edge response. To better compare the methods, we define the point-wise error ratio between two methods as

$$\mathbf{Q}(method1, method2)_{j,k} = \log_{10}\left(\frac{\left|\mathbf{F}_{method2_{j,k}} - \mathbf{F}_{j,k}\right| + \varepsilon}{\left|\mathbf{F}_{method1_{j,k}} - \mathbf{F}_{j,k}\right| + \varepsilon}\right) \tag{2.17}$$

For all the reconstruction methods, the accuracy in the neighborhood of edges is first order. To analyze the reconstruction accuracy in smooth regions quantitatively, some results will explicitly exclude cells in the neighborhood of edges of f_2. The radius of the neighborhood ignored will be given by the parameter η. Unless otherwise specified, we let $\eta = 0$, meaning no edge cells are excluded. We let C_{edge} be the count of edge cells excluded.

The primary two-dimensional test cases are given in Table 2.2. Unless otherwise specified, the grid point count will be $N = 128$, the Gaussian sampling pattern will be used with $\zeta = \frac{1}{40}$, where $\dim\left(\widehat{\mathbf{F}}\right) = \zeta N^2$, and the polynomial annihilation order for the edge detector in (2.8) will be 3. We also use a $\frac{\mu}{\lambda}$ ratio of 4.46 in (2.11a). This was determined by trial and error, finding the mid-point of a set of $\frac{\mu}{\lambda}$ values that achieved good results across a range of noise levels and sub-sampling ratios. Of course, in general this value will depend on the specific level of noise, the amount of sub-sampling, and the form of the test function.

To demonstrate the effect of noise and sub-sampling, Figure 2.10 shows the Fourier reconstructions, $\mathcal{F}^{-1}\widehat{\mathbf{V}}(\mathcal{F}^{-1})^T$ for each of the test cases in Table 2.2. The oscillations associated with Gibbs phenomenon are clearly visible in test case 1.



(a) Test case 1

(b) Test case 2

(c) Test case 3

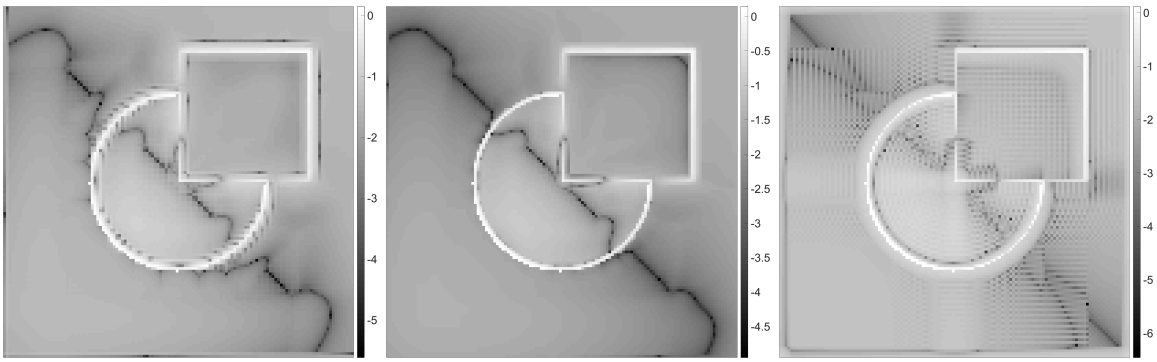
(d) Test case 4

Figure 2.10: The Fourier Reconstruction of f_2 Associated with the Test Cases.

Table 2.2: Parameters Selected for Each Two-dimensional Test Case. Unless otherwise Specified We Let $N = 128$, $\zeta = \frac{1}{40}$, and the Gaussian Sampling Pattern Will Be Used.

Test Case	SNR	γ
1	∞	1
2	∞	0.5
3	7dB	1
4	7dB	0.5

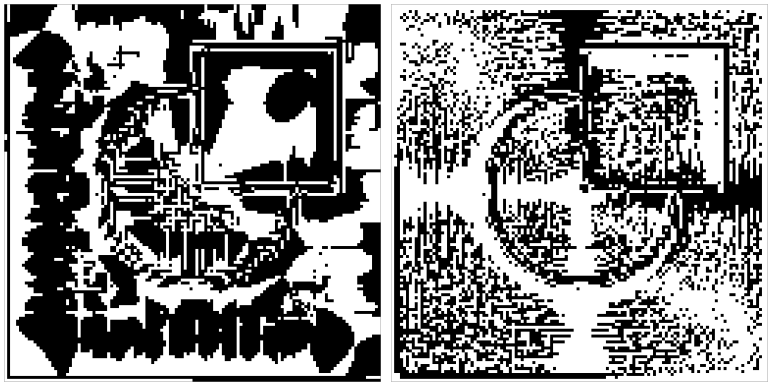
Figure 2.11 shows the reconstruction errors associated with the three methods: $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for test case 1. Figure 2.12 highlights the regions where the reconstruction accuracy for $\mathbf{F}_{wav}$ is superior to that of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. Figure 2.13 shows the reconstructions and associated reconstruction errors for each of the methods along the cross-section, $y = \frac{1}{8}$. In test case 1, $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$ have similar levels of accuracy in smooth regions, but they differ in the neighborhood of edges. The total variation reconstruction suffers from the "staircase" artifact, which is clearly visible in the rapid oscillations in relative accuracy seen in Figure 2.12.



(a) Wavelet (b) PA(2) (c) TV

Figure 2.11: Test Case 1 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$.

In test case 2, the original data set is sub-sampled. Figure 2.14 shows the reconstruction errors associated with the three methods: $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for



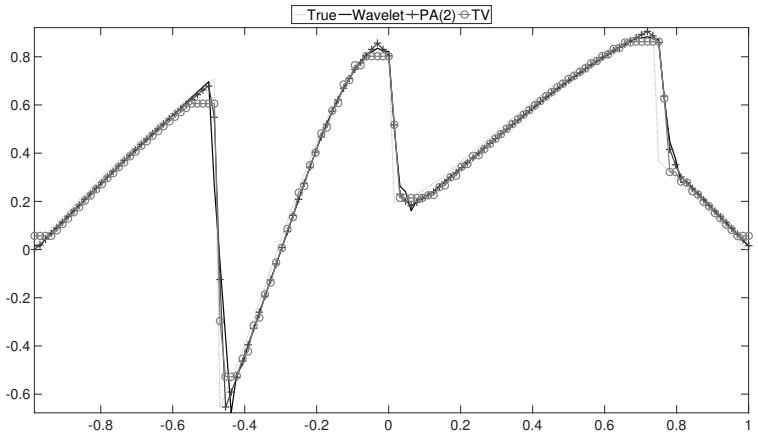
(a) PA(2) (b) TV=PA(1)

Figure 2.12: Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 1. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate.

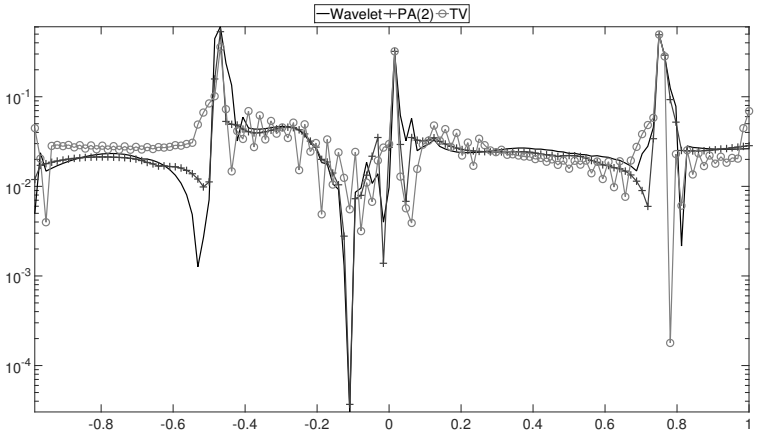
test case 2. Figure 2.15 highlights the regions where the reconstruction accuracy for $\mathbf{F}_{wav}$ is superior to that of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. Figure 2.16 shows the reconstructions and associated reconstruction errors for each of the methods along the cross-section, $y = \frac{1}{8}$. Visually there is little difference in the relative performance of the methods in the case of 50% sub-sampling as compared to no sub-sampling as shown in Figures 2.15 and 2.12 respectively, but subtle variations in the relative errors are visible in the cross-section graphs, Figures 2.13 and 2.16.

In test case 3, the original sample locations are unchanged, but complex Gaussian noise is added to each sample. Figure 2.17 shows the reconstruction errors associated with the three methods: $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for test case 3. Figure 2.18 highlights the regions where the reconstruction accuracy for $\mathbf{F}_{wav}$ is superior to that of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. Figure 2.19 shows the reconstructions and associated reconstruction errors for each of the methods along the cross-section, $y = \frac{1}{8}$. In this test, $\mathbf{F}_{wav}$ has increased accuracy in smooth regions but also has poorer edge resolution relative to the other methods as shown in Figures 2.18 and 2.19.

We perform sub-sampling and add noise in test case 4. Figure 2.20 shows the reconstruction errors associated with the three methods: $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for

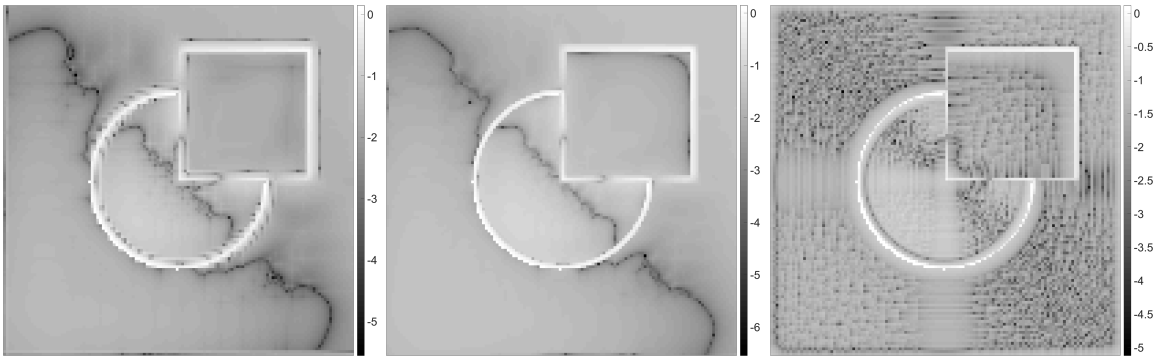


(a) PA(2)



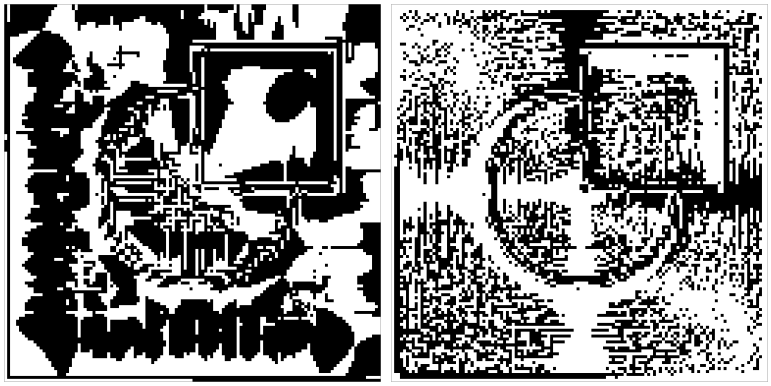
(b) TV=PA(1)

Figure 2.13: The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 1 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$.



(a) Wavelet (b) PA(2) (c) TV=PA(1)

Figure 2.14: Test Case 2 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$.



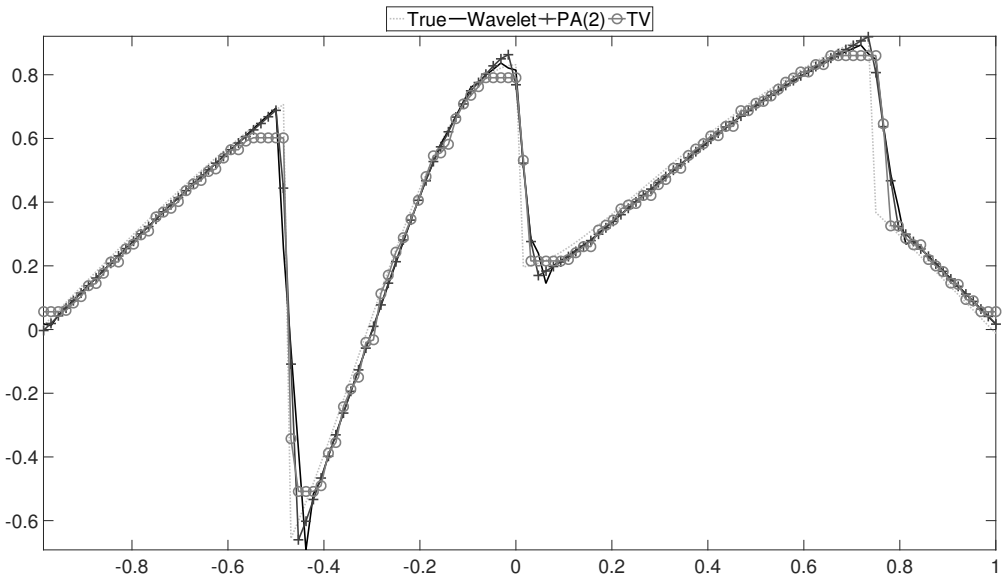
(a) PA(2) (b) TV=PA(1)

Figure 2.15: Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 2. White Areas Indicate $\mathbf{F}_{wav}$ is More Accurate.

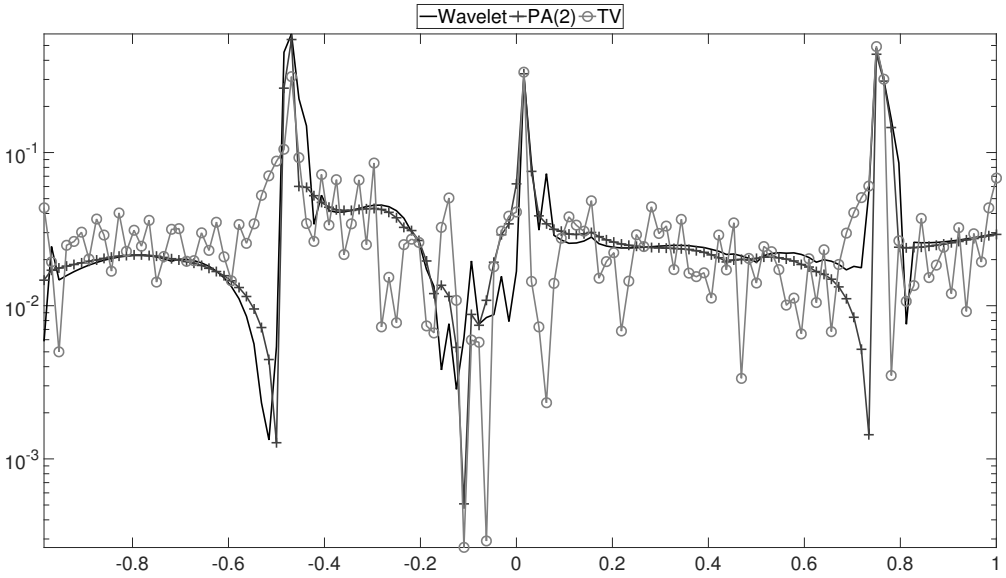
test case 4. Figure 2.21 highlights the regions where the reconstruction accuracy for $\mathbf{F}_{wav}$ is superior to that of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$. Figure 2.22 shows the reconstructions and associated reconstruction errors for each of the methods along the cross-section, $y = \frac{1}{8}$. As in test case 3, the wavelet based reconstruction, (2.8), appears to perform better than the other methods as seen by examining Figure 2.22. Since this test case represents the worst corruption of the provided samples, Figure 2.23 shows a side by side comparison of the original test function, the Fourier reconstruction from the the sub-sampled and noisy data, and the reconstructions $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$. The "staircase" artifact is clearly visible in the total variation reconstruction and oscillations in the reconstruction are best controlled in $\mathbf{F}_{wav}$.

Quantitative Comparison of Methods

We compare the reconstruction methods quantitatively by using the ratio of reconstruction errors, (2.17). In Table 2.3, the fraction of the cells in the reconstruction having better accuracy using (2.8) are tabulated. We see that in test cases 1 and 2, the regularization using the second order polynomial annihilation edge detector alone is superior, while (2.8) has better performance in the presence of noise. In all cases,

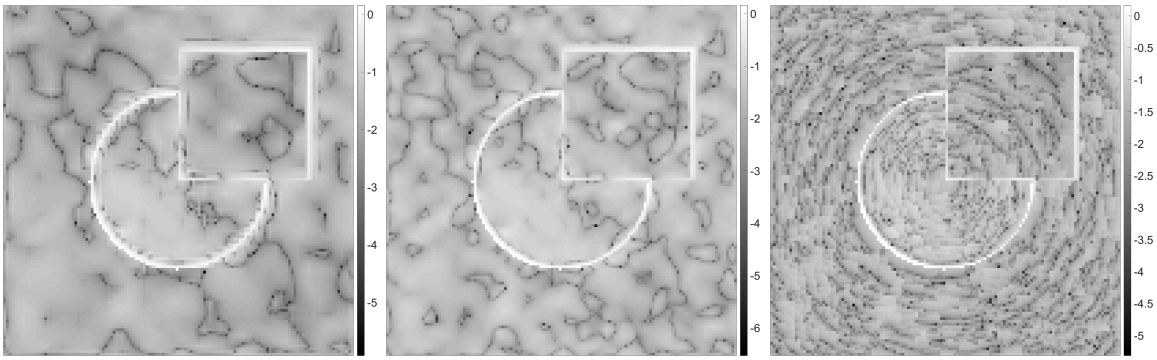


(a) PA(2)



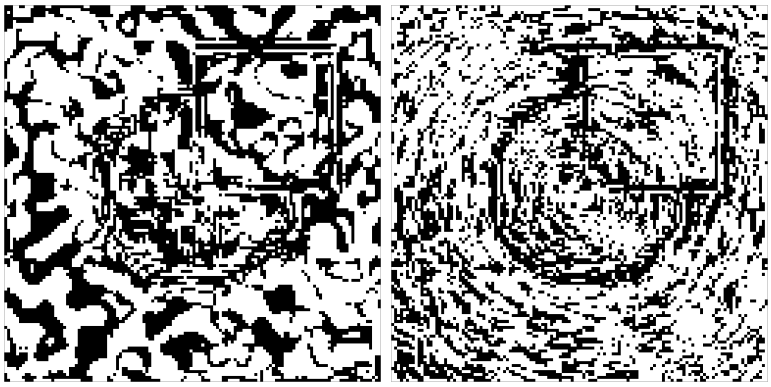
(b) TV=PA(1)

Figure 2.16: The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 2 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$.



(a) Wavelet (b) PA(2) (c) TV=PA(1)

Figure 2.17: Test Case 3 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$.



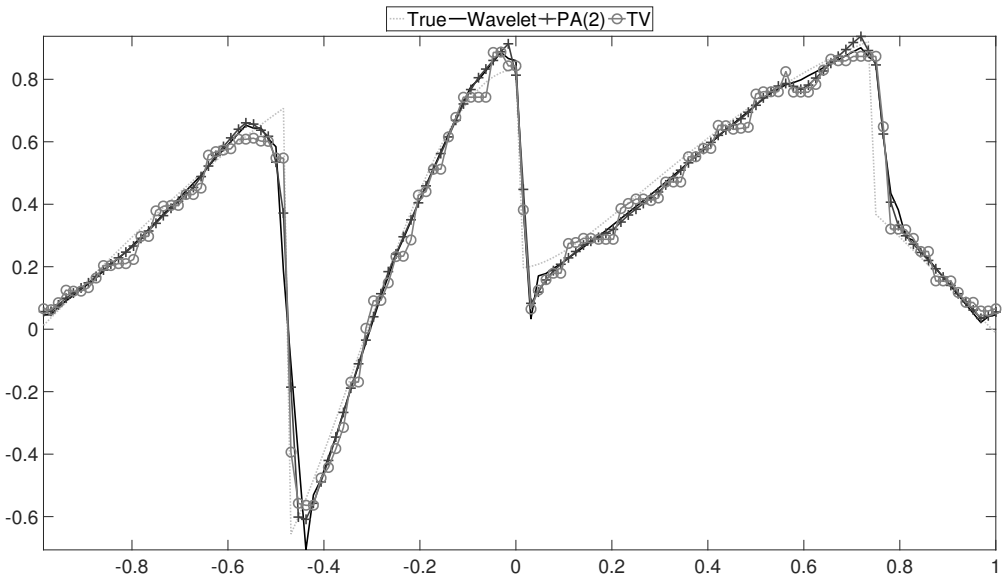
(a) PA(2) (b) TV=PA(1)

Figure 2.18: Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 3. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate.

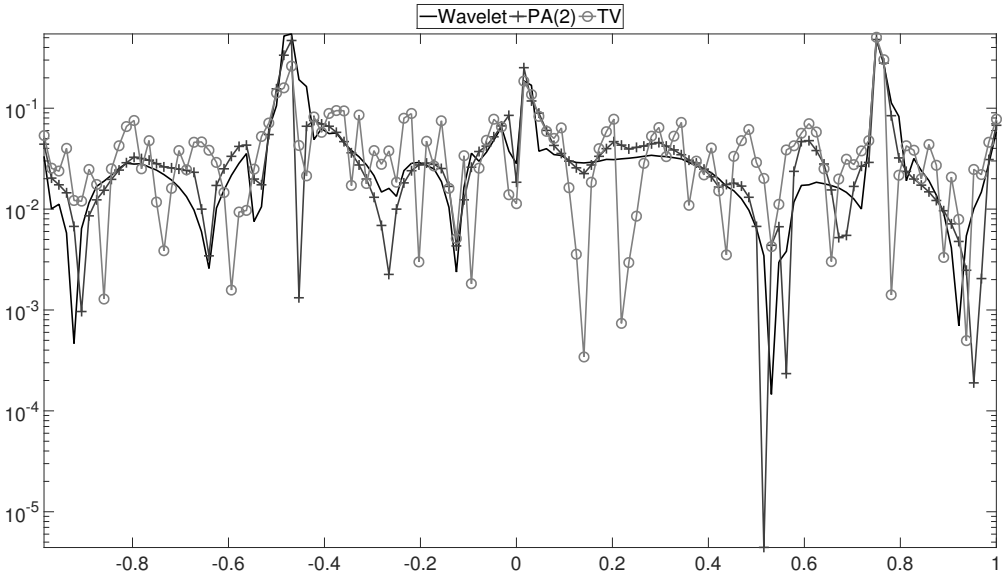
(2.8) has better performance than the total variation regularization.

Table 2.4 shows the sum of (2.17) across all cells normalized by the size of the image. This provides an indication about whether (2.8) has relatively large reconstruction errors or successes compared to the other methods. This method of measuring performance follows the pattern in table 2.3 indicating that the matrix of reconstruction error ratios do not have a few dominant cells.

Recognizing the reconstructions in the neighborhood of edges is first order and that the individual methods have different edge resolution, we now focus on the performance of the methods in smooth regions by tabulating the ratio of reconstruction

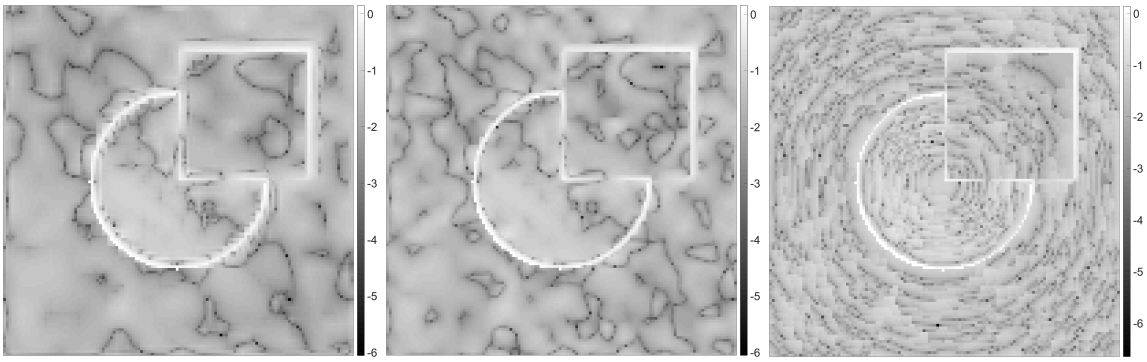


(a) PA(2)



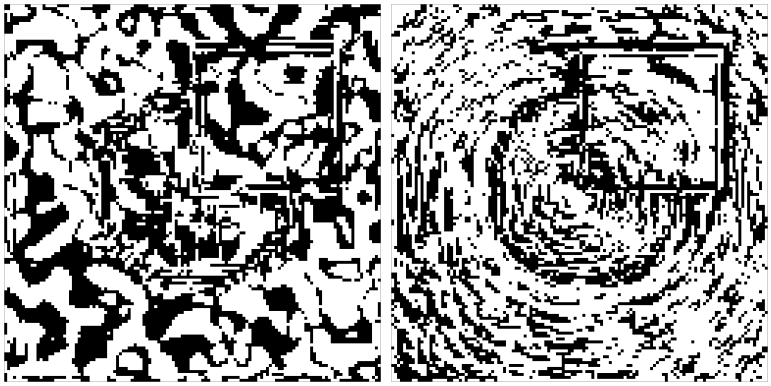
(b) TV=PA(1)

Figure 2.19: The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 3 and log_{10} Reconstruction Errors Along the Cross-section $y = \frac{1}{8}$.



(a) Wavelet (b) PA(2) (c) TV=PA(1)

Figure 2.20: Test Case 4 log_{10} Reconstruction Errors in $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$.



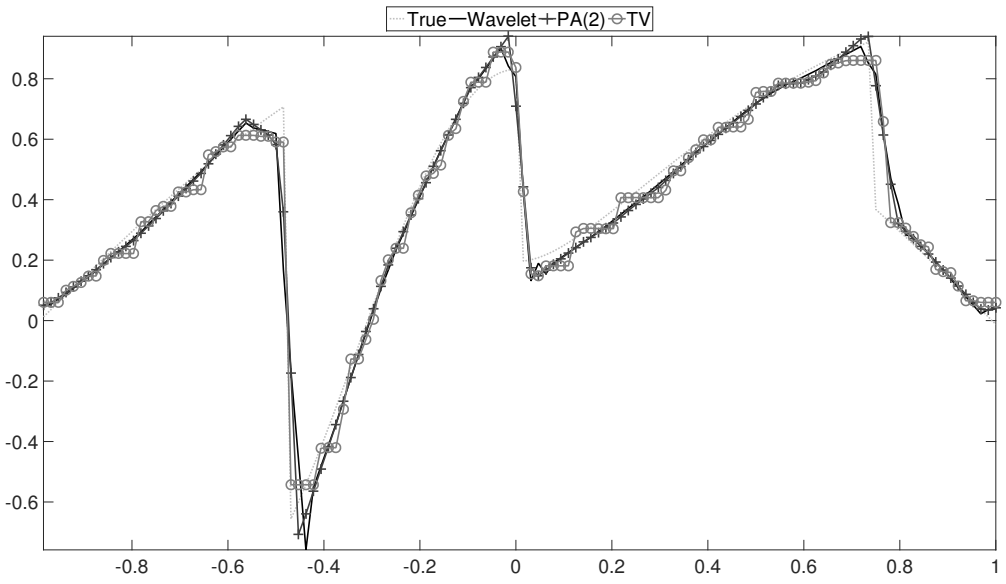
(a) PA(2) (b) TV=PA(1)

Figure 2.21: Comparison of Accuracy between $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$, $\mathbf{F}_{TV}$ Respectively for Test Case 4. White Areas Indicate $\mathbf{F}_{wav}$ Is More Accurate.

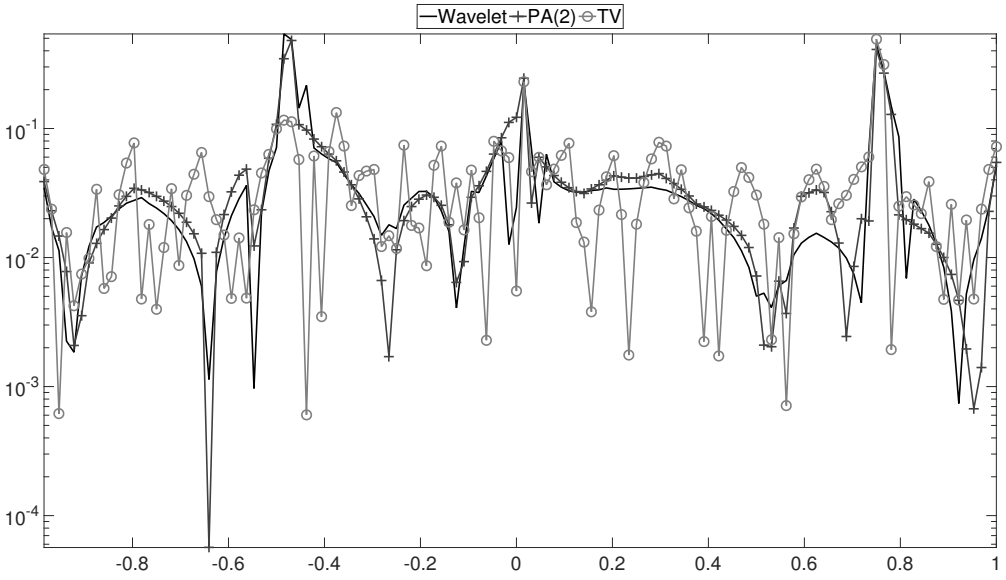
errors in cells away from edges with neighborhood radius, $\eta = 4$. In Table 2.5, the fraction of the cells in the reconstruction having better accuracy using (2.8) in smooth regions is tabulated. We see the same pattern as in table 2.3. Indeed, the performance of (2.4) increases relative to (2.8) in the absence of noise, while the performance of (2.8) relative to (2.4) increases in the presence of noise.

2.9.1 Filtering

In Archibald *et al.* (2015), the results include a preprocessing step involving the application of a spectral filter to the sample set, $\hat{\mathbf{V}}$. Such filtering accelerates con-

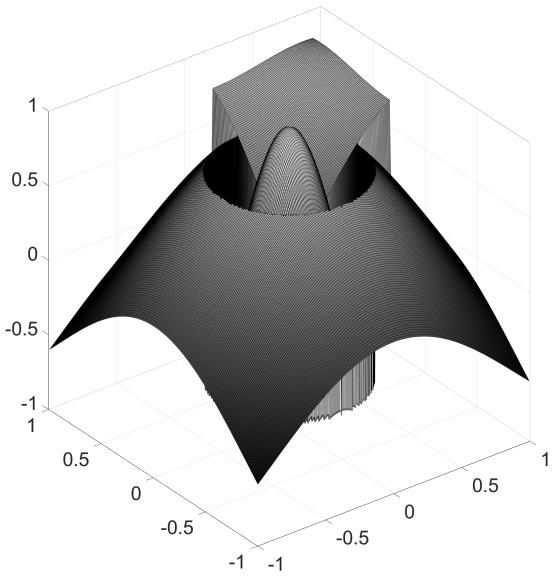


(a) PA(2)

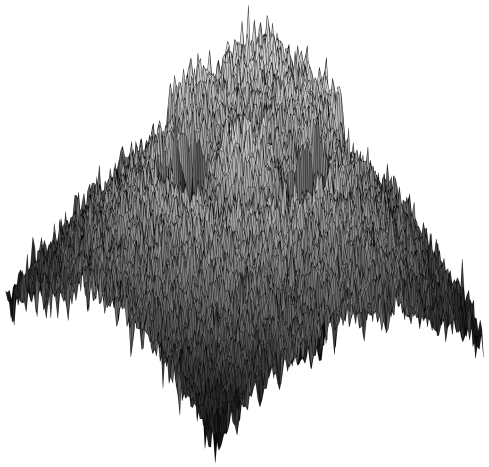


(b) TV=PA(1)

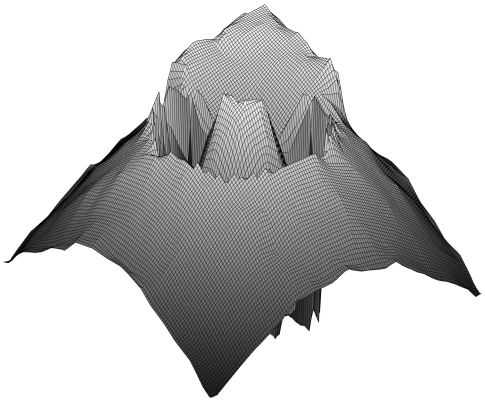
Figure 2.22: The One-dimensional Reconstruction: $\mathbf{F}_{wav}$,$\mathbf{F}_{nowav}$, and $\mathbf{F}_{TV}$ for Test Case 4 and log_{10} Reconstruction Errors along the Cross-section $y = \frac{1}{8}$.



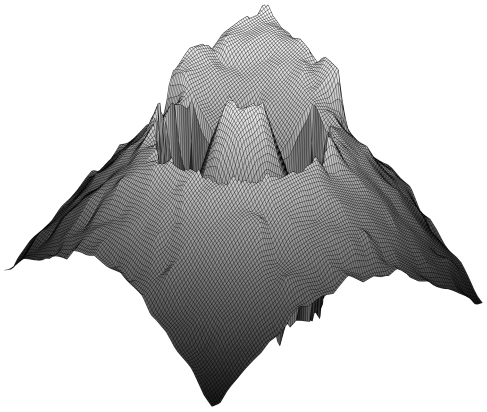
(a) $\mathbf{F}$



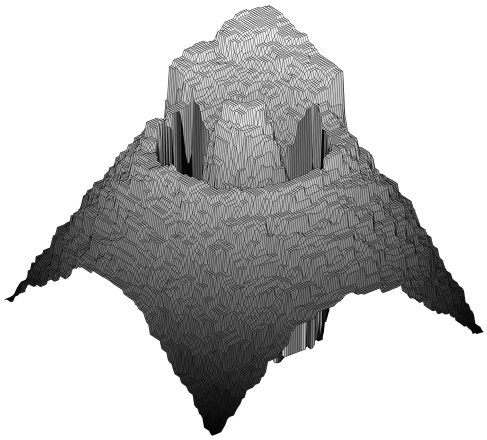
(b) $\mathcal{F}^{-1}\widehat{\mathbf{V}}\left(\mathcal{F}^{-1}\right)^T$



(c) $\mathbf{F}_{wav}$



(d) $\mathbf{F}_{nowav}$



(e) $\mathbf{F}_{TV}$

Figure 2.23: Comparison of Surface Reconstructions for Test Case 4.

Table 2.3: Fraction of Cells in which the Reconstruction Accuracy of $\mathbf{F}_{wav}$ Is Better Than That of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$.

Test Case	$\frac{\text{count}(Q(wav,nowav)\geqslant 0)}{N^2}$	$\frac{\text{count}(Q(wav,TV)\geqslant 0)}{N^2}$
Test case 1	0.430	0.547
Test case 2	0.443	0.604
Test case 3	0.581	0.635
Test case 4	0.550	0.639

Table 2.4: Sum of log_{10} Ratio of Reconstruction Errors with $\mathbf{F}_{wav}$ Compared to $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$.

Test Case	$\frac{\sum Q(wav,nowav)}{N^2}$	$\frac{\sum Q(wav,TV)}{N^2}$
Test case 1	-0.019	0.020
Test case 2	-0.032	0.058
Test case 3	0.019	0.112
Test case 4	0.019	0.128

vergence in smooth regions while reducing the impact of noise. Additional details regarding filtering will be provided in Section 4.1. We now test the effect of filtering by scaling $\hat{\mathbf{V}}$ with weights from a discrete exponential spectral filter defined by

$$\mathbf{\Sigma}_{k_x,k_y} = e^{-\beta\left(\frac{|k_x|}{\frac{N}{2}}\right)^{\rho}} e^{-\beta\left(\frac{|k_y|}{\frac{N}{2}}\right)^{\rho}} \tag{2.18}$$

Here the filter power, ρ, is a positive even integer and β is a defined such that $e^{-\beta} = \varepsilon_{machine}$, where $\varepsilon_{machine}$ is machine epsilon. The integer sample frequencies are represented by (k_x, k_y). The weight matrix, $\mathbf{\Sigma}$, is applied using the Hadamard product as in (2.8). Filters of various powers are shown in Figure 2.24.

Table 2.5: Fraction of Cells in which the Reconstruction Accuracy of $\mathbf{F}_{wav}$ is Better Than That of $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ Away From Edges.

Test Case	$\frac{\text{count}(Q(wav,nowav)\geqslant 0)}{N^2-C_{edge}}$	$\frac{\text{count}(Q(wav,TV)\geqslant 0)}{N^2-C_{edge}}$
Test case 1	0.404	0.532
Test case 2	0.438	0.602
Test case 3	0.615	0.673
Test case 4	0.585	0.677

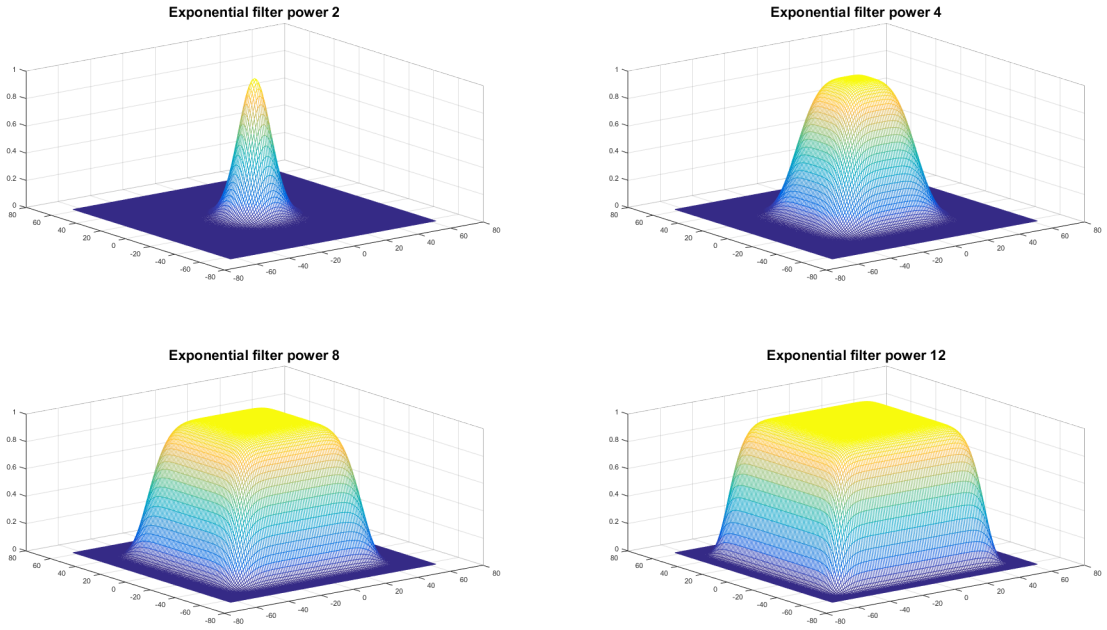
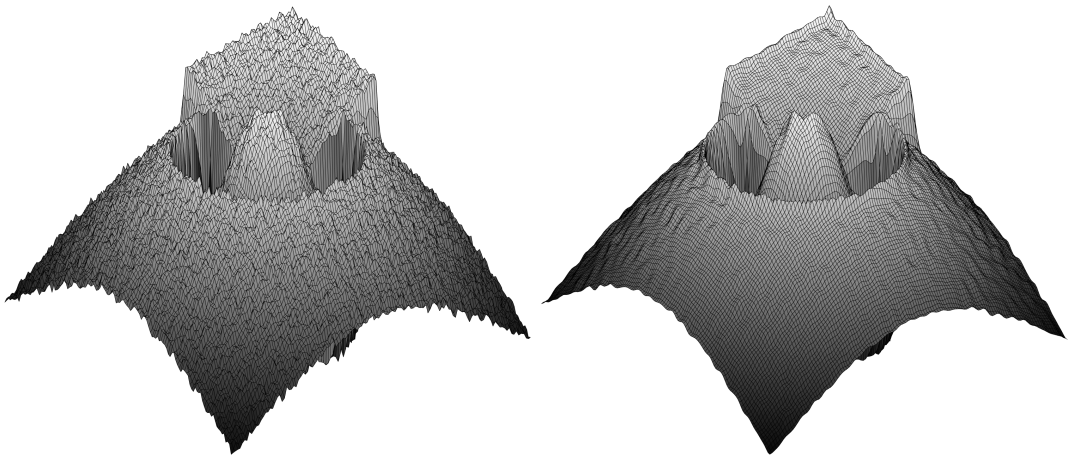


Figure 2.24: Filter Powers of Various Orders

By comparing Figures 2.24 and 2.9, it is evident that high-order filters barely impact the sub-sampled set of data. Thus, a low-order filter must be chosen, with the downside being that diffusion is introduced. To match the extent of the set of available coefficients, we choose $\rho = 4$, recognizing that diffusion will be introduced around edges. Figure 2.25 shows the effect of a fourth order filter on direct Fourier reconstuction without regularization of the sub-sampled data in test case 2. Table 2.6 shows the impact of filtering by comparing the ratio of reconstruction errors with and without filtering for each of the reconstruction methods. Table 2.7 shows the



(a) Without filtering (b) With filtering

Figure 2.25: The Effect of a Fourth Order Filter on the Fourier Reconstruction without Regularization Applied to Test Case 2.

same calculations, but excludes cells in the $\eta = 4$ neighborhood of edges. We see that when considering all cells, filtering has mixed results, but when focusing on smooth regions, filtering improves accuracy as expected, with the exception of test case 1 for (2.4). With less sub-sampling, a higher order filter would have been applicable, reducing the impact on edges.

Table 2.6: Fraction of Cells in which the Reconstruction Accuracy is Improved by Filtering for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$

Test Case	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2}$,$\mathbf{F}_{wav}$	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2}$,$\mathbf{F}_{nowav}$	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2}$,$\mathbf{F}_{TV}$
Test case 1	0.509	0.490	0.483
Test case 2	0.481	0.505	0.480
Test case 3	0.513	0.578	0.497
Test case 4	0.499	0.515	0.504

Spiral Sampling

We now switch our attention to examining the effect of spiral sampling vs. Gaussian sampling. Table 2.8 contains the parameters used in a new set of test cases.

We change γ to 0.3 in these tests only to make the sampling pattern clearly

Table 2.7: Fraction of Cells in which the Reconstruction Accuracy is Improved by Filtering for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$ in Smooth Regions.

Test Case	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{wav}$	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{nowav}$	$\frac{\text{count}(Q(filter,nofil.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{TV}$
Test case 1	0.516	0.488	0.534
Test case 2	0.502	0.506	0.524
Test case 3	0.559	0.629	0.543
Test case 4	0.524	0.532	0.526

Table 2.8: Parameters Selected for Each Two-dimensional Spiral Sampling Test Case.

Test Case	SNR	γ	Sampling Method
5	∞	0.3	Gaussian
6	∞	0.3	Spiral
7	7dB	0.3	Gaussian
8	7dB	0.3	Spiral

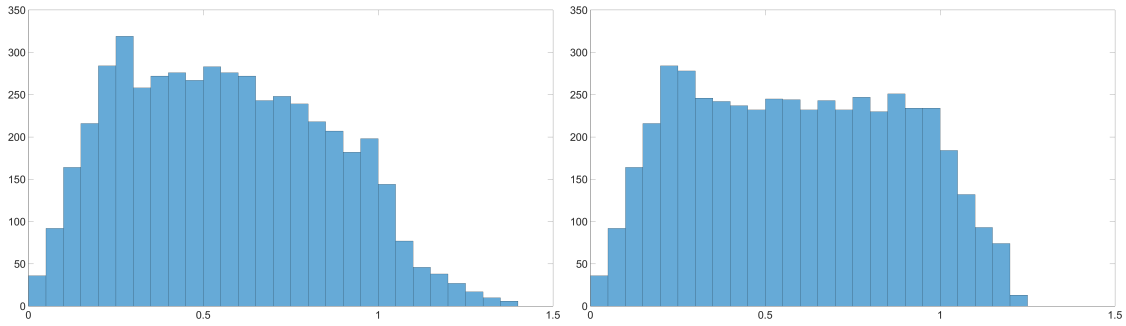
distinguishable from low density uniform sampling. Table 2.9 shows the effect of changing the sampling trajectory by comparing the ratio of reconstruction errors using Gaussian and spiral sampling for each of the reconstruction methods. Table 2.10 shows the same calculations, but excludes cells in the $\eta = 4$ neighborhood of edges. We see that with the addition of noise, spiral sampling results in a decrease of accuracy compared to Gaussian sampling. This is explained by the fact that with the chosen spiral sampling pattern, more high frequency samples are present as compared to the Gaussian sampling pattern as shown in Figure 2.26. High frequency samples tend to have smaller magnitudes and thus are more easily impacted by noise.

Table 2.9: Fraction of Cells in which the Reconstruction Accuracy Is Improved by Using Spiral Sampling for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$

Test Case	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2}$,$\mathbf{F}_{wav}$	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2}$,$\mathbf{F}_{nowav}$	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2}$,$\mathbf{F}_{TV}$
Test cases 5,6	0.530	0.521	0.522
Test cases 7,8	0.450	0.470	0.482

Table 2.10: Fraction of Cells in which the Reconstruction Accuracy Is Improved by Using Spiral Sampling in Smooth Regions for $\mathbf{F}_{wav}$, $\mathbf{F}_{nowav}$ and $\mathbf{F}_{TV}$.

Test Case	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{wav}$	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{nowav}$	$\frac{\text{count}(Q(spir.,Gaus.)\geqslant 0)}{N^2-C_{edge}}$,$\mathbf{F}_{TV}$
Test case 5,6	0.526	0.512	0.504
Test case 7,8	0.431	0.455	0.480

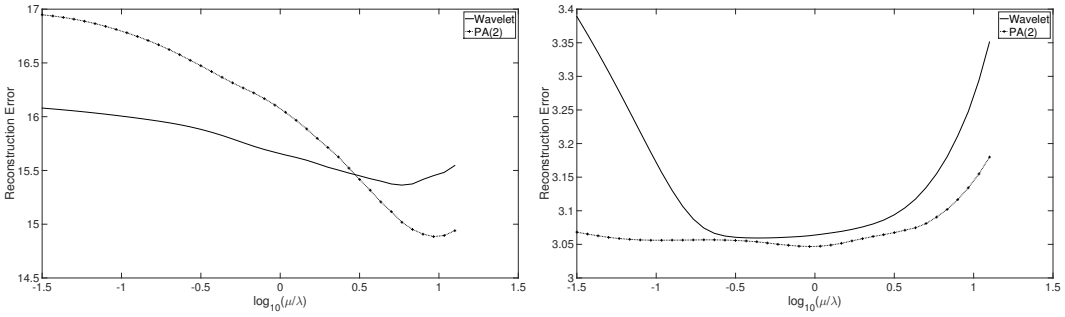


(a) No noise (b) With noise

Figure 2.26: Comparison of Radial Sampling Distributions for Gaussian and Spiral Sampling Patterns.

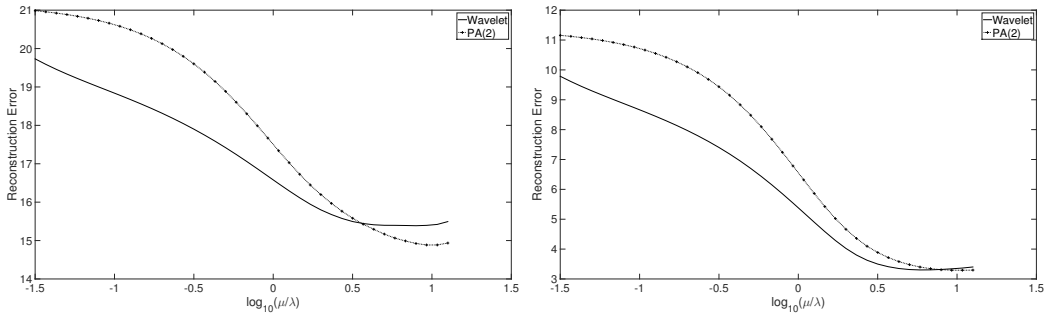
Dependence on Regularization Parameters

As a final test of the methods, we examine robustness with regard to the regularization parameter $\frac{\mu}{\lambda}$. Figure 2.27 shows the 2-norm of the reconstruction error as a function of $log_{10}\frac{\mu}{\lambda}$ for $\mathbf{F}_{wav}$ and $\mathbf{F}_{nowav}$ for test case 2. We consider all cells as well as those away from the $\eta = 4$ neighborhood of edges. Figure 2.28 shows the equivalent results for test case 4. We see that in general the slope of the regularization parameter dependence curve is more shallow for the wavelet based reconstruction indicating more robustness with regard to parameter selection. More importantly, we see crossings in the reconstruction error curves near the optimal parameter value. This indicates that the relative success of (2.8) vs. (2.4) is very much dependent on the choice of $\frac{\mu}{\lambda}$. The conclusion being that the two methods may differ in the details of their reconstructions, but are overall very similar on average.



(a) All cells (b) Smooth regions only

Figure 2.27: The 2-norm of the Reconstruction Errors as a Function of $log_{10}\frac{\mu}{\lambda}$ for Test Case 2.



(a) All cells (b) Smooth regions only

Figure 2.28: The 2-norm of the Reconstruction Errors as a Function of $log_{10}\frac{\mu}{\lambda}$ for Test Case 4.

2.10 Analysis

We summarize the results in Section 2.9. The wavelet based regularization, (2.8), is effective at suppressing noise and high frequency oscillations in smooth regions. Also, in the neighborhood of edges, the wavelet reconstruction is irregular, as shown in Figure 2.29, and edge induced oscillations are not as localized as those seen with (2.4) or with the total variation regularization. We now explain the reasons for this behavior.

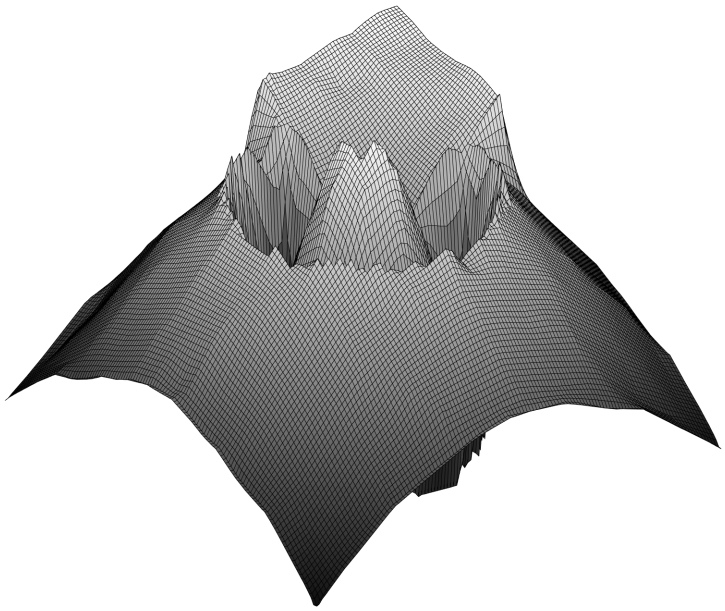


Figure 2.29: Artifacts Present in the $\mathbf{F}_{wav}$ Reconstruction.

2.10.1 Edge Response

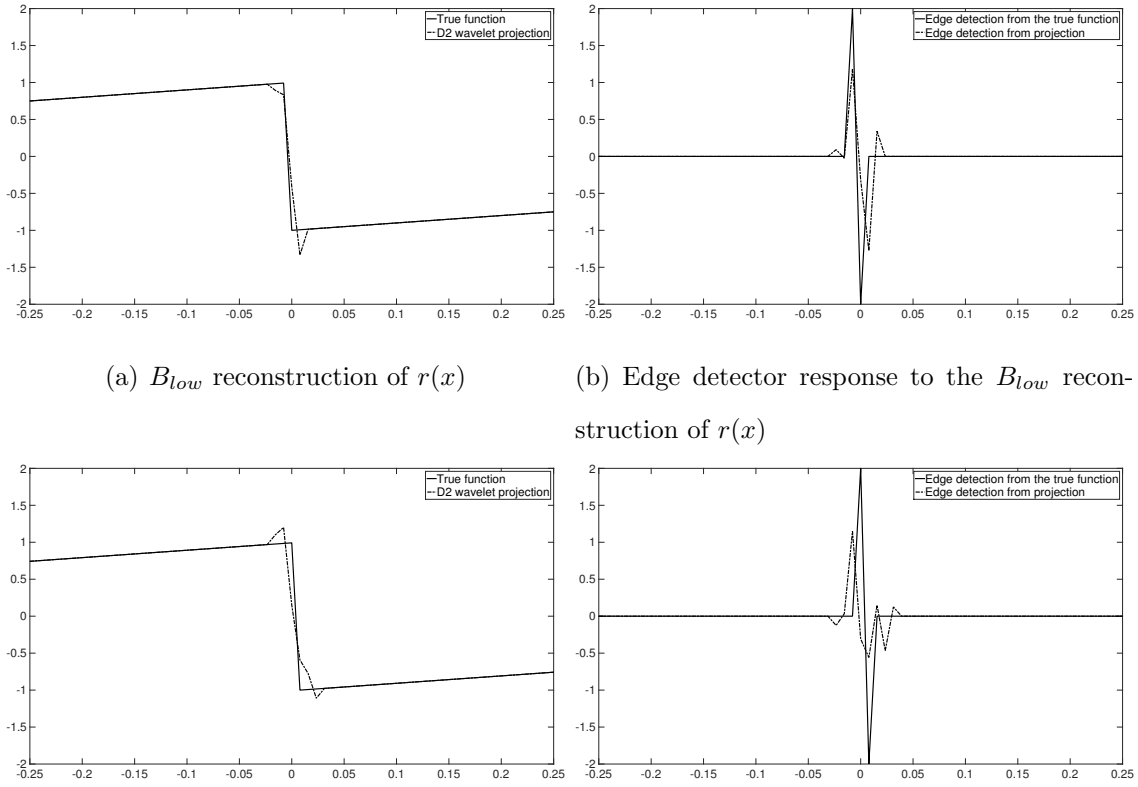
As discussed in Section 2.3, choosing the number of vanishing moments in the wavelet basis, P, results in the physical support of the B_{high} basis functions being $2P$. Increasing P, implies not only that each member of B_{high} is wider, but also that more members of B_{high} will intersect an edge as shown in Figure 2.2. Therefore, the projection of a discontinuity onto the B_{low} subspace results in a wide, high-order oscillatory reconstruction, which is expanded even more by the edge detector.

To explain the irregular edge reconstructions, we note that the Daubeschies wavelets are not symmetric. Thus, the reconstruction from the B_{low} projection of a piecewise smooth function is not translation invariant. We demonstrate this by examining the reconstruction of a ramp function given by

$$r(x) = \begin{cases} x+1 & -1 \leq x < 0 \\ x-1 & 0 \leq x < 1. \end{cases} \tag{2.19}$$

We let $\hat{r}$ be the $2N$ lowest Fourier coefficients of r resulting from the continuous

Fourier transform. In Figure 2.30, the B_{low} reconstruction of $r(x)$ is compared to the B_{low} reconstruction of $r(x - \frac{2}{N})$. We see that the shapes of the reconstructions are different and consequently the edge detector response to these reconstructions differ as well. This leads to (2.8) producing irregular edges based on the particular grid cell through which the edge passes. This lack of symmetry could be corrected by using a bi-orthogonal wavelet basis.



(a) B_{low} reconstruction of $r(x)$

(b) Edge detector response to the B_{low} reconstruction of $r(x)$

(c) B_{low} reconstruction of $r(x - \frac{2}{N})$

(d) $E_2\left(\mathcal{F}^{-1}\hat{r}\right)$

Figure 2.30: The B_{low} Reconstruction of $r(x)$ as Compared to the B_{low} Reconstruction of $r(x - \frac{2}{N})$ and the Associated Edge Detector Responses.

2.10.2 Gibbs Oscillation Suppression and Implicit Filtering

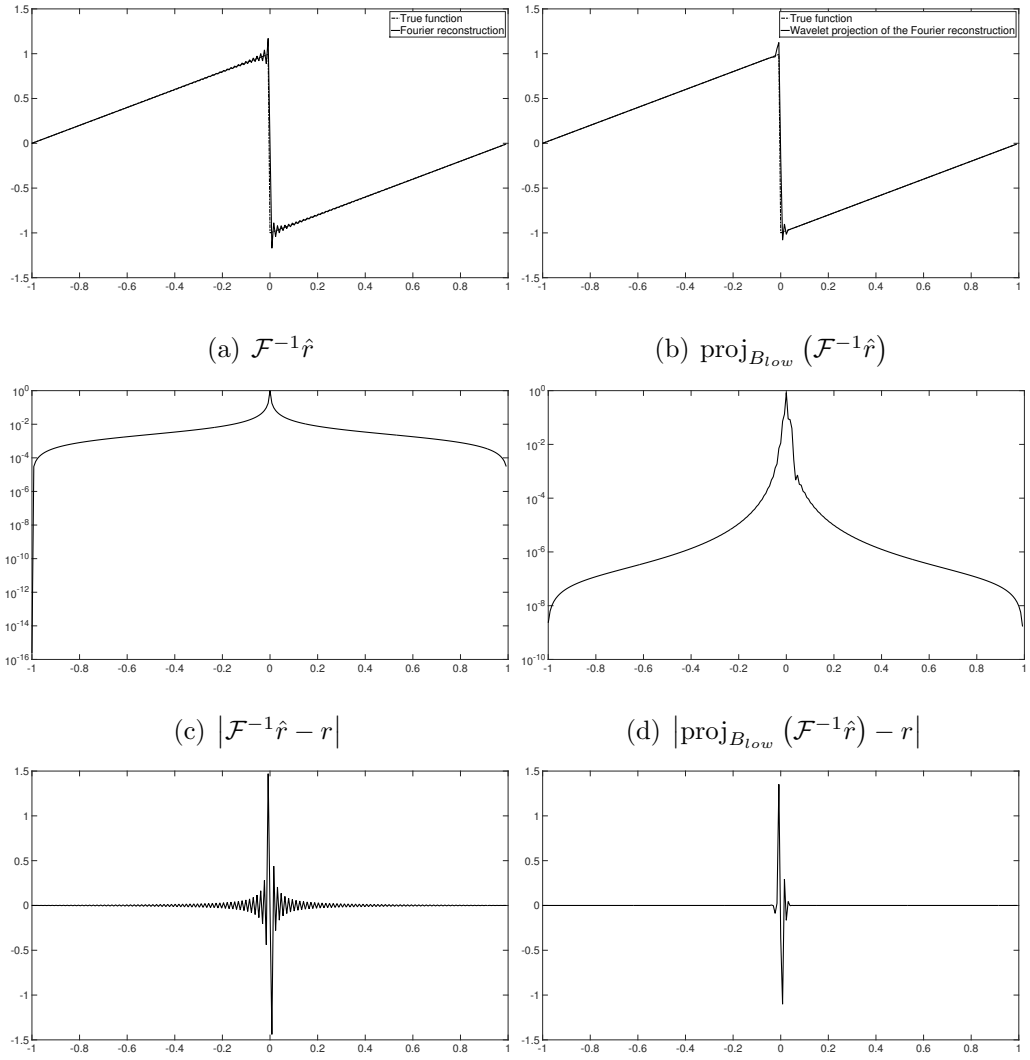
In addition to the reduced accuracy in the reconstruction caused by the slow decay of Fourier coefficients, the Gibbs phenomenon introduces rapid oscillations that

radiate from discontinuities. The results from Section 2.9 show that the wavelet based reconstruction is effective at suppressing these oscillations. This suppression of Gibbs related oscillations can result directly from the projection of the solution onto the span of the B_{low} basis, without the need for regularization. In Figure 2.31, the ramp function, $r(x)$, is shown along with its reconstruction from 128 Fourier modes. When the Fourier reconstruction is projected onto the B_{low} subspace of the D2 wavelets, the oscillations are dramatically reduced as would be seen with a spectral filter. The edge detector response of the projection is also confined to the neighborhood of the edge, rather than decaying slowly as occurs without the projection step.

This suppression of oscillations is similar to the application of a spectral filter. Consider the raised cosine filter, a second order spectral filter with a discrete formulation given by

$$\sigma_k^{rcos} = \frac{1}{2}\left(1 + \cos\left(\frac{2\pi k}{N}\right)\right). \tag{2.20}$$

Figure 2.32 shows the application of this filter to $\hat{r}$ has a reconstruction error almost identical to that resulting from the wavelet projection.

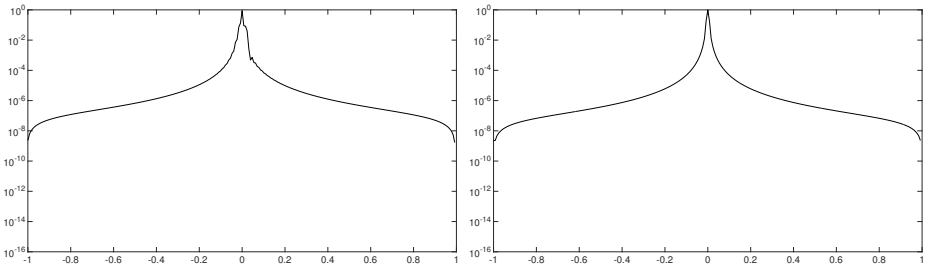


(a) $\mathcal{F}^{-1}\hat{r}$ (b) $\text{proj}_{B_{low}}\left(\mathcal{F}^{-1}\hat{r}\right)$

(c) $\left|\mathcal{F}^{-1}\hat{r} - r\right|$ (d) $\left|\text{proj}_{B_{low}}\left(\mathcal{F}^{-1}\hat{r}\right) - r\right|$

(e) $E_2\left(\mathcal{F}^{-1}\hat{r}\right)$ (f) $E_2\left(\text{proj}_{B_{low}}\left(\mathcal{F}^{-1}\hat{r}\right)\right)$

Figure 2.31: The Suppression of Gibbs Phenomenon As a Consequence of Projecting $r(x)$ onto the B_{low} Basis.



(a) $\left|\text{proj}_{B_{low}}\left(\mathcal{F}^{-1}\hat{r}\right) - r\right|$ (b) $\left|\left(\mathcal{F}^{-1}\left(\sigma_k^{r\cos} \circ \hat{r}\right)\right) - r\right|$

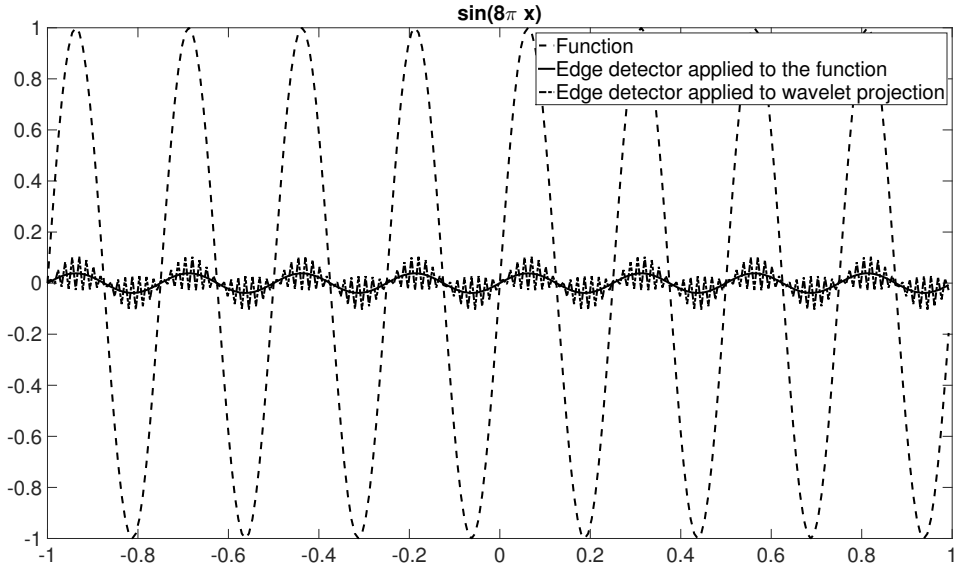
Figure 2.32: A Comparison of the Raised Cosine Filter and the Wavelet Projection.

2.10.3 Edge Detection Response for High Order Functions

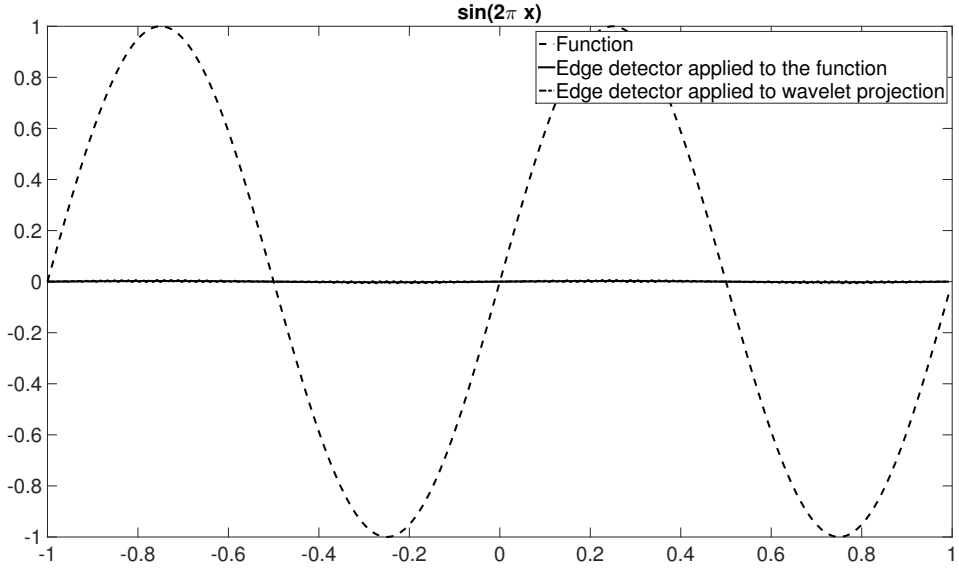
Another artifact seen in Figure 2.29 is that in smooth regions, the reconstruction is continuous, but contains abrupt slope changes. The polynomial annihilation edge detector operates by detecting when the function values evaluated at stencil points cannot be interpolated by a low order polynomial. When a function is smooth but has non-negligible derivatives of order higher than that of the detector there is a non-zero response from the detector, but this response tends to be smooth, i.e. there are no discontinuities in the first derivative of the edge detector response. When a similar function is projected onto the B_{low} subspace, the recontruction is smooth in areas that are well approximated by low order polynomials. In areas where the local Taylor series expansion has large high order terms, the B_{low} approximation fails by introducing discontinuities in the first derivative. This is further amplified by the edge detector, as shown in Figure 2.33. In Figure 2.33, we see the effect of the edge detector when applied directly to the function $\sin(8\pi x)$, which has large Taylor series coefficients beyond the third term. The response from the edge detector is smooth. After wavelet projection, the edge detector has a larger and non-smooth response. For the function $\sin(2\pi x)$, the wavelet projection has little impact on the edge detector response because of the rapid decay in Taylor series coefficients.

2.10.4 Concluding Remarks

From the test results, the main advantage of (2.8) over prior methods such as (2.4) is robustness with respect to the selection of the regularization parameter, $\frac{\lambda}{\mu}$, and the innate inclusion of a spectral filter in the regularization that suppresses high frequency oscillations. Nonetheless, in smooth regions the overall accuracy of (2.8) is very similar to that of (2.4) and small variations in the regularization parameter tend



(a) $\sin(8\pi x)$



(b) $\sin(2\pi x)$

Figure 2.33: The Effect of the B_{low} Projection on High-order Smooth Functions.

to favor one method over the other. In the neighborhood of edges, prior methods are superior at localizing oscillations. This negative aspect of the reconstruction with (2.8) outweighs the benefit in terms of robustness. Switching to the bi-orthogonal wavelet basis would most likely help with the edge reconstructions. Techniques for

including some members of the B_{high} set based on edge recovery may also result in better edge reconstructions, but there is no overwhelming advantage to (2.8) to justify additional processing steps compared to (2.4).

Chapter 3

EDGE DETECTION OF PIECEWISE SMOOTH FUNCTIONS FROM UNDER-SAMPLED FOURIER DATA

Several important applications including magnetic resonance imaging (MRI) and synthetic aperture radar (SAR) acquire data by way of Fourier sampling Yan (2002); Gor (2010); Cheney and Borden (2009); Richards *et al.* (2010). To reduce data acquisition times, some MRI techniques forgo collection of frequency data on a uniform Cartesian grid, opting instead to collect information on trajectories that densely sample low frequencies and sub-sample high frequencies Pipe (1999b); Zuo *et al.* (2006); Li *et al.* (2015); Ilievska and Ivanovski (2011). Recovering edge information from this data is useful in a variety of situations, such as producing edge maps that can be interpreted directly by practitioners, as a source of data for feature detection and categorization, or as data that can be used in combination with other processing techniques to enhance image recovery.

The concentration factor edge detection method recovers edge information directly from acquired Fourier data, Gelb and Tadmor (1999). It operates by applying a set of concentration factors to the supplied Fourier samples. Starting with a complete, band limited set of Fourier samples the recovered "edge map", or synonymously "jump map", exhibits a corresponding response at the location of true edges, but also exhibits oscillations that decay away from edges. Isolating the true edges requires some form of thresholding. The particular structure of these oscillations is dependent on the chosen set of concentration factors, and in Gelb and Tadmor (2006) results from different concentration factors were combined with the minmod algorithm to remove some of these oscillations. Unfortunately, additional oscillations occur when

the sampled Fourier data are noisy or in cases where the band of Fourier coefficients is not adequately resolved. Unless removed by additional thresholding, these oscillations in smooth regions translate into false positive edge detections. But the additional thresholding may cause failure in identifying true edges.

In this chapter we demonstrate that in spite of these false positives, there is valuable information contained in the structure of the oscillations in the neighborhood of jumps that can be extracted to recover the edges of the underlying image. In particular we note that at jump locations and in smooth regions the different jump approximations are similar, while in the neighborhood of jumps there is an exploitable variance in behavior. On measuring this variance, we note a "two peak" signature surrounding jumps that is not generally present around spurious oscillations not in the vicinity of edges. Thus we develop a new algorithm to detect these variance signatures as a way of filtering out false positives from the edge map approximation.

The filtering method based on the variance in behavior between concentration factors succeeds in eliminating false positives detections, that are the result of oscillations inherent in the concentration method edge approximations. Nonetheless, false edges in the Fourier reconstruction from incomplete or noisy Fourier data may become indistinguishable from true edges without additional prior information. To eliminate these false edges, we call upon the assumption that the underlying function is piecewise smooth with a sparse edge map. In Archibald *et al.* (2015), piecewise smooth functions were reconstructed from under-sampled and noisy Fourier data using ℓ^1 regularization to promote the sparsity of edges. In spite of sub-sampling and noise, smooth regions are recovered accurately with this method. To achieve this accuracy, the sparsifying transform must distinguish smooth regions with high regularity from edges. In particular, TV cannot be used for the sparsifying transform because of the resulting "staircasing effect". Instead we use the PA transform of order 2, as

developed in Archibald *et al.* (2015), which yields faster convergence in the smooth regions. With the ability to accurately reconstruct smooth regions from incomplete data, we introduce treatments involving subsets of the already sub-sampled Fourier data set. The variance in the reconstruction from these subsets exhibits a similar "two peak" signature, which can be used to eliminate false positive edge detections. We develop a new method that validates edges generated by the concentration factor edge detection technique based on their presence inbetween regions of high variance. The important advantage of this method is that false jumps resulting from the incomplete and noisy Fourier samples are distinguishable from true jumps based on the regularized reconstruction.

The rest of the chapter is organized as follows. In Section 3.1 we review the necessary background and establish the test case framework. In Section 3.2 we examine the one dimensional results of methods based on the concentration factor edge detector, developed in Gelb and Tadmor (1999), demonstrate the structure of the concentration factor approximation, and propose a new method of edge detection based on the variance in approximations associated with different concentration factors. We then introduce a new method for detecting edges based on the variance in regularized reconstruction. In Section 3.3 we extend the results to two dimensions. Concluding remarks are provided in Section 3.4.

3.1 Preliminaries

Let f be a periodic piecewise smooth function defined on $[-1, 1)$. We define the corresponding jump function, $[f]$, as the difference between the right-hand and left-hand limits of the function i.e.

$$[f](x) = f\left(x^+\right) - f\left(x^-\right). \tag{3.1}$$

Thus, in smooth regions $[f] = 0$ and at discontinuities $[f]$ is the value of the jump. Since our ultimate goal is to construct an edge map of f, we discretize x uniformly as $D = \{x_j = \frac{j}{N}\}_{j=-N}^{N}$. We note that the jump function approximation method described here does not restrict the solution to uniform points, but using this distribution may improve its computational efficiency. We also make the assumption that there is at most one jump discontinuity within a cell $I_j = [x_j, x_{j+1}]$, which is reasonable when data are acquired as the first $2N + 1$ Fourier coefficients, or some sparse subset of those values. Thus, if $[f](x_j)$ is the value of the jump occurring in I_j, we can write

$$[f](x) = \sum_{j=-N}^{N-1} [f](x_j)\chi_{I_j}(x), \tag{3.2}$$

where $\chi_{I_j}(x)$ is defined as

$$\chi_{I_j}(x) = \begin{cases} 1 & \text{if } x \in I_j \\ 0 & \text{for all other } x. \end{cases} \tag{3.3}$$

For simplicity, the numerical algorithms used in this investigation all place the jump discontinuity at the left boundary of its corresponding cell.

To demonstrate basic features of the algorithms discussed in this chapter, we will consider the following function, displayed in Figure 3.1.

$$f_1(x) = \begin{cases} 2 - \frac{\pi x}{3} + \sin\left(\pi x + \frac{1}{16}\right) & -\frac{3}{4} \leqslant x < \frac{3}{16} \\ 4\pi x - 9 & \frac{1}{2} \leqslant x < \frac{15}{16} \\ 0 & otherwise, \end{cases} \tag{3.4}$$

where

$$[f_1](x) = \begin{cases} 2.04 & x = -\frac{3}{4} \\ -2.41 & x = \frac{3}{16} \\ -2.72 & x = \frac{1}{2} \\ 2.71 & x = \frac{15}{16}. \end{cases}$$

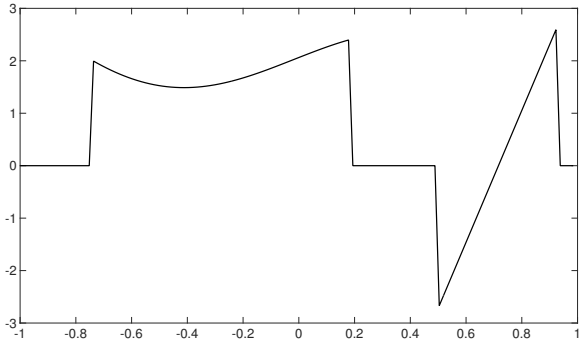


Figure 3.1: $f_1(x)$.

3.1.1 The Concentration Factor Edge Detection Method

For algorithmic development, let us assume that f has a single discontinuity at $x = \xi$, $\xi \in (-1, 1)$. Suppose we are given a (sparse) subset of Fourier coefficients,

$$\hat{f}(k) = \frac{1}{2}\int_{-1}^{1} f(x)\, e^{-ik\pi x} dx, \quad -N \le k \le N. \tag{3.5}$$

Repeated integration by parts provides the relationship

$$\hat{f}(k) = \frac{1}{2\pi}\left(\frac{[f](\xi)}{ik} + \frac{[f'](\xi)}{(ik)^2} + \frac{[f''](\xi)}{(ik)^3} + \dots\right) e^{-ik\pi\xi} = \frac{e^{-ik\pi\xi}}{2\pi ik} + \mathcal{O}\left(\frac{1}{k^2}\right), \tag{3.6}$$

where $[f](\xi)$ is the value of the jump at $x = \xi$. In Gelb and Tadmor (1999), (3.6) was used to generate the *concentration factor edge detection method*, given by

$$S_N^{\sigma}[f](x) = i\sum_{k=-N, k\neq 0}^{N} \hat{f}(k)\mathrm{sgn}(k)\sigma\left(\frac{|k|}{N}\right)e^{ik\pi x}, \tag{3.7}$$

where the "concentration factor" $\sigma(\eta)$, $\eta \in [0, 1]$, discretized at $\left(\frac{|k|}{N}\right)$, satisfies the following admissibility requirements:

1. $\frac{\sigma(\eta)}{\eta} \in C^2(0, 1)$
2. $\int_{\varepsilon}^{1} \frac{\sigma(\eta)}{\eta} \to -\pi$ where ϵ is small.

Some examples of admissible concentration factors are listed in Table 3.1, while the results of $S_N^{\sigma_G}[f_1](x)$ for $N = 32, 64$, and 128 are shown in Figure 3.3(a).

Table 3.1: Sample Concentration Factors

Concentration factor type	expression	details
Trigonometric	$\sigma_G(\eta) = \frac{\pi \sin(\pi\eta)}{\mathrm{Si}(\pi)}$	$\mathrm{Si}(\pi) = \int_0^\pi \frac{\sin x}{x} dx$
Polynomial	$\sigma_P^p(\eta) = p\pi\eta^p$	p is the order of polynomial concentration factor.
Exponential	$\sigma_E^\alpha(\eta) = C\eta e^{\frac{1}{\alpha\eta(\eta-1)}}$	α is the order of the exponential concentration factor. C is a scale factor given by $C = \frac{\pi}{\int_{1/N}^{1-1/N} e^{\frac{1}{\alpha\tau(\tau-1)}} d\tau}$

Due to the relationship obtained in (3.6), the concentration factor edge detection method in (3.7) can be characterized using the ramp function $r_\xi(x) = r(x-\xi)$, where

$$r(x) = \begin{cases} -\frac{x+1}{2} & -1 \le x < 0 \\ -\frac{x-1}{2} & 0 \le x < 1, \end{cases} \tag{3.8}$$

and

$$[r_\xi](x) = \begin{cases} 1 & x = \xi \\ 0 & x \neq \xi. \end{cases} \tag{3.9}$$

The corresponding Fourier coefficients of $r_\xi(x)$ are

$$\widehat{r_\xi}(k) = \begin{cases} \frac{e^{-ik\pi\xi}}{2ik\pi} & k \neq 0 \\ 0 & k = 0. \end{cases} \tag{3.10}$$

Using the above notation, a piecewise linear approximation of piecewise smooth f with a single discontinuity at $x = \xi$ can now be written as

$$f(x) \approx [f](\xi) r_\xi(x). \tag{3.11}$$

As shown in (3.6), the jump value associated with the j^{th} derivative of f, $[f^{(j)}]$, leads to terms in $\hat{f}_k$ that decay as $\frac{1}{k^{j+1}}$. Conversly, the concentration factors in Table

3.1 yield small values for $\sigma(\frac{|k|}{N})$ when k is small. Thus the jumps, $[f]$ and $[r]$, are of primary importance in characterizing the concentration factor edge detection method.

Analogously to (3.11), if f has jump discontinuities at ξ_m, $m = 1, \cdots M$, the linear approximation of f is

$$f(x) \approx R(x) = \sum_{m=1}^{M} [f](\xi_m)\, r_{\xi_m}(x). \tag{3.12}$$

Substituting the corresponding (multiple jump) values of $\widehat{r_\xi}(k)$ in (3.10) into $\hat{f}_k$ in (3.6), it is evident that (3.7) approximates $[R](x)$. Moreover, by translating each discontinuity location a distance ξ_m, we can further characterize (3.7) by defining

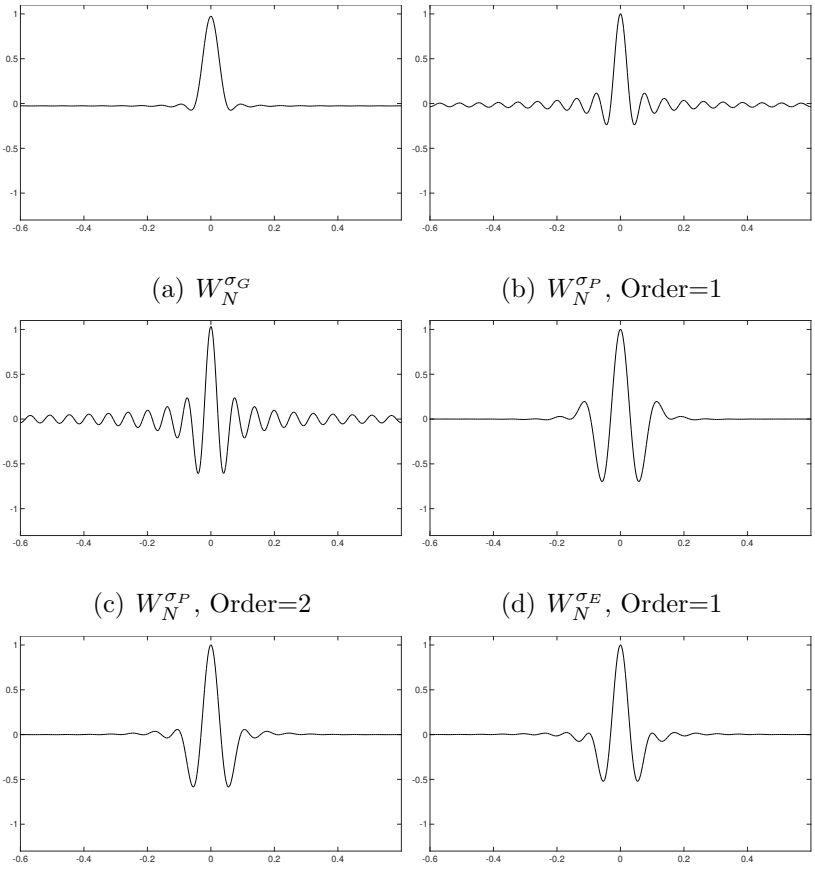
Definition 1 (The jump response)**.** *Given concentration factor σ, we define the jump response as*

$$W_N^\sigma(x) := S_N^\sigma[r](x) = \frac{1}{2\pi} \sum_{0<k\le N} \frac{\sigma\left(\frac{|k|}{N}\right)}{|k|} e^{ik\pi x}. \tag{3.13}$$

Figure 3.2 shows the jump response using various specific concentration factors. It demonstrates that in the neighborhood of a jump discontinuity, the behavior of (3.13) is highly dependent on the particular concentration factor chosen, while away from discontinuities the convergence rate is $\mathcal{O}(\frac{1}{N})$. These different behavior patterns have been exploited in Gelb and Tadmor (2006), where post processing algorithms were developed to pinpoint the edges. While such techniques were shown to be effective given all $2N + 1$ Fourier samples in (3.5), we will demonstrate that they are not as robust when the data are noisy or further sub-sampled. Instead we will employ an l^1 regularization technique that exploits the sparsity of $[f](x)$, as observed by the sparse number of non-zero coefficients in (3.2). We first review l^1 regularization below.

3.1.2 Sparsity Promoting Regularization

Regularization is a well-known technique for solving ill-conditioned inverse problems, and regularizations using ℓ^1 penalty terms have enjoyed great success in the



(a) $W_N^{\sigma_G}$ (b) $W_N^{\sigma_P}$, Order=1

(c) $W_N^{\sigma_P}$, Order=2 (d) $W_N^{\sigma_E}$, Order=1

(e) $W_N^{\sigma_E}$, Order=2 (f) $W_N^{\sigma_E}$, Order=3

Figure 3.2: Sample Jump Response Behavior for Various Concentration Factors with $N = 32$.

fields of image processing and compressed sensing, Rudin *et al.* (1992); Yin *et al.* (2008). A driving principle in many of these techniques is that some attribute of a good solution will be sparse. An ideal ℓ^0 penalty term would seek to minimize the count of non-zero entries in its argument, but ℓ^0 terms are not convex and a general numerical solution is impossible. Hence ℓ^1 terms are used as convex relaxations for ℓ^0 terms in compressed sensing, Candès *et al.* (2006). For the algorithms described below we will use regularizations of the general form

$$\vec{u}^* = \underset{\vec{u}}{\operatorname{argmin}} \left(\left\| \mathcal{G}\vec{u} - \mathcal{H}\vec{b} \right\|_2^2 + \sum_j \lambda_j \|\phi_j(\vec{u})\|_1 \right). \tag{3.14}$$

Here $\vec{u}^*$ is the desired solution, $\vec{b}$ is the provided data, $\mathcal{G}$ and $\mathcal{H}$ are optional operators to make trial solutions, $\vec{u}$, compatible with the provided data, $\vec{b}$, $\{\phi_j\}$ is a set of sparsifying operators, and $\{\lambda_j\}$ is a set of regularization parameters weighting individual penalty terms. For the algorithms described below the operators $\mathcal{G}$, $\mathcal{H}$ and $\{\lambda_j\}$ will be linear and the minimization problem in two dimensions can be efficiently solved using the Split-Bregman algorithm, Goldstein and Osher (2009).

3.1.3 Polynomial Annihilation Edge Detection As a Sparsifying Operator

We now review the polynomial annihilation method that was introduced in Archibald *et al.* (2005) and used as a sparsifying operator in a penalty term in Archibald *et al.* (2015); Stefan *et al.* (2010); Wasserman *et al.* (2015) as a means for reconstructing functions from sparse or noisy Fourier data. Although the method was developed for multiple dimensions and non-uniform data, we have found that for a uniform grid it is most efficient to employ the polynomial annihilation operator dimension by dimension, Archibald *et al.* (2015); Wasserman *et al.* (2015). Thus we describe the technique for given data $f(x_j)$, $x_j = \frac{j}{N}$, $j = -N, \cdots, N$. In practice $f(x_j)$ will be the elements of the solution vector $\vec{u}$ in (3.14). The polynomial annihilation edge detection method is defined as

$$L^p f(x) = \frac{1}{q^p(x)} \sum_{x_j \in S} c_j(x) f(x_j),$$

where $c_j(x)$ are polynomial annihilation coefficients, $q^p(x)$ is a normalization factor, and S_x is a set of $p+1$ grid points surrounding x, which can be extended periodically as necessary.[1] The polynomial annihilation coefficients, $c_j(x)$, are designed to annihilate

[1]The polynomial annihilation method does not restrict the class of underlying functions to be periodic. Indeed, the stencils S_x can be made one sided as the boundaries of the domain are approached.

polynomials up to degree p, and are obtained by solving the system

$$\sum_{x_j \in S_x} c_j(x)\varphi_\ell(x_j) = \varphi_\ell^{(p)}(x),\ j = 1, \ldots, p+1, \tag{3.15}$$

where φ_ℓ, $\ell = 0, \ldots, p$, is a basis of for the space of polynomials of degree $\leq p$. With the restrictions just described, the solution to (3.15) is given by

$$c_j = \frac{p!}{\prod_{k=1\ldots p+1, k\neq j} (j-k)\, h}, \tag{3.16}$$

where h is the distance between grid points $\frac{1}{N}$. Thus, in the uniform case, c_j is independent of $x_k \in D$. The normalization factor, $q^p(x)$, assures the proper convergence of $L^p f(x)$ to the jump value at each discontinuity and is given by

$$q^p(x) = \sum_{x_j \in S_x, x_j \geqslant x} c_j(x). \tag{3.17}$$

When f is periodic, the uniform distribution of grid points implies that q^p is also independent of choice of x_k. We can therefore define the polynomial edge detection operator $\mathbf{P}^p$ as a circulant matrix such that

$$(\mathbf{P}^p \vec{v})_k = \frac{1}{q^p} \sum_{j=1}^{p+1} c_j \vec{v}_{k+j-\frac{2+p}{2}}. \tag{3.18}$$

For example, when $p = 2$ we have

$$\mathbf{P}^2 = \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & -1 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 2 & -1 \\ -1 & 0 & \cdots & 0 & -1 & 2 \end{pmatrix}.$$

3.1.4 Sample Sets

We assume the Fourier coefficients of a piecewise periodic smooth function, f, are drawn from a set $\hat{\mathbf{U}} = \left\{ \hat{f}_k : -N \leqslant k \leqslant N \right\}$, and that these Fourier coefficients may be subject to complex Gaussian noise. Each test case is characterized by a set of parameters. We let SNR represent the signal to noise ratio, with SNR $= \infty$ representing no noise. The initial set of coefficients, $\hat{\mathbf{U}}$, may be sub-sampled as controlled by parameters γ, β, and ζ to form

$$\hat{\mathbf{V}} = \hat{\mathbf{F}} \cup \hat{\mathbf{R}}, \tag{3.19}$$

where $dim(\hat{\mathbf{V}}) = \gamma(2N+1)$,

$$\hat{\mathbf{F}} = \left\{ \hat{f}_k : -\beta N \leq k \leq \beta N \right\},$$

with $0 \leq \beta \leq 1$, and $\hat{\mathbf{R}}$ is a set of randomly selected coefficients $\hat{f}_k$, $|k| > \beta N$ ($|k| \leq N$ when $\beta = 0$) from a normal distribution.[2]

Now, for each test we construct $q = 1, \cdots, Q$ subsets from $\hat{\mathbf{V}}$ as

$$\hat{\mathbf{T}}_q = \hat{\mathbf{F}} \cup \hat{\mathbf{R}}_q, \tag{3.20}$$

where $\hat{\mathbf{R}}_q$ has $\zeta(\gamma - \beta)(2N+1)$ randomly selected values of $\hat{\mathbf{R}}$. We will apply the numerical algorithms in this investigation to a set of test cases with parameters specified in Table 3.2.

Different values of N will demonstrate the various features of our new algorithm. In applications where the sampling is limited to the small wave numbers, that is small N, we do not expect to use much compression. However, as discussed in the

[2]Numerical experiments were also performed for uniformly distributed sets with no noticeable difference for $\beta \geq .3$ and various choices of γ. However, for $0 < \beta < .3$ and $\gamma << 1$, it becomes imperative to keep the low modes of the given data set.

Table 3.2: Parameters Selected for Each Test Case. Unless otherwise Specified, We also Choose $N = 64$, $Q = 30$ and $\zeta = .5$.

Test Case	SNR	β	γ
1	∞	0	1
2	∞	0.3	0.75
3	12.5dB	0.3	1
4	12.5dB	0.3	0.75

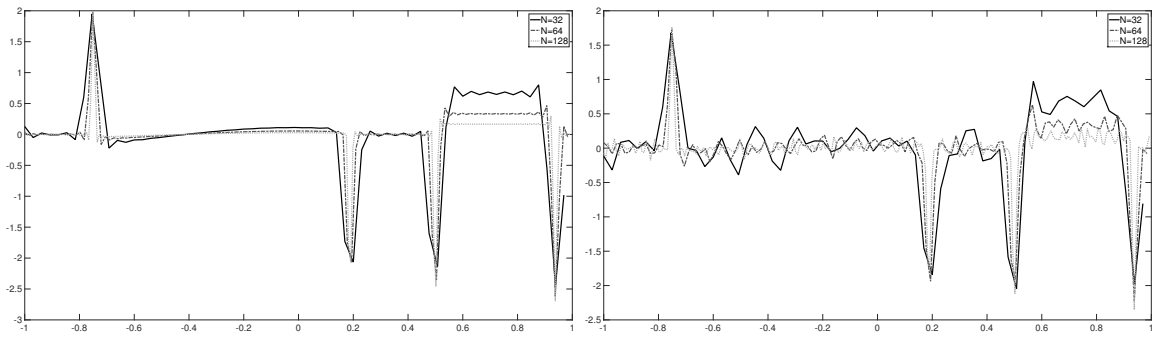
introduction, in applications such as MRI, some sampling trajectories are designed to oversample the low frequency modes while sparsely sampling in the high frequency range. Moreover, if very large data sets are obtained, compression could be used to process the data more efficiently. Our numerical experiments give insight into both situations. Similarly, we demonstrate that our method is robust when the given data are noisy.

3.2 Edge Detection for Under-sampled Fourier Data

3.2.1 The Concentration Factor Edge Detection Method

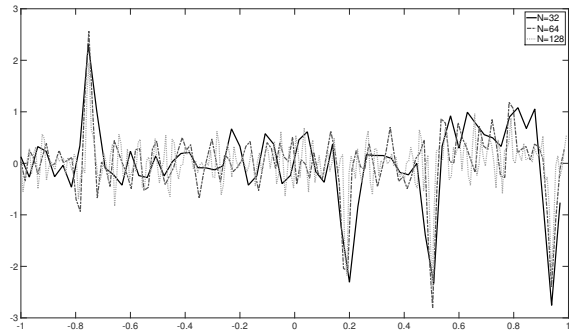
We first examine the direct application of the concentration factor edge detection method, given in (3.7). As shown in Figure 3.3(a), this method works well when given the full set of $2N+1$ noiseless Fourier coefficients, but it loses its effectiveness as either the signal to noise ratio decreases, (Figure 3.3(c)), or the number of available Fourier samples decreases, (Figure 3.3(b) and (d)). Oscillations appear in the smooth regions, and as they increase in magnitude, effective thresholding becomes more difficult.

The minmod algorithm was used in Gelb and Tadmor (2006) to exploit the variability in the jump responses apparent in Figure 3.2. Specifically, the results in (3.7)

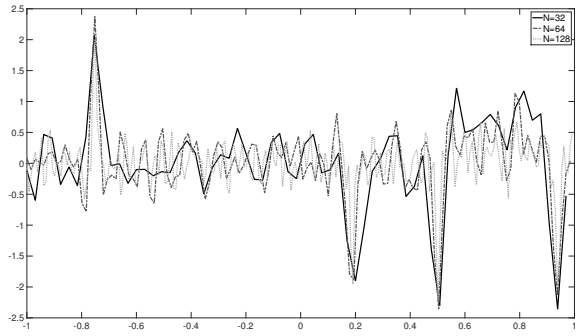


(a) test case 1

(b) test case 2



(c) test case 3



(d) test case 4

Figure 3.3: Concentration Factor Edge Detection Method, $S_N^{\sigma_G}[f_1](x)$.

using admissible concentration factors, $\sigma_1, \cdots, \sigma_n$, were jointly post processed as

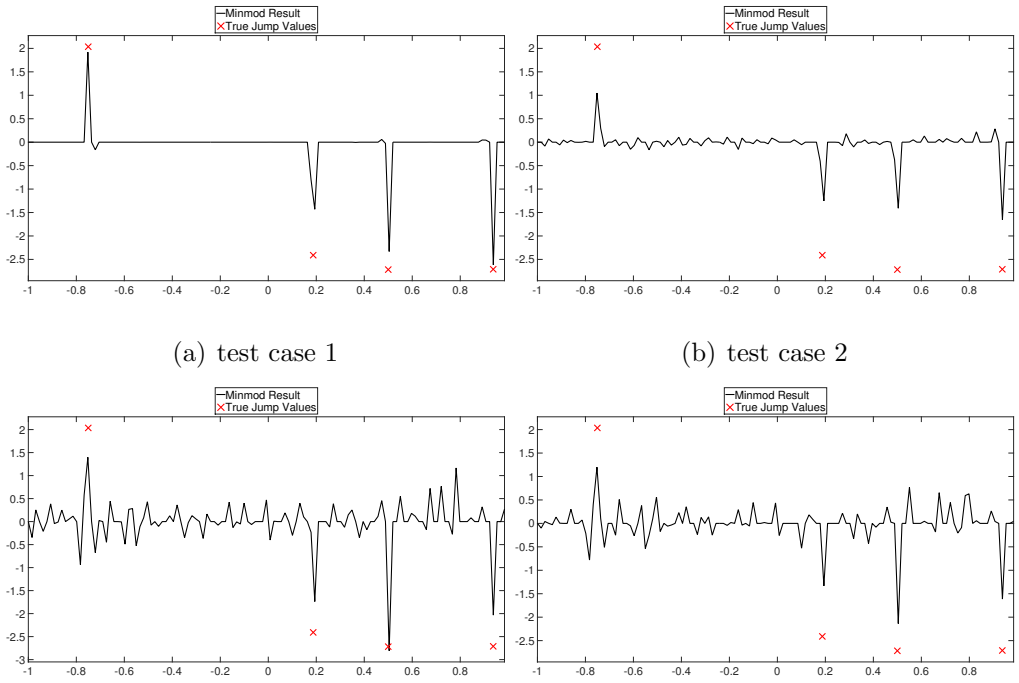
$$S_N^{MM}[f](x) = \text{minmod}\left\{S_N^{\sigma_1}[f](x), S_N^{\sigma_2}[f](x), \ldots, S_N^{\sigma_n}[f](x)\right\}, \tag{3.21}$$

where

$$\text{minmod}\left\{a_1, a_2, \ldots a_n\right\} = \begin{cases} s\min\left(|a_1|, |a_2|, \ldots |a_n|\right) & \text{if } \operatorname{sgn}(a_1) = \ldots \operatorname{sgn}(a_n) = s \\ 0 & \text{otherwise.} \end{cases}$$

The results of (3.21) using the set $\{\sigma_G, \sigma_P^1, \sigma_P^2, \sigma_E^1, \sigma_E^2\}$ are displayed in Figure 3.4. Although some artificial oscillations are reduced, these improvements become less evident as more noise and less data are used. In some cases, adding more concentration factors in (3.21) may help in further reducing the magnitude of the oscillations. That said, although (3.21) is not an effective means of differentiating the results using various concentration factors in these cases, it still is possible that this variation can be

exploited in a different way. In Section 3.2.4 we introduce a new algorithm with this in mind.



(a) test case 1

(b) test case 2

(c) test case 3

(d) test case 4

Figure 3.4: $S_N^{MM}\left[f_1\right](x)$ in (3.21).

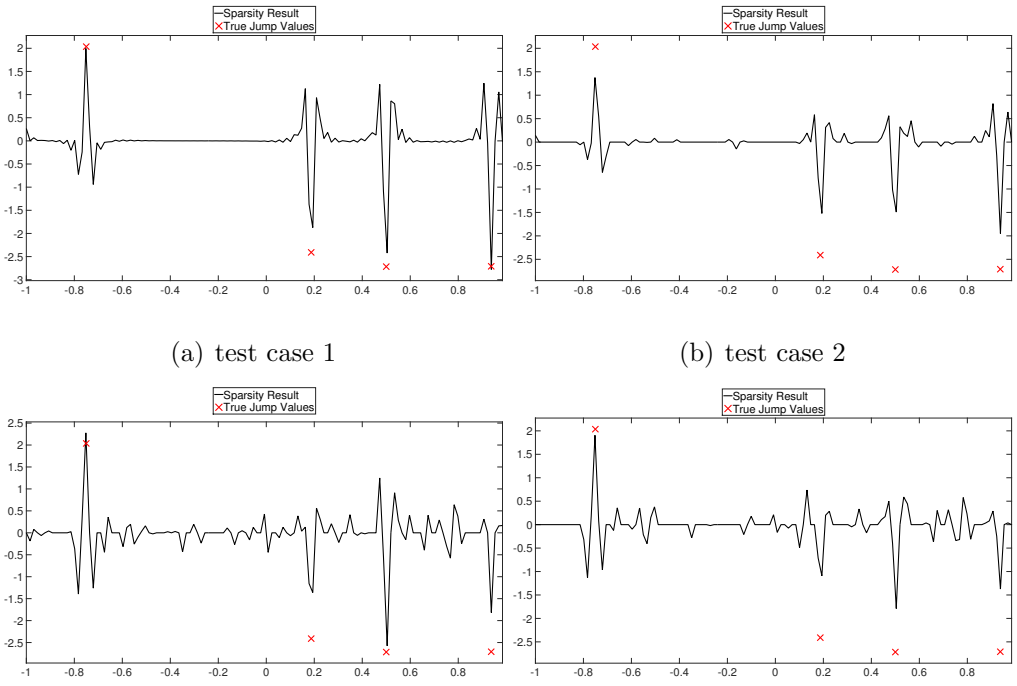
3.2.2 The Sparsity Enforcing Edge Detection Method

The sparsity enforcing edge detection method for determining edges in noisy and/or sub-sampled environments using regularization was developed in Stefan *et al.* (2012) and given by

$$S_N^{SE}\left[f\right] = \underset{\vec{g}}{\operatorname{argmin}} \left(\|\vec{g} - S_N^{\sigma_\sigma}\left[f\right]\|_2^2 + \lambda \|\vec{g}\|_1 \right). \tag{3.22}$$

Here $\vec{g}$ is a vector discretized on a physical-space grid, $\left\{x_j = \frac{j}{N}\right\}_{j=-N}^{N}$.

Figure 3.5 displays the results of (3.22) using σ_G. Comparing Figures 3.3, 3.4, and 3.5, we see that (3.22) yields better results than (3.7) when given all $2N+1$ noiseless coefficients, although post processing with the minmod algorithm, (3.21), is more



(a) test case 1 (b) test case 2

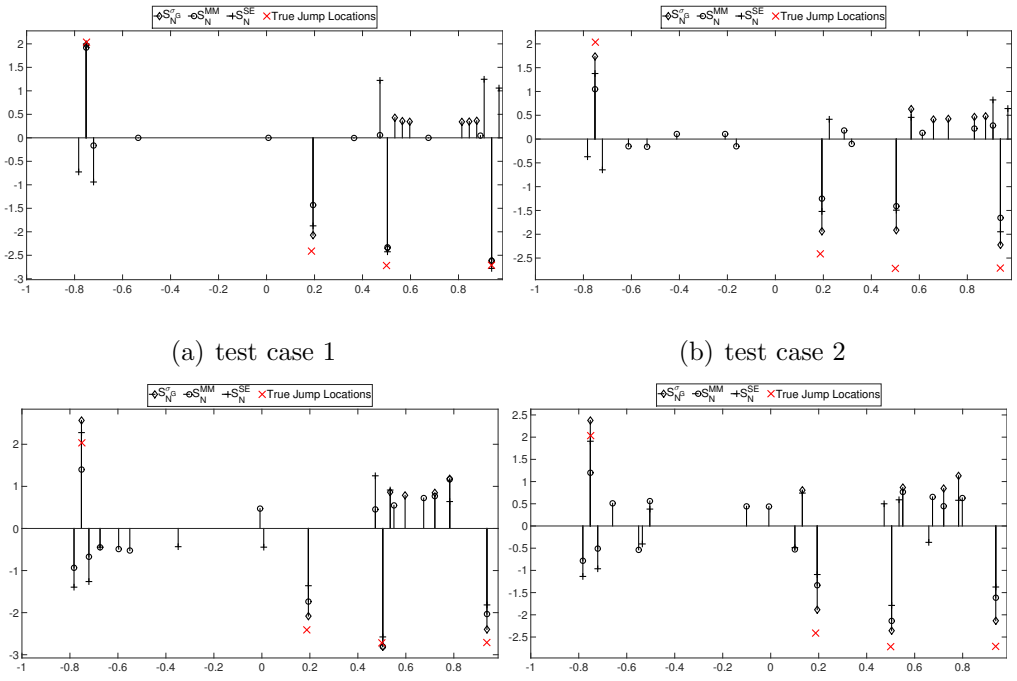
(c) test case 3 (d) test case 4

Figure 3.5: Edge Map of $f_1(x)$ Resulting from (3.22).

effective. However, using the regularization in (3.22) is more robust in cases where data are noisy and under-sampled, although the results inherently depend on the regularization parameter chosen. Thus, we observe a tradeoff between artificial jump suppression and the failure to detect true edges. Nonetheless, this method provides an effective jump response that varies from those generated using the concentration factor edge detection method in (3.7), which may be exploited in the procedure described below.

3.2.3 Generating an Edge Map Using Thresholding

Let us call $E[f]$ the result acquired either by (3.7), (3.21), or (3.22). It is evident that some degree of post processing is needed to reduce the number of false positives and form an accurate edge map. One way to do this is to use thresholding. Rather than prescribe a cut off value related to the $|E[f](x)|$, which may not be robust with



(a) test case 1

(b) test case 2

(c) test case 3

(d) test case 4

Figure 3.6: Algorithm 1 Using $S_N^{\sigma_G}[f_1](x)$, $S_N^{MM}[f_1](x)$, and $S_N^{SE}[f_1](x)$ and Threshold $c = 7/8$.

respect to noise or sub-sampling, we appeal to the presumed sparsity of the underlying jump function, $[f]$, to choose a threshold, $\epsilon = \epsilon(c)$, such that $\dim(\{|E[f](x)| < \epsilon\}) = c(2N+1)$. Typically $c \in (\frac{9}{10}, 1)$. Although the corresponding threshold is seemingly large, it is reasonable under the assumption that $[f]$ has a sparse representation, and that there should be relatively few values of $|E[f](x)| > \epsilon$. Additionally, only the extrema of $E[f](x)$ are considered as jumps in the post processed jump function approximation, $E^{PP}[f](x)$, and these extrema are separated by a distance of at least d. That is, we define each local jump region to have a distance of d, where d is dependent on the noise and amount of sub-sampling. The process of thresholding and extrema detection can then be combined as shown in Algorithm 1.

Figure 3.6 compares the thresholded edge maps generated by $S_N^{\sigma_G}[f_1](x)$, $S_N^{MM}[f_1](x)$, and $S_N^{SE}[f_1](x)$, with $c = \frac{7}{8}$. We note that a larger value for c would isolate the four

Algorithm 1 Generating an Edge Map Using Thresholding

Choose a jump function approximation method to construct $E[f]$, a threshold $\epsilon = \epsilon(c)$ as described above, and a distance d which defines the distance of each local jump region. Although not necessary for periodic functions, for simplicity assume that no jumps occur between $[x(-N), x(-N+d)]$ and $[x(N-d), x(N)]$.

for $j = -N + d, \cdots, N - d$

if $|E[f](x_j)| > \epsilon$ and $|E[f](x_j)| > |E[f](x_l)|$ for $|x_j - x_l| < d$, then

- $E_N^{PP}[f](x_j) = E[f](x_j)$

else

- $E_N^{PP}[f](x_j) = 0$

end if

end for

true jumps, but as can be seen in test case 4, the scale of false jumps approaches that of true jumps. Without advance knowledge of the true jump heights, increasing the threshold, c, risks eliminating true jumps.

3.2.4 Variance of Jump Function Responses

As is evident from Figure 3.6, simple thresholding becomes less effective as noise is increased or as the data are further under-sampled. We now introduce a new method for detecting edges by defining the variance vector of the set of jump function approximations, $\mathbf{E} = \{\overrightarrow{E_1 f}, \cdots, \overrightarrow{E_n f}\}$, where each $\overrightarrow{E_\ell f}$ is calculated on a set of points

$\vec{x} = \{x_j\}_{j=-N}^{N}$, as[3]

$$\vec{v}(\mathbf{E})_n = \frac{1}{dim(\mathbf{E})} \sum_{E_j \in \mathbf{E}} (\overrightarrow{E_j f}_n - \frac{1}{dim(\mathbf{E})} \sum_{E_k \in \mathbf{E}} \overrightarrow{E_k f}_n)^2, \quad n = 1 \ldots 2N+1. \tag{3.23}$$

From the previous discussions, possible sets of jump function approximations evaluated at grid points $x_j, j = -N, \cdots, N$, include

$$\mathbf{E_1} = \{\overrightarrow{S_N^{\sigma_k} f}\}_{\sigma_k \in G}, \tag{3.24a}$$

$$\mathbf{E_2} = \{\overrightarrow{S_N^{\sigma_k} f}\}_{\sigma_k \in G} \cup \overrightarrow{S_N^{SE} f}, \tag{3.24b}$$

where G is a set of concentration factors and $S_N^{SE}[f]$ is given in (3.22). The result of (3.23) with $\mathbf{E_1}$ for $f(x) = r(x)$ is displayed in Figure 3.7. It is evident that in regions away from the neighborhood of the jump, $\xi = 0$, $\vec{v}(\mathbf{E_1})$ is small as desired. The largest variance is seen in the neighborhood of the jump discontinuities, but the variance at $x = \xi$ is again small. This behavior is consistent with the oscillatory behavior observed in the jump responses for each concentration factor observed in Figure 3.2. Figure 3.8 displays the variance using $\mathbf{E_1}$ for the same concentration factors on $f_1(x)$.

The results of (3.23) using $\mathbf{E_2}$ are shown in Figure 3.9. We see that including information from (3.22) does not seem to yield any more information that will help to isolate edges, so it is not utilized further in our investigation.

[3]We note that a related idea that exploits the jump responses in (3.13) using maximum likelihood estimators was developed in Petersen *et al.* (2012). The main goal of that investigation was to produce the best ROC curve in noisy environments when given the first $2N + 1$ Fourier coefficients of a piecewise smooth function.

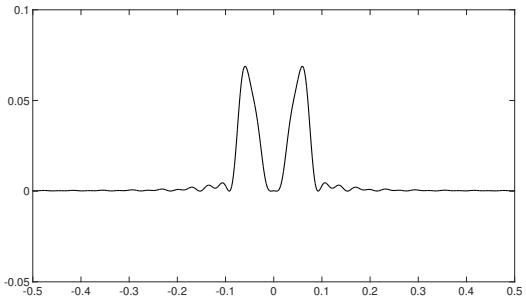
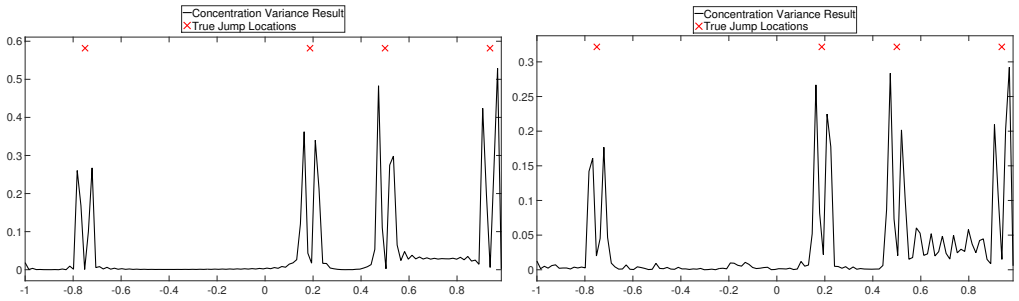
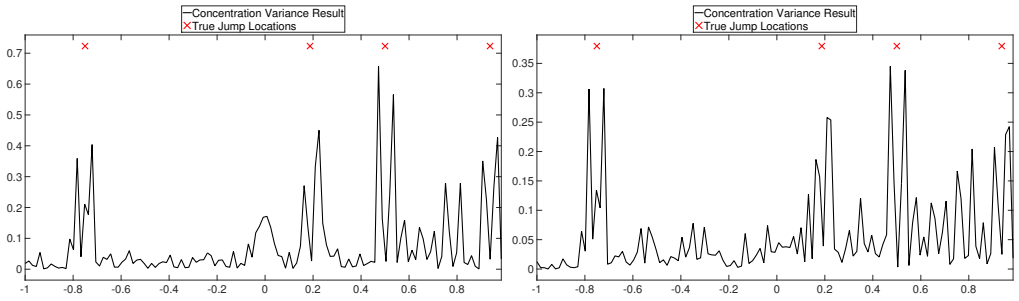


Figure 3.7: $\vec{v}(\mathbf{E_1})(x_j)$ in (3.23) for $r(x_j)$ in (3.8), $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24a) Is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$.



(a) test case 1

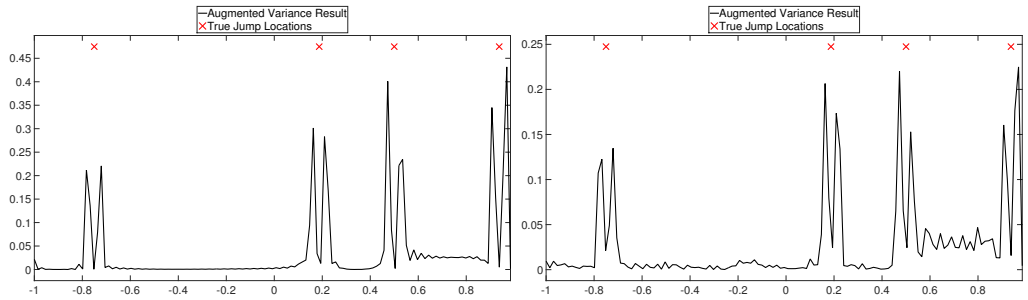
(b) test case 2



(c) test case 3

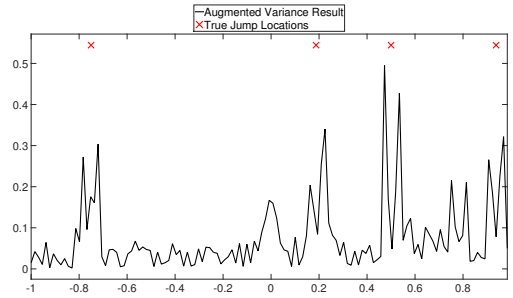
(d) test case 4

Figure 3.8: $\vec{v}(\mathbf{E_1})(x_j)$ in (3.23) for $f_1(x)$ and $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24a) Is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$.

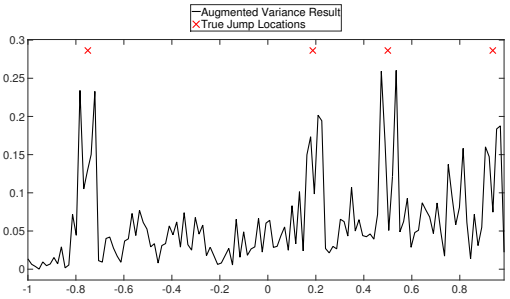


(a) test case 1

(b) test case 2



(c) test case 3



(d) test case 4

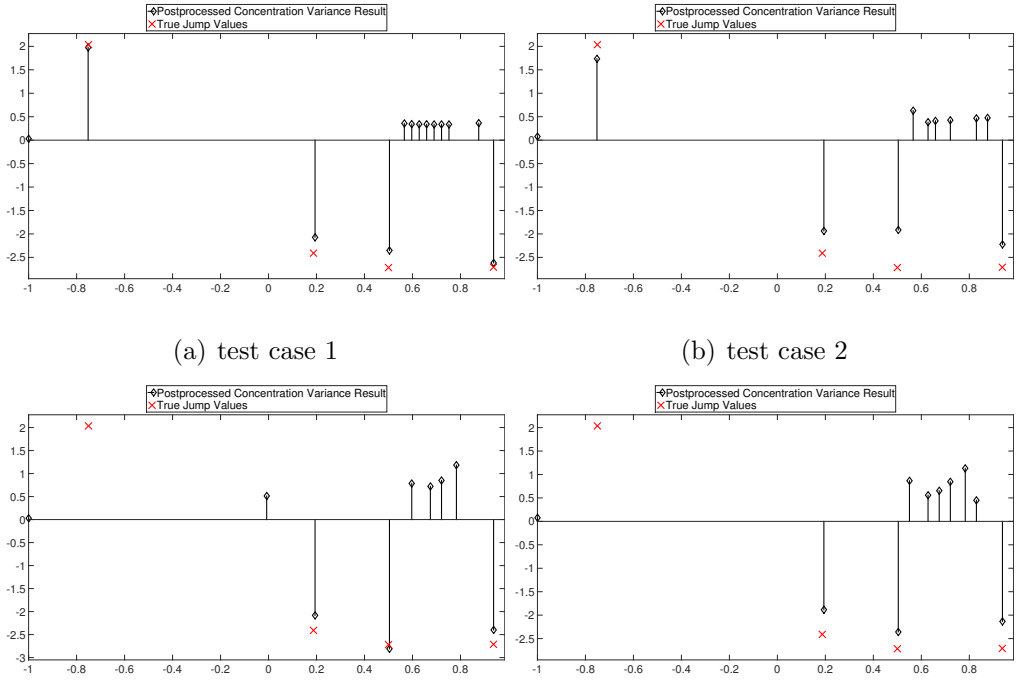
Figure 3.9: $\vec{v}(\mathbf{E_2})(x_j)$ in (3.23) for $f_1(x)$ and $x_j = \frac{j}{N}, j = -N, \cdots, N$. G in (3.24b) is Given by $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$.

3.2.5 Post Processing Edge Detection Using Variance

The variance computed in (3.23) allows us to distinguish where an edge is likely to occur from an artifact caused by the oscillatory response of the jump function approximation. Below, we provide an algorithm that allows us to convert the variance in (3.23) to an edge map.

As in Algorithm 1, we first determine which points x_j, $j = -N, \cdots, N$, describe local jump regions in the domain $[-1, 1]$. It is evident from Figures 3.8 and 3.9 that each edge occurs between two locations where $var(\mathbf{E})$ is a local maximum. Indeed, a good guess might be to choose the midpoint value between these peaks. Hence, we define the local jump regions as the distance between two neighboring peaks, which is described in Algorithm 2.

The algorithm also requires thresholding to avoid false local region detections. We threshold $v(\mathbf{E})$ in a manner similar to that applied in Algorithm 1, appealing to the presumed sparsity of the underlying jump function $[f]$, and choosing a threshold, $\epsilon(c) = \epsilon$, such that $\dim\left(\left\{v(\mathbf{E})_{x_j} < \epsilon\right\}\right) = c\,(2N+1)$. Typically $\frac{3}{4} \leq c < 1$. Algorithm 3 isolates the edges from within each jump region.



(a) test case 1

(b) test case 2

(c) test case 3

(d) test case 4

Figure 3.10: Edge Map Generated Using Algorithm 3 for $f_1(x)$. Here $\delta = 5$, $c = \frac{7}{8}$.

Figure 3.10 demonstrates application of Algorithm 3 on $f_1(x)$ using $\mathbf{E_1}$ in (3.23). While the algorithm is effective in noiseless environments with the first $2N+1$ Fourier coefficients given, it is evident that adding noise and reducing the size of the sampling set makes it difficult for the algorithm to recover true edges without producing false ones. Indeed, in test cases 3 and 4, the edge at $x = -.75$ is completely missed while false edges occur in other parts of the domain. This is not so surprising given the results in Figure 3.8, which shows large variance in smooth regions. The convergence

Algorithm 2 Local Jump Region Determination

Choose a variance threshold, ϵ as described above, and a maximum distance between peaks threshold, δ, needed to establish two neighboring peaks. Although not needed for periodic functions, we also choose d so that no jump occurs between $[x(-N), x(-N+d)]$ or $[x(N-d), x(N)]$.

set counter for the number of potential edges, $\nu = 0$.

for $j = -N+d, \cdots, N-d$

- if $\vec{v}(\mathbf{E})_{x_j} > \vec{v}(\mathbf{E})_{x_{j-1}}$ and $\vec{v}(\mathbf{E})_{x_j} > \vec{v}(\mathbf{E})_{x_{j+1}}$ and $\vec{v}(\mathbf{E})_{x_j} > \epsilon$, then
  1. $\nu = \nu + 1$
  2. $y_\nu = x_j$;
- end if

end for

set counter for the number of jump regions, $\ell = 0$.

for $n = 1, \cdots, \nu - 1$

- if $dist(y_n, y_{n+1}) < \delta$ then (we are in a jump region – otherwise, the large variance indicated a false edge))
  1. $\ell = \ell + 1$
  2. $B_\ell = \{x_j : y_n \leq x_j \leq y_{n+1}\}$
- end if

end for

Algorithm 3 Edge Map Generation

Input variables: Jump regions B_ℓ, $\ell = 1, \cdots, L$, and $S_N^\sigma[f](x)$.

form $S_N^{PP}[f](x)$ from $S_N^\sigma[f](x)$ using Algorithm 1

for $\ell = 1, \cdots, L$

1. $\xi_\ell = \arg\max_{x_j \in B_\ell} |S_N^{PP}[f](x_j)|$
2. $E(\xi_\ell) = S_N^\sigma[f](\xi_\ell)$

end for

set counter for $\ell = 1$

for $j = -N, \cdots, N$

- if $x_j = \xi_\ell$ and $\ell \leq L$ then
  1. $E(x_j) = E(\xi_\ell)$
  2. $\ell = \ell + 1$
- else $E(x_j) = 0$.

end for

---

of the concentration factor edge detection method in smooth regions given the first $2N + 1$ Fourier coefficients is such that $\vec{v}(\mathbf{E_1})$ is a valid predictor of where edges are. However, no such proof of convergence exists for the concentration factor method when the data are sub-sampled as in (3.19). Hence, we seek to develop an algorithm where the variance between the jump discontinuities, that is, in smooth regions, remains small, even when the data are sub-sampled. This is accomplished in Section 3.2.6.

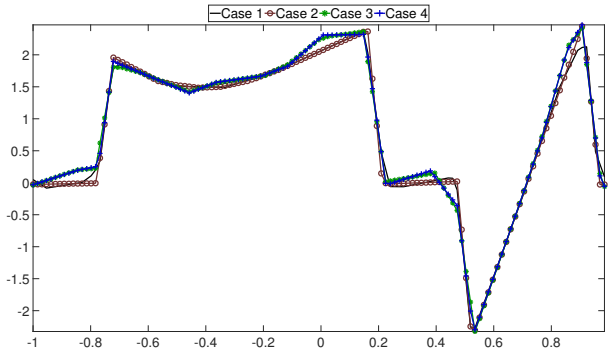
3.2.6 Determining Edges from Regularized Reconstruction

As observed in Section 3.2.5, the limitations on post processing the jump function approximation using (3.23) using either (3.24a) or (3.24b) are due to *too much* variability away from the edges in smooth regions. However, as will be shown below, using (3.23) may still be an effective tool when the input set, (**E** in (3.23)), has the property that each element in **E** differs in convergence properties only within each jump region, that is, the convergence is similar in smooth regions. Thus, we seek a new set **E** for which this property holds. As it turns out, the method developed in Archibald *et al.* (2015), which approximates piecewise smooth functions from under-sampled Fourier data using l^1 regularization, leads to such a set. The method is briefly reviewed below.

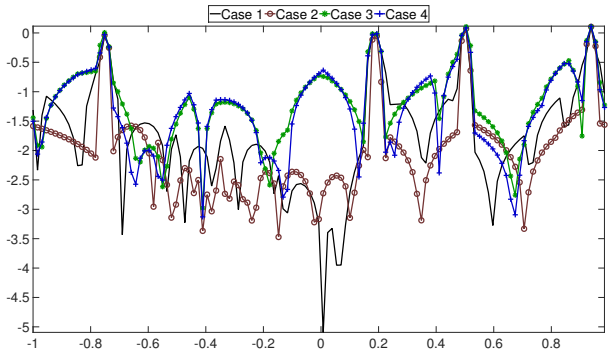
Once again we are given the (noisy) Fourier coefficients of a piecewise smooth function from the set $\hat{\mathbf{V}}$ in (3.19). For each sub-sampled set $\hat{\mathbf{T}}_q$ in (3.20), we reconstruct f on a set of uniform grid-points x_j, $j = -N, \cdots, N$, as

$$\vec{f_q} = \underset{\vec{u}}{argmin}\, ||F_q\vec{u} - \vec{\hat{f}_q}||_2^2 + \lambda||\mathbf{P}^p\vec{u}||_1. \tag{3.25}$$

Here $\vec{u}$ is a vector discretized on the physical-space grid, $\left\{x_j = \frac{j}{N}\right\}_{j=-N}^{N}$, $\vec{\hat{f}_q}$ is the vector of Fourier coefficients formed from the sub-sampled set $\hat{\mathbf{T}}_q$, and F_q is the discrete Fourier transform generating the coefficients to match $\vec{\hat{f}_q}$. The polynomial annihilation transform, $\mathbf{P}^p$ in (3.18), is used as a sparsifying operator in the penalty term. We choose $p = 2$ in our numerical experiments. We note that when $p = 1$, which is equivalent to using total variation as the l^1 term, the reconstruction does not exhibit the needed variance in the neighborhood of edges, causing edge locations to shift in (3.25). Consequently, (3.26) is not effective in this case. Conversely, choosing $p > 2$ creates more oscillations near the jump discontinuities, leading to an extended variance region with multiple peaks in each jump region, making it difficult to isolate



(a) Reconstruction



(b) log_{10} Reconstruction error

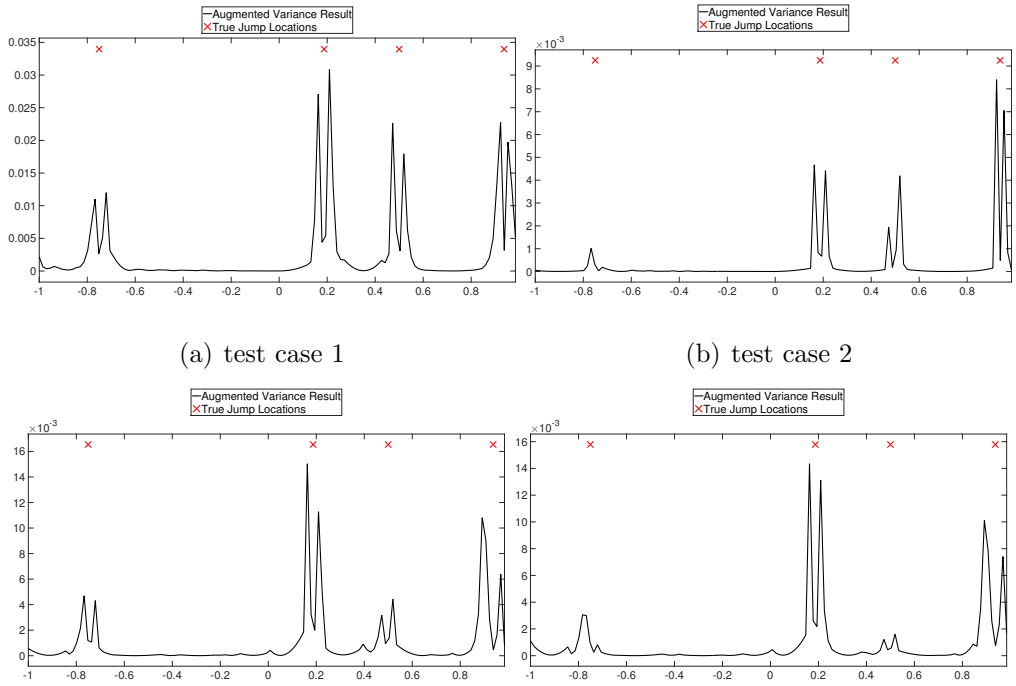
Figure 3.11: Sample Reconstructions and Pointwise Reconstruction Errors from Applying (3.25) to the Test Cases in Table 3.2.

the edges. Figure 3.11 shows the approximation of $f_1(x)$ for each of the four test cases and the corresponding pointwise errors. Observe that for the first two test cases, (3.25) converges accurately away from the discontinuities but is not accurate in the neighborhoods of the internal edges. For test cases 3 and 4, (3.25) is less accurate. Nonetheless, as we will show shortly, the solutions $\vec{f}_q$, $q = 1, \cdots, Q$, vary very little in smooth regions, while they exhibit more variation near the discontinuities. As before, we will exploit this variation in the approximation to determine the edge locations of f. However, now we choose $\mathbf{E}$ to contain the approximations of f given in (3.25). Moreover, instead of choosing different approximation parameters, e.g. σ in (3.7), we

consider different sampling sets, $\hat{\mathbf{T}}_q$, $q = 1, \cdots, Q$, in (3.20).[4] Thus for $\mathbf{Q} = \{\vec{f}_q\}_{q=1}^{Q}$, where each $\vec{f}_q$ is calculated on a set of points x_j, $j = -N, \cdots, N$, we have

$$\vec{v}(\mathbf{Q})_j = \frac{1}{Q}\sum_{q=1}^{Q}(\overrightarrow{f_q}_j - \frac{1}{Q}\sum_{q=1}^{Q}\overrightarrow{f_q}_j)^2, \quad j = 1 \ldots 2N+1. \tag{3.26}$$

Figure 3.12 show the results of (3.26) for $f_1(x)$ using each test case in Table 3.2. Once (3.26) is calculated, we can apply Algorithms 2 and 3 to recover an edge map, in this case with (3.26) replacing (3.23). Figure 3.13 shows these results.

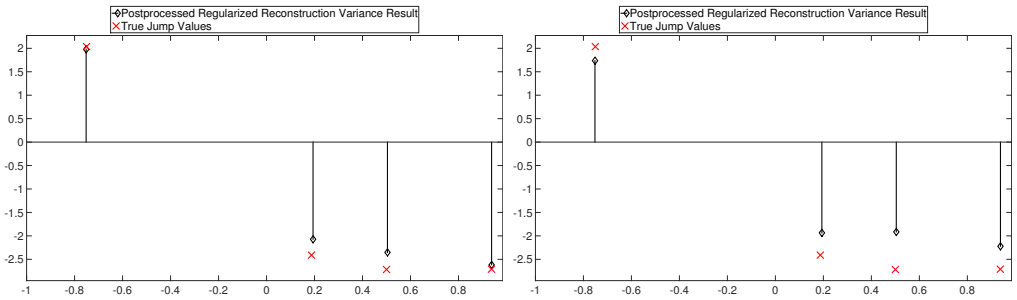


(a) test case 1 (b) test case 2

(c) test case 3 (d) test case 4

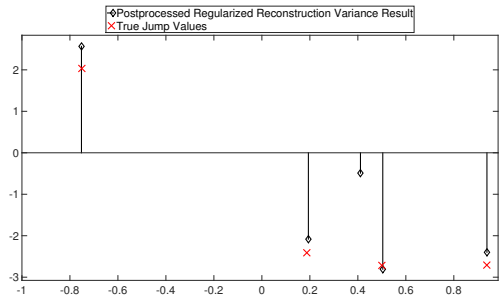
Figure 3.12: $\vec{v}(\mathbf{Q})$ for f_1 Using the Parameters in Table 3.2.

[4]We note that it is possible to vary the sampling sets for $\mathbf{E}$ in (3.23) as well. Unfortunately, for the same reasons as stated at the end of Section 3.2.5, using different sampling sets did not reduce the variance sufficiently in the smooth regions of the domain.

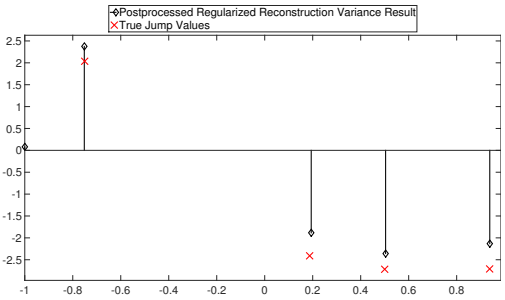


(a) test case 1

(b) test case 2



(c) test case 3



(d) test case 4

Figure 3.13: Algorithms 2 and 3 Applied to $\vec{v}(\mathbf{Q})$ for f_1 Using the Parameters in Table 3.2. Here $\delta = 5$ and $c = \frac{7}{8}$.

3.3 Determining the Two-dimensional Edge Map Using Variance

Below we describe how the methods developed in Section 3.2 can be expanded to two dimensional functions. In this case we assume that $f : \mathbb{R}^2 \to \mathbb{R}$ is a periodic piecewise smooth function on $[-1, 1]^2$, and that we are given Fourier coefficients

$$\hat{f}_{k,l} = \frac{1}{4} \int_{-1}^{1} \int_{-1}^{1} f(x, y) e^{-i\pi(kx+ly)} dy dx, \tag{3.27}$$

drawn from a set $\hat{\mathbf{U}} = \left\{ \hat{f}_{k,l} : -N \leq k, l \leq N \right\}$. For ease of presentation, we will consider the same test cases as those displayed in Table 3.2, with the equivalent parameters used in each direction.

The following function, displayed in Figure 3.14, will be used to test our algo-

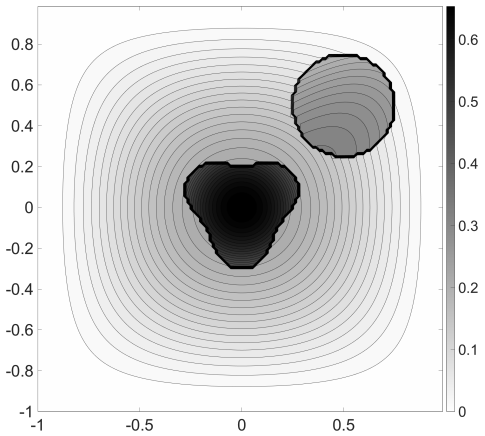


Figure 3.14: $f_2(x, y)$.

rithms:

$$f_2(x,y) = \begin{cases} \frac{1}{3}(1-x^3) + \frac{1}{2}(xy - y^2) & \text{if } \sqrt{\left(x-\frac{1}{2}\right)^2 + \left(y-\frac{1}{2}\right)^2} < \frac{1}{4} \\ \frac{2}{3} - \frac{5}{2}(x^2+y^2) & \text{if } \sqrt{x^2+y^2} < \frac{1}{4} + \frac{1}{20}\sin\left(\operatorname{atan}\left(\frac{y}{x}\right)\right) \\ \frac{1}{4}(x^4 - 2x^2 + 1)(y^4 - 2y^2 + 1) & \text{otherwise.} \end{cases} \tag{3.28}$$

We seek to recover the scalar valued jump function approximation, defined in two dimensions as

$$[f](x,y) := \sum_{j=1}^{M} [f](P_j)\chi_{P_j}(x,y), \tag{3.29}$$

where the x and y coordinates of each discontinuity P_j, $j = 1, \cdots, M$, are given by (ξ_j, η_j), and $\chi_{P_j}(x,y)$ is the two dimensional extension of (3.3), which has value 1 at each cell containing P_j and 0 everywhere else. As in the one dimensional case, we observe that there are only a sparse number of nonzero coefficients in (3.29).

We wish to recover (3.29) from (3.27) at a finite set of uniform grid points, (x_n, y_m), for $n, m = -N, \cdots, N$. Determining $[f]$ is considerably more difficult in two dimensions, because its non-zero values depend upon how each edge is approached. Since we are essentially interested in generating an edge map on a Cartesian grid, we will say that each jump value, $[f](\xi, \eta)$, is approximated as the difference of f in the x direction multiplied by the difference in f in the y direction across an internal boundary

curve at the point (ξ, η). This interpretation allows us to define the two dimensional concentration factor edge detection method as

$$S_N^\sigma[f](x,y) = -\sum_{l=-N,l\neq 0}^{N}\sum_{k=-N,k\neq 0}^{N} \hat{f}_{k,l}\text{sgn}(k)\text{sgn}(l)\sigma(\frac{|k|}{N})\sigma(\frac{|l|}{N})e^{i\pi(kx+ly)}. \tag{3.30}$$

Using (3.30) has some inherent limitations, however. In particular, if the underlying image has an edge that consists of a straight line in either coordinate direction, the method would not "see" it in that direction, as (3.30) would return zero. As discussed in Martinez *et al.* (2014), one option to improve the performance in these cases is to apply (3.30) twice, once in the Cartesian coordinate system and once again after rotating the image, which will detect edges that align with the coordinate axes. However, for generating an edge map on a Cartesian grid, we have found that applying (3.7), (3.23), and Algorithm 3 dimension by dimension, is efficient and robust. To this end, we write the dimension by dimension equivalents of (3.7) as

$$S_{x,N}^\sigma[f](x,y_m) = i\sum_{l=-N}^{N}\sum_{k=-N,k\neq 0}^{N} \hat{f}_{k,l}sgn(k)\sigma\left(\frac{|k|}{N}\right)e^{i\pi(kx+ly_m)}, \tag{3.31a}$$

$$S_{y,N}^\sigma[f](x_n,y) = i\sum_{k=-N}^{N}\sum_{l=-N,l\neq 0}^{N} \hat{f}_{k,l}sgn(l)\sigma\left(\frac{|l|}{N}\right)e^{i\pi(kx_n+ly)}, \tag{3.31b}$$

for each $x_n = \frac{n}{N}$, and $y_m = \frac{m}{N}$, $-N \leq n, m \leq N$. We compute (3.31a) and (3.31b) on x_n and y_m respectively. For algorithmic purposes, we will use $S_{x,N}^\sigma[\mathbf{F}]_{m,n}$ to represent (3.31a) applied row by row or in the $\vec{x}$ direction and correspondingly $S_{y,N}^\sigma[\mathbf{F}]_{m,n}$ to represent (3.31b) applied column by column or in the $\vec{y}$ direction. We can then combine these results to form

$$S_{C,N}^\sigma[\mathbf{F}]_{m,n} = \sqrt{\left(S_{x,N}^\sigma[\mathbf{F}]_{m,n}\right)^2 + \left(S_{y,N}^\sigma[\mathbf{F}]_{m,n}\right)^2}.$$

As in Algorithm 1, an edge map, $S_{C,N}^{PP}[\mathbf{F}]$, can be obtained by thresholding $S_{C,N}^\sigma[\mathbf{F}]$ as demonstrated below.

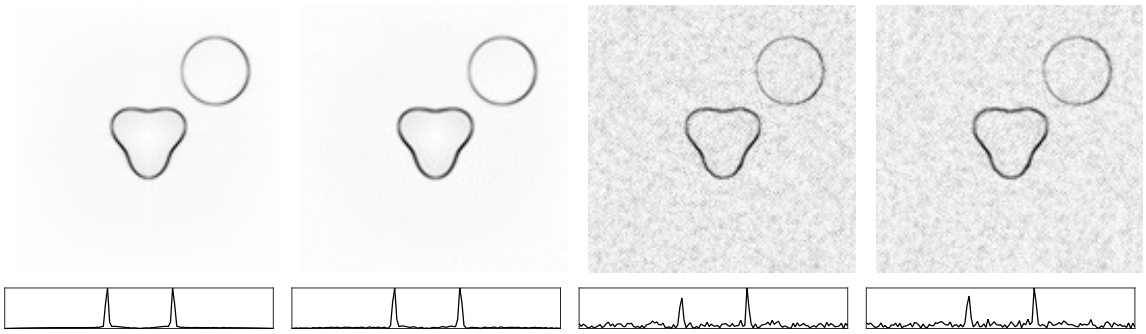
Algorithm 4 Two Dimensional Edge Map Generation Using Thresholding

Input variables: $S^{\sigma}_{C,N}[\mathbf{F}]_{m,n}$.

for $j = -N, \cdots, N$

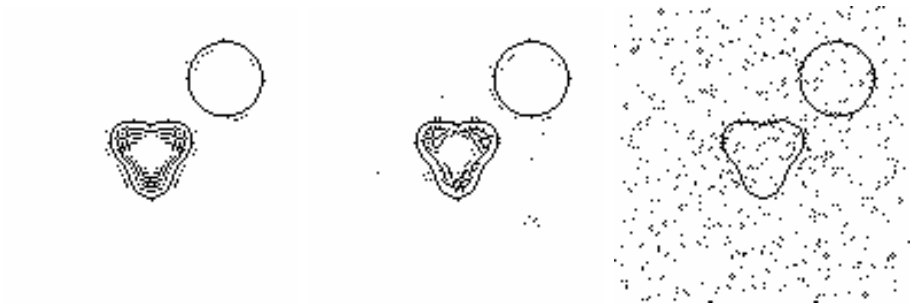
1. let $\vec{u}$ be the thresholded jump map generated by Algorithm 1 using $S^{\sigma}_{C,N}[\mathbf{F}]_{j,1\ldots 2N+1}$
2. for $k = -N, \cdots, N$
   (a) let $\vec{v}$ be the thresholded jump map generated by Algorithm 1 using $S^{\sigma}_{C,N}[\mathbf{F}]_{1\ldots 2N+1,k}$
   (b) if $|\vec{v}_j| > |\vec{u}_k|$
      i. $S^{PP}_{C,N}[\mathbf{F}]_{j,k} = \vec{v}_j$
   (c) else
      i. $S^{PP}_{C,N}[\mathbf{F}]_{j,k} = \vec{u}_k$
3. end for

end for

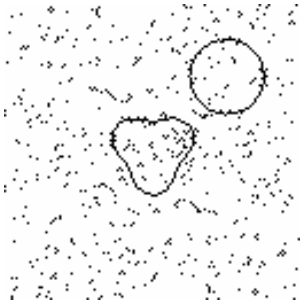


(a) test case 1 (b) test case 2 (c) test case 3 (d) test case 4

Figure 3.15: $S^{\sigma_G}_{C,N}[\mathbf{F_2}]_{m,n}$ and a Cross-section at $S^{\sigma_G}_{C,N}[\mathbf{F_2}]_{m,N+1}$.



(a) test case 1 (b) test case 2 (c) test case 3



(d) test case 4

Figure 3.16: The Results of Applying Algorithm 4 to $S^{\sigma_G}_{C,N}\left[\mathbf{F_2}\right]_{m,n}$ with Thresholding at $c = 15/16$.

Figure 3.15 and 3.16 show $S^{\sigma}_{C,N}\left[\mathbf{F_2}\right]_{m,n}$ and the thresholded result respectively. Thresholding is difficult even for test case 1, and becomes less effective with fewer available coefficients or more noise, as oscillations appear both at nearby edge points as well as in smooth regions.

As in the one dimensional case, the primary advantage of the minmod algorithm, (3.21), is that it reduces oscillations that result from the so called "side lobes" of any particular concentration factor. Indeed, the internal oscillations seen in test case 1 of Figure 3.16 are greatly reduced. However, this advantage becomes negligible as the data are increasingly noisy or under-sampled. The sparsity enforcing edge detection method, (3.22), is also readily extended to multiple dimensions using a dimension by dimension approach, and the results are analogous to the one dimensional case. Since neither offers a significant improvement over the standard concentration factor edge

detection method, the results are not included here.

3.3.1 Variance of Jump Function Responses

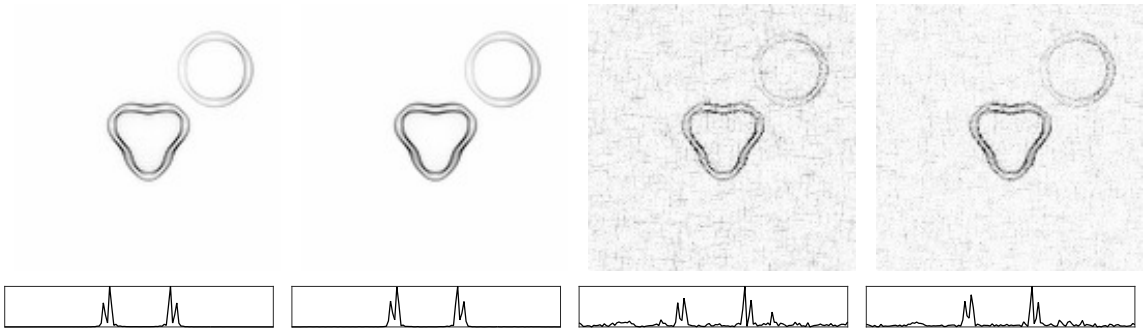
Analogously to the one dimensional case in (3.23), we define the two dimensional variance on the set of two dimensional jump function approximations, $\mathbf{E} = \{E_1\mathbf{F}, \cdots, E_n\mathbf{F}\}$, where each $E_j\mathbf{F}$ is calculated on a set of points (x_n, y_m), with $-N \leq m, n \leq N$, as

$$\mathbf{V}(\mathbf{E})_{m,n} = \frac{1}{dim(\mathbf{E})} \sum_{E_j \in \mathbf{E}} (E_j\mathbf{F}_{m,n} - \frac{1}{dim(\mathbf{E})} \sum_{E_k \in \mathbf{E}} E_k\mathbf{F}_{m,n})^2. \tag{3.32}$$

Here

$$\mathbf{E} = \{S^{\sigma_k}_{C,N}[\mathbf{F}]_{m,n}\}_{\sigma_k \in G}.$$

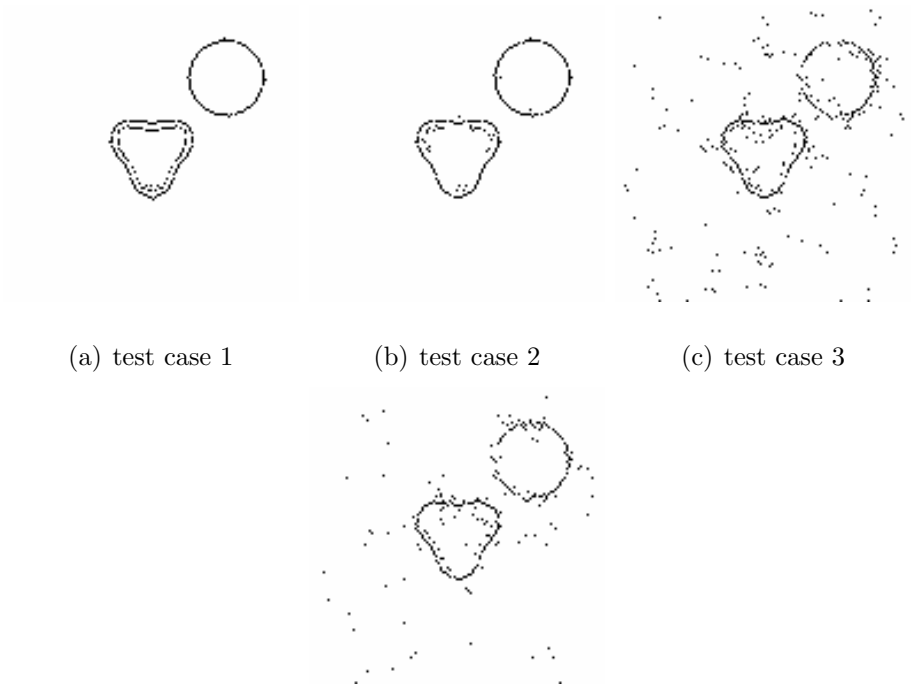
Algorithm 5 describes how to generate a two dimensional edge map from (3.32). As with Algorithm 2, there is a need for thresholding. We now choose $\epsilon = \epsilon(c)$ such that $\dim\left(\left\{\mathbf{V}(\mathbf{E})_{m,n} < \epsilon\right\}\right) = c\,(2N+1)^2$. We note that Algorithm 5 actually returns an indicator of relative edge heights rather than a pair of directional jump values. These can be determined if necessary by referencing $S^{\sigma}_{N,X}[\mathbf{F}]_{m,n}$ and $S^{\sigma}_{N,Y}[\mathbf{F}]_{m,n}$ at edge points.



(a) test case 1 (b) test case 2 (c) test case 3 (d) test case 4

Figure 3.17: $\mathbf{V}(\mathbf{E})_{m,n}$ Using $G = \{\sigma_G, \sigma^1_P, \sigma^2_E, \sigma^4_E\}$ and a Cross-section $\mathbf{V}(\mathbf{E})_{m,N+1}$.

The variance of jump response, $\mathbf{V}(\mathbf{E})_{m,n}$, using the concentration factor set $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$, is shown in Figure 3.17 with the results from Algorithm 5 shown in Figure 3.18. The high variance regions in the neighborhood of edges are well defined with the full set of noise free Fourier coefficients as well as when those coefficients are sub-sampled. However, it is evident that large variance levels appear in smooth regions in the presence of noise. Thresholding $\mathbf{V}(\mathbf{E})_{m,n}$ in Algorithm 2 causes some false jump regions to be identified and some true jump regions to be discarded, leading to false positives and missed edges in Algorithm 5 (Figure 3.18(c) and (d)).



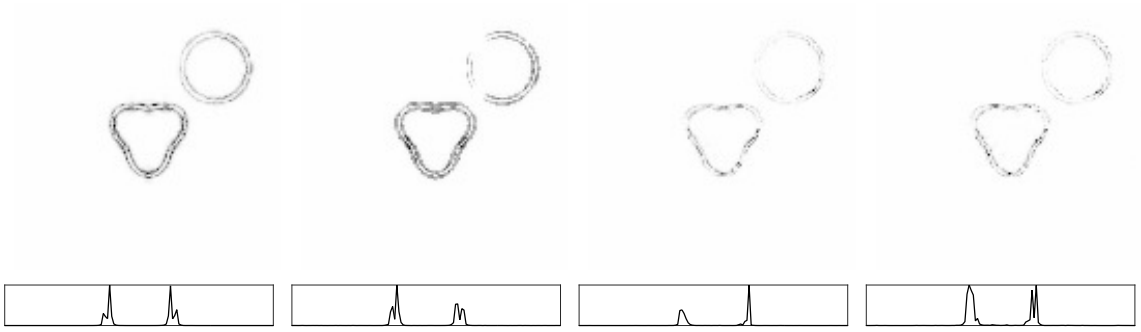
(a) test case 1 (b) test case 2 (c) test case 3

(d) test case 4

Figure 3.18: The Results of Applying Algorithm 5 to $\mathbf{V}(\mathbf{E})_{m,n}$ Using $G = \{\sigma_G, \sigma_P^1, \sigma_E^2, \sigma_E^4\}$, Maximum Distance between Peaks, $\delta = 7$, Threshold $c = 15/16$.

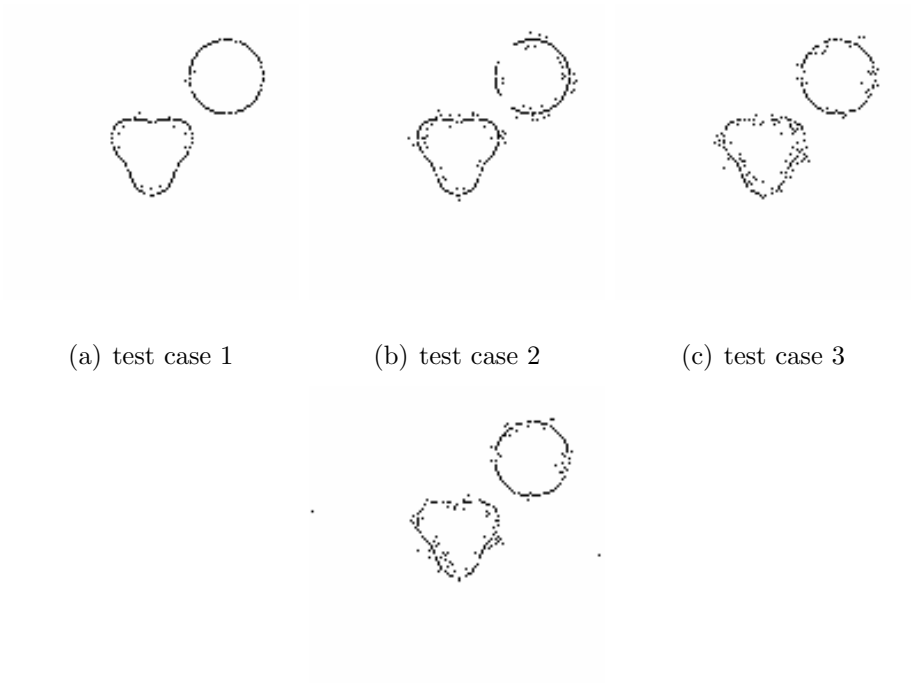
3.3.2 Determining Edges by Regularized Reconstruction

Algorithm 5 works well when the behavior of the variance can properly identify local jump regions. As in the one dimensional case, the variance of the regularized



(a) test case 1 (b) test case 2 (c) test case 3 (d) test case 4

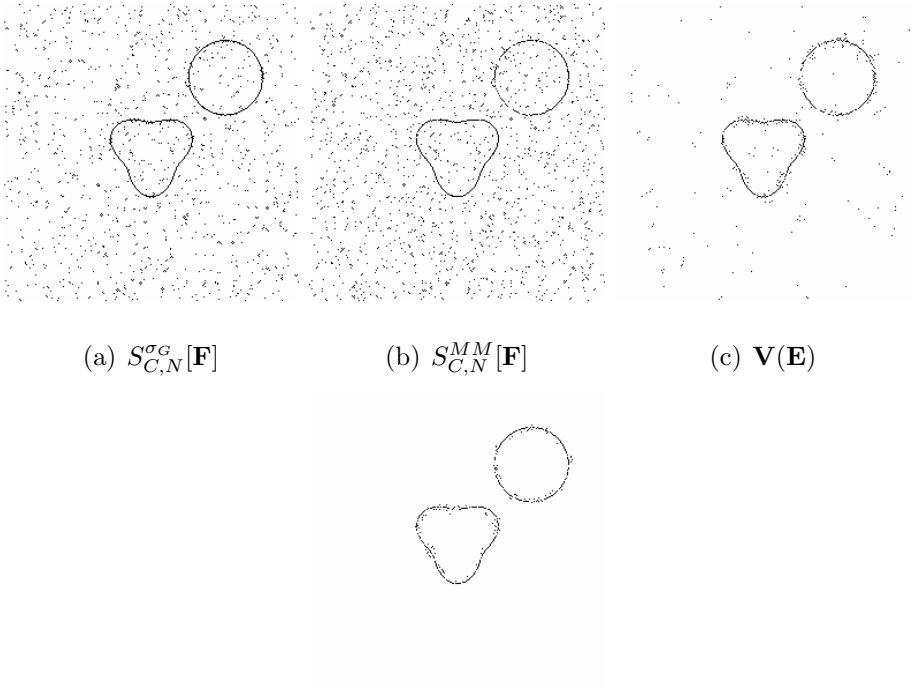
Figure 3.19: $\mathbf{V}(\mathbf{Q})_{m,n}$ Using $Q = 40$ and a Cross-section $\mathbf{V}(\mathbf{Q})_{m,N+1}$.



(a) test case 1 (b) test case 2 (c) test case 3

(d) test case 4

Figure 3.20: The Results of Applying Algorithm 5 to $\mathbf{V}(\mathbf{Q})_{m,n}$ with $Q = 30$, Maximum Distance between Peaks, $\delta = 7$, threshold $c = 15/16$.



(a) $S^{\sigma_G}_{C,N}[\mathbf{F}]$ (b) $S^{MM}_{C,N}[\mathbf{F}]$ (c) $\mathbf{V}(\mathbf{E})$

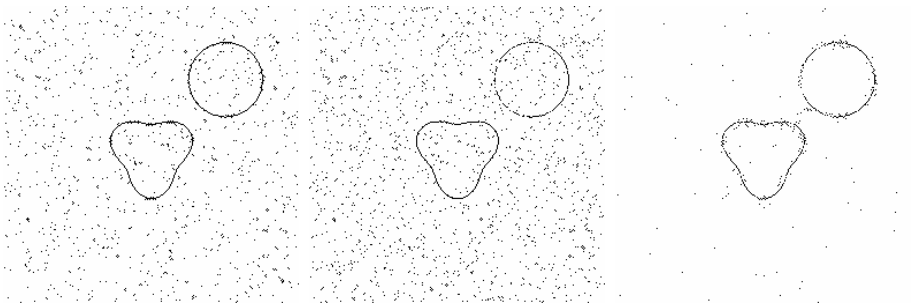
(d) $\mathbf{V}(\mathbf{Q})$

Figure 3.21: Comparison of Algorithm 5 Results Using N=128, SNR=12.5dB, $\gamma = .75$, $\beta = 0.3$, $\delta = 7$, Threshold $c = 31/32$.

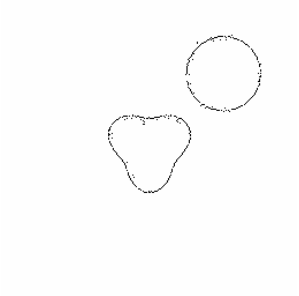
reconstruction method, (3.25), augments the edge detection criterion further by recognizing the piecewise smooth structure of the underlying function. In two dimensions, the l^1 regularized reconstruction for each sampling set $\hat{\mathbf{T}}_q$, $q = 1, \cdots, Q$ is achieved by solving the regularization in Wasserman *et al.* (2015); Archibald *et al.* (2015),

$$\mathbf{F}_{(m,n),q} = \underset{\mathbf{G}}{argmin}\, ||F_q\mathbf{G} - \hat{\mathbf{f}}_q||_F^2 + \lambda_x|| \, vec\left(\mathbf{P}_x^p\mathbf{G}\right)||_1 + \lambda_y|| \, vec\left(\mathbf{P}_x^p\mathbf{G}^T\right)||_1, \quad (3.33)$$

where $\mathbf{P}_x^p$ corresponds to using (3.18) in the x direction for each fixed y_m, $-N \leq m \leq N$, and similarly applying $\mathbf{P}_x^p$ to $\mathbf{G}^T$ performs the calculation in the y direction. Here $\mathbf{G}$ represents the physical-space image discretized on the uniform grid in the domain $[-1, 1) \times [-1, 1)$. We note that in Archibald *et al.* (2015) the split Bregman algorithm, Goldstein and Osher (2009), was shown to be applicable for the polynomial annihilation transform operator, making (3.33) computationally efficient. Analogous



(a) $S^{\sigma_G}_{C,N}[\mathbf{F}]$ (b) $S^{MM}_{C,N}[\mathbf{F}]$ (c) $\mathbf{V}(\mathbf{E})$



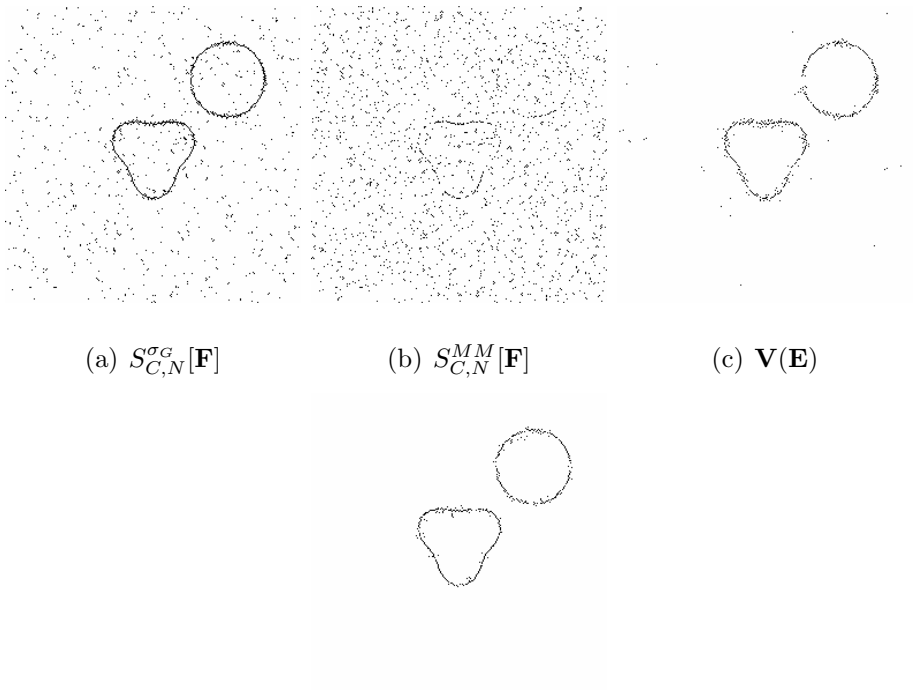
(d) $\mathbf{V}(\mathbf{Q})$

Figure 3.22: Comparison of Algorithm 5 Results Using N=128, SNR=12.5dB, $\gamma = \frac{9}{16}$, $\beta = 0.09$, $\delta = 7$, Threshold $c = 31/32$.

to (3.26), the variance of the two dimensional regularized reconstructions is given by

$$\mathbf{V}(\mathbf{Q})_{(m,n)} = \frac{1}{Q}\sum_{q=1}^{Q}(\mathbf{F}_{(m,n),q} - \frac{1}{Q}\sum_{q=1}^{Q}\mathbf{F}_{(m,n),q})^2. \tag{3.34}$$

Figure 3.19 illustrates the results of (3.34), while the results after post processing with Algorithm 5 are shown in Figure 3.20. The identification of false edges in smooth but variable regions results from a lack of resolution in the given data. Thresholding reduces the false detects but also discards some true edges. Numerical experiments indicate that increasing the size of the initial data set U defined in Section 3.1.4 for the polynomial annihilation transform of order 2 improves the performance of the algorithm. Figure 3.21 displays the results of each algorithm for $N = 128$ with the remaining parameters in Table 3.2 held constant. The threshold constant for Algorithm 1, c, is adjusted to reflect the increase in resolution. Figure 3.22 shows



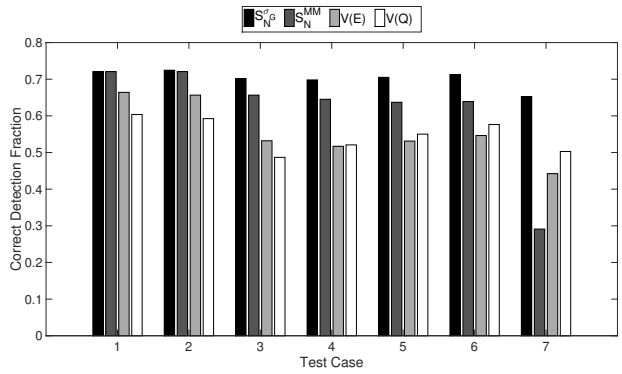
(a) $S^{\sigma_G}_{C,N}[\mathbf{F}]$ (b) $S^{MM}_{C,N}[\mathbf{F}]$ (c) $\mathbf{V}(\mathbf{E})$

(d) $\mathbf{V}(\mathbf{Q})$

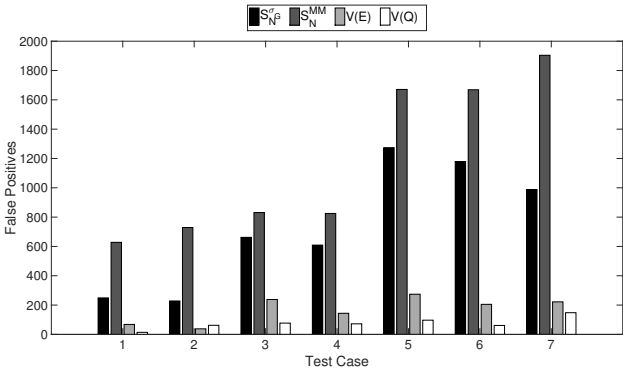
Figure 3.23: Comparison of Algorithm 5 Results Using N=128, SNR=8dB, $\gamma = \frac{1}{5}$, $\beta = 0.09$ $\delta = 7$, Threshold $c = 31/32$.

how each algorithm is affected when the initial data set U is increased to $N = 128$, but the sample size for each test, $\gamma(2N+1)^2$ is held fixed (that is, to the case when $N = 64$). Finally, Figure 3.23 demonstrates that for $N = 128$, similar results can be obtained with noisier data ($SNR = 8$) and more sub-sampling ($\gamma = .2$).

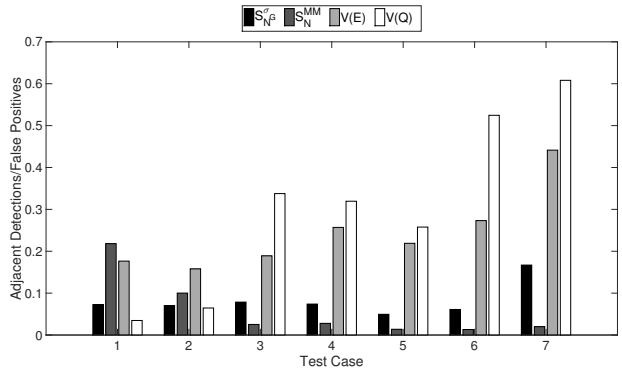
Figure 3.24 compares our methods quantitatively for each of the four test cases, as well as the examples from Figures 3.21, 3.22, and 3.23, by measuring their ability to properly classify points as edges. The figure displays the fraction of edge cells that were correctly detected, the count of false positives, and the fraction of false positives that lie in the 8-connected neighborhood of true edge cells. Observe that when the variance for the regularized reconstruction is used, most of the false positives are near the true edges.



(a) Correct detections/edge cells



(b) False positives



(c) False positives adjacent to edge cells/false positives

Figure 3.24: Comparison of Methods Using Algorithm 4 on $S_{C,N}^{\sigma_G}$ [$\mathbf{F_2}$] and $S_{C,N}^{MM}$ [$\mathbf{F_2}$] and Algorithm 5 on $\mathbf{V}(\mathbf{E})$ and $\mathbf{V}(\mathbf{Q})$ for the Test Cases in Table 3.2 and Figures 3.21, 3.22, and 3.23.

3.4 Concluding Remarks

Assuming sufficient resolution, the concentration factor edge detection method, (3.7), effectively recovers the edges of a piecewise smooth function from its first $2N+1$ noiseless Fourier coefficients. The method is subject to many false positives if the data are noisy or if values from the set of $2N+1$ Fourier coefficients are missing or otherwise unusable. Thresholding helps, but the threshold value is problem dependent and is therefore not very robust to added noise or reduced sample size. Moreover, when jumps are small, thresholding may eliminate true edges. Processing the results via the minmod algorithm, (3.21), helps to eliminate some false edges that are artifacts of oscillatory jump function responses, but is still not robust when data are noisy or under-sampled. Finally, while the sparsity promoting l^1 regularization edge detection algorithm, (3.22), is in general more robust to noise and under-sampling, it still does not fare well under increasing noise and/or under-sampling.

This investigation has demonstrated that post-processing the concentration factor edge detection method by evaluating the *variance* of each jump function reconstruction, (3.23), is more effective at eliminating false positives generated from noisy data. It is furthermore fast and easy to implement, as it requires no iterative solutions, and each test can be performed in parallel. However, it also loses some effectiveness as the Fourier data become increasingly noisy, or when data are more sparsely sampled in the high frequency range. This lack of robustness can be explained by the fact that as the level of noise or the amount of sub-sampling increase, the concentration factor edge detection method using any concentration factor will recover false jumps in smooth regions, and thus result in low variance there. Hence this investigation sought an algorithm that is able to maintain its high order convergence properties in smooth regions while effectively isolating jump discontinuities, even in the pres-

ence of noise or under-sampling. We achieve this by adding the prior information that the true function is piecewise smooth. To this end, we observe that the l^1 regularized reconstruction algorithm, (3.25), which reconstructs images from sparsely sampled noisy Fourier data, does indeed maintain the convergence rate of the sparsifying transform operator in smooth regions so long as the l^1 regularization term is at least second order (implying that total variation is not an effective choice). Thus, we see that applying the variance technique in (3.26) to the l^1 regularized reconstruction algorithm is effective in detecting the edges of an image, which can in turn be used to classify regions of interest. Our method is robust to increasing levels of noise and sub-sampling, and is efficient since each test can be performed in parallel. Future investigations will include a more rigorous study of parameter choices. Future investigations will also include the case where the Fourier data are sampled non-uniformly, which occurs in applications such as propeller and parallel MRI. We believe that in these cases our algorithm will further demonstrate its computational efficiency, since in the non-uniform case, the FFT is not as readily used, so reducing the amount of data needed becomes more critical.

Algorithm 5 Two Dimensional Edge Map Generation

Input variables: $S^{\sigma}_{C,N}[\mathbf{F}]_{m,n}$, $\mathbf{V}(\mathbf{E})_{m,n}$.

for $j = -N, \cdots, N$

1. $\vec{u} = \mathbf{V}(\mathbf{E})_{j,1\ldots 2N+1}$
2. $\vec{w} = S^{\sigma}_{C,N}[\mathbf{F}]_{j,1\ldots 2N+1}$
3. determine B_ℓ from $\vec{u}$ by Algorithm 2
4. determine $(\mathbf{E_X})_{j,1\ldots 2N+1}$ from B_ℓ and $\vec{w}$ by Algorithm 3

end for

for $j = -N, \cdots, N$

1. $\vec{u} = \mathbf{V}(\mathbf{E})_{1\ldots 2N+1}, j$
2. $\vec{w} = S^{\sigma}_{C,N}[\mathbf{F}]_{1\ldots 2N+1}, j$
3. determine B_ℓ from $\vec{u}$ by Algorithm 2
4. determine $(\mathbf{E_Y})_{1\ldots 2N+1,j}$ from B_ℓ and $\vec{w}$ by Algorithm 3

end for

for $j = -N, \cdots, N$

1. for $k = -N, \cdots, N$
   (a) $\mathbf{E}_{j,k} = \left|(\mathbf{E_X})_{j,k}\right| + \left|(\mathbf{E_Y})_{j,k}\right|$
2. end for

end for

Chapter 4

AN ADAPTIVE FOURIER FILTER FOR RELAXING TIME STEPPING CONSTRAINTS FOR EXPLICIT SOLVERS

Filters are often used to stabilize piecewise smooth solutions. In order to maintain spectral accuracy away from discontinuities, such filters must decay with high order smoothness, Hesthaven *et al.* (2007); Tadmor (2007). Unfortunately, high order filters require small time steps to maintain stability in partially filtered modes; and achieving high order smoothness results in diffusion in some innately stable modes. Apart from using filters to improve accuracy of under-resolved solutions, the resolution of the solution space can be increased; but this comes at the cost of even smaller step sizes and greater computational effort per step.

As a way of better balancing accuracy and computational cost, we introduce an adaptive filter, which maintains stability without diffusion when the numerical solution is well resolved, but acts as a high order filter when spectral support is large. The modification to a standard filter is simple to implement and has negligible computational cost. The numerical tests show this filter can achieve a lasting increase in solution accuracy even after the time when solutions become permanently under-resolved and traditional filtering is required.

This chapter is organized as follows: In section 4.1 the necessary background in filter construction and the sources of instability are reviewed. In section 4.2 a simple chop filter is introduced, which is further refined into an adaptive filter. In section 4.3 we present the results of a variety of numerical tests using this filter.

4.1 Background

It is well known that the pseudo-spectrum of the spatial discretization must sit within the stability region of the time integration scheme, Reddy and Trefethen (1992). Violating this condition leads to exponential growth of modes with eigenvalues outside the region. Even when solution spectral support is limited to stable modes, numerical noise can perturb modes with growth factors larger than one, which then grow exponentially, LeVeque (2002).

A direct solution to this problem is to reduce time step size, which also increases accuracy, but may be prohibitively expensive computationally. For piecewise smooth solutions, filtering promotes stability and can control modes with large growth factors, Gelb and Tadmor (2000). The requirements for high quality filters that promote spectral accuracy has been well studied, Hesthaven *et al.* (2007).

Definition 2. *Let the ratio of a given Fourier frequency to the highest allowable frequency in the solution space be given by:* $\eta = \frac{|j|}{N}, j = -N \dots N$. *An even function,* $\sigma(\eta) \geq 0$ *is a filter of order* q *provided that:*

- $\sigma(\eta) \in C^{q-1}[-\infty, \infty]$
- $\sigma(0) = 1$ *and* $\sigma(\eta) = 0, \eta \geq 1$
- $\sigma^{(m)}(0) = \sigma^{(m)}(1) = 0, \forall m \in [1, ..., q-1]$

A commonly used filter is the exponential filter, given by:

$$\sigma(\eta) = \begin{cases} 1 & \eta \leq \eta_c \\ e^{-\alpha\left(\frac{\eta-\eta_c}{1-\eta_c}\right)^p} & \eta > \eta_c \end{cases} \tag{4.1}$$

Typically α *is chosen such that* $\sigma(1) = \mathcal{O}(\varepsilon_{machine})$. *The filter acts on modes starting*

at η_c *and when* $\eta_c = 0$*, we have:*

$$\sigma(\eta) = e^{-\alpha\eta^p}. \tag{4.2}$$

Reconstruction quality in smooth regions can be improved by increasing the power of the exponential filter, p. Stability is better for smaller p, but filter induced diffusion extends into low frequency modes with smaller values of p and η_c, as illustrated in Figure 4.1. Note that by Definition 2 increasing p or η_c requires more Fourier modes, Tadmor (2007). Thus we see that balancing spectral accuracy with performance leads to filters that introduce some level of diffusion in otherwise stable modes.

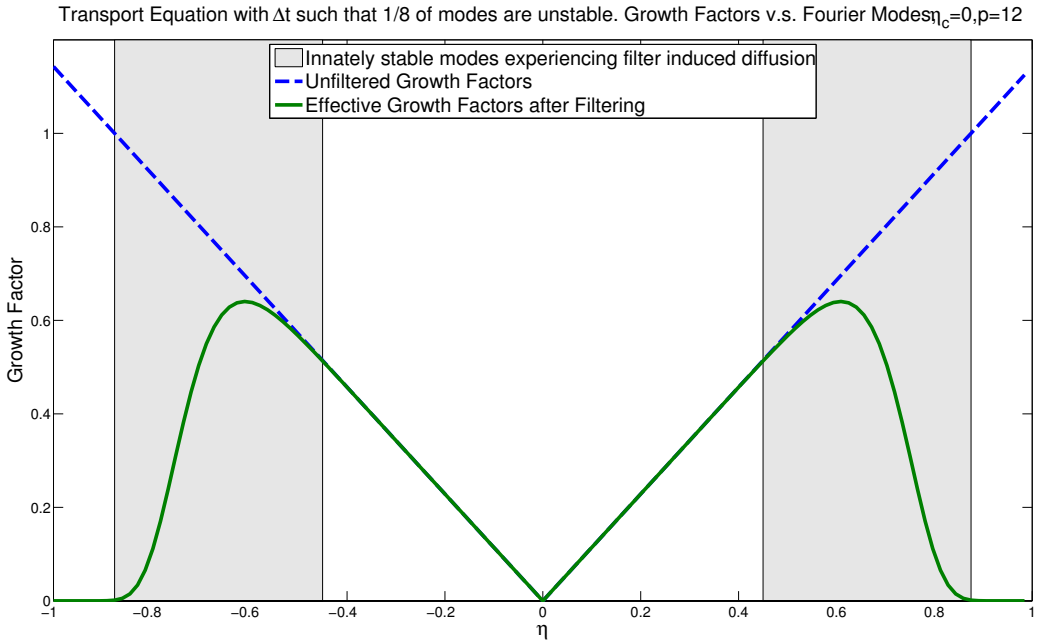


Figure 4.1: The Effect of Filtering on Growth Factors.

4.2 The Proposed Filter

When a solution has small spectral support it is possible that the exact solution is zero in the unstable modes. Until the solution expands into unstable regions the only trigger for instabilities is numerical noise. In particular, one often uses the FFT in

solving nonlinear problems and round-off errors in the FFT are sufficient to perturb unstable modes that then grow exponentially, Schatzman (1996).

Consider the following filter:

$$chop\left(\hat{u}_j\right) = \begin{cases} \hat{u}_j & |\hat{u}_j| > \lambda \\ 0 & |\hat{u}_j| \leq \lambda \end{cases}, \quad \lambda \approx 250\varepsilon_{Machine} \tag{4.3}$$

The motivation for (4.3) comes from the shrink operator used in l_1 regularization problems and the proposal for solving PDEs by maintaining solution sparsity using this operator, Schaeffer *et al.* (2013); Osher and Li (2009). For well scaled and well posed problems, $|\hat{u}_j|$ in modes associated with the actual solution will exceed the threshold and will be unaffected leaving only noise driven modes to be corrected to zero. Non-linear PDE terms can introduce data in modes that are indistinguishable from noise, but in practice for well scaled problems these effects are fleeting and exist at a level many orders of magnitude smaller than the accuracy of the numerical scheme.

The chop filter thus allows time steps that may exceed the CFL condition without introducing diffusion; and the solution remains stable as long as its instantaneous support sits in the stability region of the numerical scheme. Nevertheless, problems of interest will have support that spends some time in unstable regions. During these periods (4.3) is not contractive and is completely ineffective at controlling instabilities.

In order to stabilize the solution once the true spectrum expands into the region of instability a standard filter can be used. A new threshold parameter, τ, can be compared against the size of the spectral support of the solution to determine whether the filter should merely chop noise, or chop noise as well as apply an exponential filter. The resulting hybrid filter maintains stability while minimizing diffusive effects when spectral support is small.

Algorithm 6 can be optimized substantially. Determination of the size of the spectrum

Algorithm 6 Adaptive Filter

1: **procedure** ADAPTIVE_FILTER($\hat{u}$,λ,threshold)
2: **for all** $\hat{u}_j$ **do**
3: **if** $|\hat{u}_j| < \lambda$ **then**
4: $\hat{u}_j \leftarrow 0$
5: $support \leftarrow \max(\{|j| : |\hat{u}_j| > \lambda\})$
6: **if** $support \geq threshold$ **then**
7: **for all** $\hat{u}_j$ **do**
8: $\hat{u}_j \leftarrow \hat{u}_j e^{-\alpha\eta^p}$ ▷ α, η, p as defined in (4.2)

on line 5 can be found as a side effect of the chop filter on lines 2-4. Other than making this determination, the chop filter affects each mode independently and can be made parallel. In addition, the chop operator, (4.3), requires the determination of the magnitude of a complex number. Numerical tests show that (4.4) is a less expensive alternative to (4.3) and has no measurable impact on the calculated solution.

$$\text{chop}^*(\hat{u}_j) = \begin{cases} \hat{u}_j & |\text{Re}(\hat{u}_j)| + |\text{Im}(\hat{u}_j)| > \lambda^* \\ 0 & otherwise \end{cases} \tag{4.4}$$

4.3 Numerical Results

In the results that follow we will make use of the following definitions:

Definition 3. *Time step acceleration factor, ω: The ratio between the smallest stable time step size for an unfiltered solution and the smallest stable time step size for the solution using the adaptive filter.*

Definition 4. *Enduring relative accuracy, Ψ: The ratio of the accuracy in the solution using only* (4.2) *vs. algorithm 6 at some time after the solution has become under-resolved and the accuracy of each method is well established.*

To demonstrate our new algorithm, we consider the following example:

Example 1.

$$u_t + c(x)\, u_x = 0 \tag{4.5a}$$

$$c(x) = \frac{1}{2}\sin^2(\beta x) + \frac{1}{\gamma}; \quad u(x,0) = \cos(x) \tag{4.5b}$$

We used a fourth order Runge-Kutta scheme for time stepping and a spatial discretization with N Fourier modes.

The form of (4.5) was chosen to help illustrate the effects of the adaptive filter, Algorithm 6. With properly chosen constants and initial conditions, the solution starts out stable and well resolved, and then the spectrum grows into the region of instability and beyond to the region where aliasing can occur.

Table 4.1: Parameter Values and Results at High Resolution.

Equation Parameters			Filter Parameters			Results	
β	γ	N modes	p power	λ, chop threshold	τ, adaptive threshold	ω accel.	Ψ accuracy
2	20	1000	12	5×10^{-13}	0.96	1.06	10.54

Figure 4.2 shows a comparison of the accuracy using the adaptive filter, Algorithm 6, vs. the exponential filter, (4.2), alone using the equation and filter parameters in Table 4.1. In the region $0 \leq t < 2.26$ the solution is stable and the spectrum sits in a region where the value of the exponential filter is essentially one. The two filters perform identically. In the region $2.26 \leq t < 2.98$, the spectrum has grown to the point where the solution is still stable and the adaptive filter is only chopping, but the exponential filter has become diffusive and the accuracy of the solution suffers. In the region $2.98 \leq t < 4.76$, the adaptive threshold is periodically exceeded and the adaptive filter must apply diffusion during some time steps. In this region, the

exponential filter is continually diffusive and accuracy decreases more rapidly than with the adaptive filter. Finally in the region $4.76 \leq t$ the spectrum sits in the region of instability and aliasing. The adaptive filter frequently acts like the exponential filter, but the early accuracy gains persist.

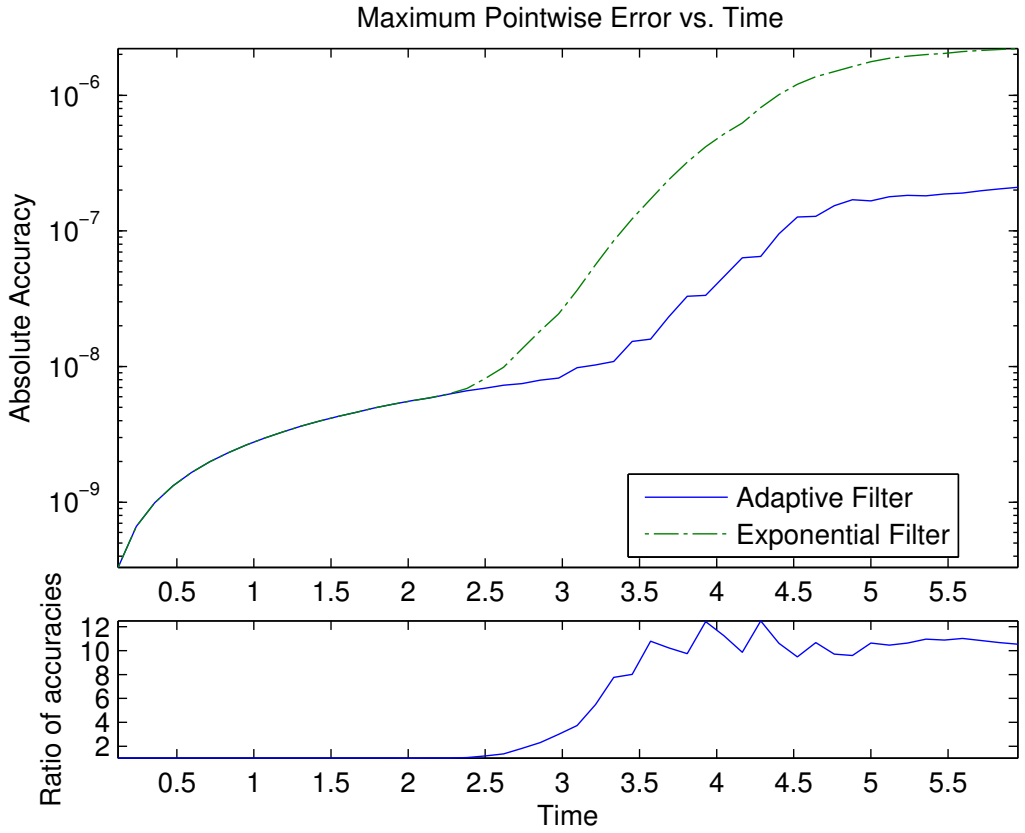


Figure 4.2: The Effect of the Adaptive Filter on Accuracy Using the Parameters in Table 4.1.

Figure 4.3 shows the source of this enduring accuracy gain for the results Figure 4.2. With the strong diffusion of the exponential filter, the spectral support never grows beyond 75% of the available modes in the numerical solution space. When the adaptive filter operates as a chop filter, the spectral support grows until it reaches the adaptive threshold value, $\tau = .96$. At this point the adaptive filter acts as the exponential filter and further support growth is limited, i.e., the effective resolution of the method is increased vs. the exponential filter.

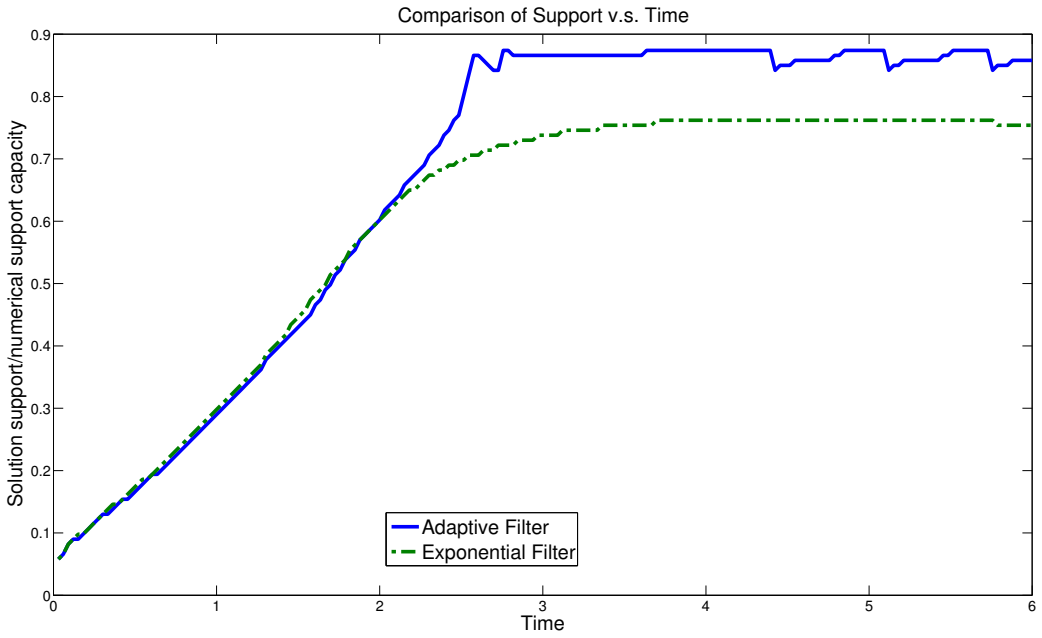


Figure 4.3: The Effective Solution Support for Example 1 as a Function of Time.

Using the parameters in Table 4.2, the problem is solved at a lower resolution and lower filter power. The results are shown in Figure 4.4. Even though the time step improvement is better, the lower resolution and smaller adaptive threshold, τ, lead to marginal accuracy gains. The solution spends very little time in the region where the adaptive filter chops and the exponential filter is diffusive.

Table 4.2: Parameter Values and Results at Low Resolution.

Equation Parameters			Filter Parameters			Results	
β	γ	N modes	p power	λ, chop threshold	τ, adaptive threshold	ω accel.	Ψ accuracy
1	18	400	8	5×10^{-13}	0.85	1.68	1.21

The parameters and results in Table 4.3 compare the behavior of the filter at various spatial resolutions. Such a comparison is difficult. We require that the support of the true solution spectrum exceeds the capacity of the numerical solution space even for high resolutions; but at low resolutions the same spectrum causes truncation error

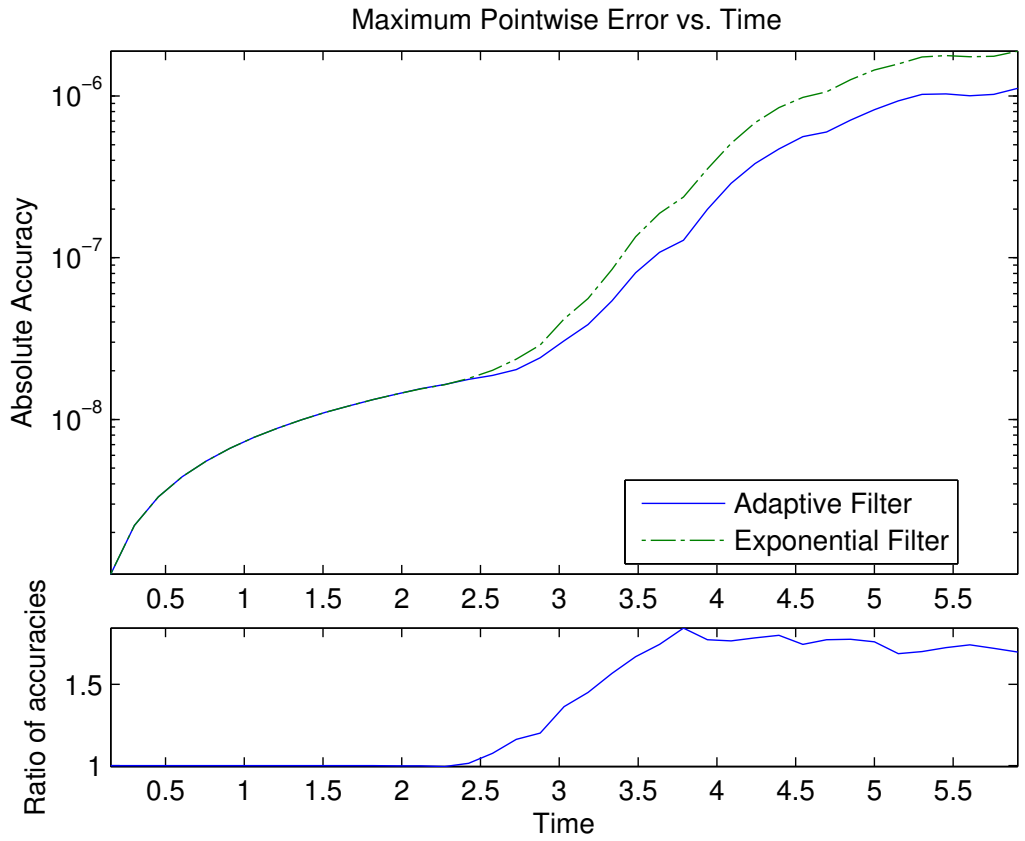


Figure 4.4: The Effect of the Adaptive Filter on Solution Accuracy Using the Parameters in Table 4.2.

to dominate. To achieve a balance we use the same equation for all resolutions, but compare the results at the moment when the support of the true solution is 1.75 times the maximum support resolved by the numerical solution space. Additionally, time step sizes that are unstable at high resolutions become stable at low resolutions eliminating the need for filtering. So, to produce a reasonable comparison the time step size needs to be dependent on spatial resolution. We use:

$$\Delta t = \alpha \Delta t_{exponential} + \alpha \Delta t_{stable} \tag{4.6}$$

Where $\Delta t_{exponential}$ is the smallest stable time step using the exponential filter and Δt_{stable} is the smallest stable time step with no filtering. We set α to 0.9 which achieves nearly the time step gains of the exponential filter (4.2), but provides room

Table 4.3: Comparison of Results Using Different Resolutions at Filter Power, $p = 12$.

N Modes	Δt Time Step	p Power	ω *Acceleration*	Ψ Accuracy	$\|u - u_{adaptive}\|_\infty$ Error
256	0.0317	12	1.43	2.52	2.56e-5
512	0.0154	12	1.45	4.53	3.657e-5
768	0.0104	12	1.47	4.70	6.39e-5
1024	0.0077	12	1.47	5.35	9.54e-5

for accuracy gains from the adaptive filter 6. For the following tests we use:

$$c(x) = \frac{1}{2}\left(\sin^2(x) + \frac{3}{2}\sin^2(2x)\right) + \frac{1}{20} \tag{4.7}$$

with the chop threshold, λ: 5×10^{-13} and the adaptive filter threshold, τ: 0.98

We see a modest improvement in accuracy between the adaptive filter and the exponential filter. The ratio gets better as the resolution grows, because the solution spends more time in the region where the adaptive filter can operate without the exponential filter. Contrary to what is normally expected, the absolute accuracy of the solution does not improve with better resolution. This is not a failure of the technique, but instead a consequence of the tests being designed to have a consistent portion of true solution support in the numerical solution space when measurements are made. The same test is performed in Table 4.4, but with lower filter power. With the lower power, larger time steps are possible, but there is also more diffusion; and the adaptive filter's accuracy improves compared to the exponential filter.

Similar numerical tests were also performed on Burger's equation and the KDV equation, showing modest accuracy improvements as well.

Table 4.4: Comparison of Results Using Different Resolutions at Filter Power, $p = 8$.

N Modes	Δt Time Step	p Power	ω *Acceleration*	Ψ Accuracy	$\|u - u_{adaptive}\|_\infty$ Error
256	0.0359	8	1.65	4.56	0.0003
512	0.0179	8	1.71	7.08	0.0005
768	0.0120	8	1.71	8.36	0.0008
1024	0.009	8	1.73	9.40	0.0011

4.4 Conclusion

The accuracy gains achieved with the adaptive filter are highly dependent on the PDE being solved and the particular parameters that are chosen as well as the degree to which the CFL condition is exceeded. Certain configurations result in very small accuracy gains. Nonetheless, in all numerical tests that were performed the adaptive filter with stability maintaining parameters outperformed the exponential filter alone. The computational and development costs of the adaptive filter are negligible, making it a simple addition to standard filtering techniques. With the current algorithm, the choice of the adaptive threshold parameter, τ, is left to the implementer. Maximum accuracy gains occur when this parameter is just below the first unstable mode. Future versions of this algorithm could determine the proper value for the parameter by analyzing the growth factors for the equation in question dynamically.

Chapter 5

CONCLUSION

The overarching conclusion of this work is that adding sparsity promoting operations to high-order numerical techniques can increase accuracy and reliability when only a subset of the required basis coefficients is available in a numerical solution or when source data samples are corrupted by noise. High-order techniques provide the foundation for accurately approximating smooth regions. The addition of sparsity promoting regularization helps to selectively eliminate high frequency artifacts and corruptions to low frequency coefficients that degrade piecewise smooth solutions.

The attempt to add a wavelet reprojection step, (2.8), to the regularization technique, (2.4), described in Archibald *et al.* (2015), resulted in a solution that was more robust with respect to the selection of the regularization parameter, λ, but this robustness was limited and came at the cost of reduced accuracy in the neighborhood of edges. The B_{low} subset of D2 wavelet basis fundamentally lacks the ability to resolve discontinuities. Various attempts were made to extend the basis with selected elements of the B_{high} set, but there was a trade-off between being able to resolve edges and the suppression of Gibbs oscillations near these edges. Other extensions to the method were considered, but the difference in accuracy of the wavelet based reconstruction in smooth regions was not substantial enough to warrant further research.

Nonetheless, the investigation was valuable, not only because it reaffirmed the effectiveness (2.4), but also because it provided insight into the behavior of (2.4) in the neighborhood of edges as well as smooth regions in the presence of noise and sub-sampling. This insight helped direct the development of the variance based edge recovery techniques. Another important development coming from the work is the

concept of an innate equality constraint in a minimization problem that seeks a solution from a class of functions defined by the subspace of a basis, i.e. by using the wavelet subspace, we force the solution to be in the class functions containing polynomials in smooth regions as opposed to only promoting this requirement through penalty terms as in Bredies *et al.* (2010).

The recovery of edges using variance is a promising technique. Concentration factor based methods are very successful at detecting true edges, but suffer from false edge detections. The addition of a variance based algorithm helps to remove many of the falsely detected edges, especially those that are far away from true edges. Treatments based on subsets of the original samples are a natural approach, but other treatments may enhance the algorithm further, especially in addressing the false negative detection errors seen in the results.

The method for reducing time stepping constraints when solving time-dependent hyperbolic PDEs was inspired by an approach based on regularization similar to (2.3) being applied at each time step in the numerical solution. That method was slow, because the added regularization required solving a non-trivial minimization problem at each time step. The added regularization also introduced bias error, which impacted the numerical solution in a manner similar to added diffusion. The modification involving only the shrink operator removed the ℓ^2 term from the regularization and therefore had minimal performance impact. It nonetheless had the effect of controlling numerical noise in unstable modes. The class of equations compatible with the adaptive filter is naturally limited, but when applicable, the filter provides real gains at almost no cost.

SAR image formation toolbox for MATLAB, vol. 7699 (2010), URL http://dx.doi.org/10.1117/12.855375.

Archibald, R., A. Gelb and R. B. Platte, "Image reconstruction from undersampled fourier data using the polynomial annihilation transform", Journal of Scientific Computing pp. 1–21, URL http://dx.doi.org/10.1007/s10915-015-0088-2 (2015).

Archibald, R., A. Gelb and J. Yoon, "Polynomial fitting for edge detection in irregularly sampled signals and images", SIAM J. Numer. Anal. **43**, 1, 259–279 (2005).

Bredies, K., K. Kunisch and T. Pock, "Total generalized variation", SIAM J. Imaging Sci. **3**, 3, 492–526 (2010).

Byrne, C. L., "Proximal minimization with bregman distances and the goldstein-osher algorithm for constrained optimization", (2015).

Candès, E. J., J. Romberg and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information", IEEE Trans. Inform. Theory **52**, 489–509 (2006).

Chen, S. S., D. L. Donoho and M. A. Saunders, "Atomic decomposition by basis pursuit", SIAM Rev. **43**, 1, 129–159, URL http://dx.doi.org/10.1137/S003614450037906X (2001).

Cheney, M. and B. Borden, *Fundamentals of Radar Imaging* (Society for Industrial and Applied Mathematics, 2009), URL http://epubs.siam.org/doi/abs/10.1137/1.9780898719291.

Daubechies, I., *Ten lectures on wavelets*, vol. 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics* (Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1992).

Daubechies, I., "Orthonormal bases of compactly supported wavelets", (1993).

Fan, J. and R. Mead, *An l1 Regularization Algorithm for Reconstructing Piecewise Smooth Functions from Fourier Data Using Wavelet Projection*, Undergraduate honors thesis, Arizona State University (2015).

Frazier, M. W., *An Introduction to Wavelets Through Linear Algebra* (Springer, 2001).

Gelb, A. and E. Tadmor, "Detection of edges in spectral data", Appl. Comp. Harmonic Anal. **7**, 101–135 (1999).

Gelb, A. and E. Tadmor, "Enhanced spectral viscosity approximations for conservation laws", Applied Numerical Mathematics **33**, 1-4, 3–21, URL http://dx.doi.org/10.1016/S0168-9274(99)00067-7 (2000).

Gelb, A. and E. Tadmor, "Adaptive edge detectors for piecewise smooth data based on the minmod limiter", J. Sci. Comput. **28**, 2-3, 279–306 (2006).

Goldstein, T. and S. Osher, "The split bregman method for l1-regularized problems", SIAM J. Imaging Sci. **2**, 2, 323–343, URL http://dx.doi.org/10.1137/080725891 (2009).

Guerquin-Kern, M., L. Lejeune, K. P. Pruessmann and M. Unser, "Realistic analytical phantoms for parallel magnetic resonance imaging", IEEE Transactions on Medical Imaging **31**, 3, 626–636, URL http://dx.doi.org/10.1109/TMI.2011.2174158 (2012).

Hesthaven, J. S., S. Gottlieb and D. Gottlieb, *Spectral methods for time-dependent problems*, Cambridge Monographs on Applied and Computational Mathematics (Cambridge University Press, 2007), URL http://dx.doi.org/10.1017/CBO9780511618352.

Ilievska, E. S. and Z. A. Ivanovski, "Customized k-space trajectory for compressed sensing mri", in "Telecommunications Forum (TELFOR), 2011 19th", pp. 631–634 (2011).

Kirby, R. M., M. Berzins and J. S. Hesthaven, eds., *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2014* (Springer International Publishing, 2015), URL http://dx.doi.org/10.1007/978-3-319-19800-2.

LeVeque, R. J., *Finite Volume Methods for Hyperbolic Problems* (Cambridge University Press, 2002).

Li, Y., R. Yang, C. Zhang, J. Zhang, S. Jia and Z. Zhou, "Analysis of generalized rosette trajectory for compressed sensing mri", Medical Physics **42**, 9, 5530–5544 (2015).

Lustig, M., D. Donoho and J. M. Pauly, "Sparse mri: The application of compressed sensing for rapid mr imaging", Magn. Reson. Med. **58**, 6, 1182–1195, URL http://dx.doi.org/10.1002/mrm.21391 (2007).

Martinez, A., A. Gelb and A. Gutierrez, "Edge detection from non-uniform fourier data using the convolutional gridding algorithm", J. Sci. Comput. **61**, 3, 490–512 (2014).

Osher, S., M. Burger, D. Goldfarb, J. Xu and W. Yin, "An iterative regularization method for total variation-based image restoration", Multiscale Model. Simul. **4**, 2, 460–489, URL http://epubs.siam.org/doi/abs/10.1137/040605412 (2005).

Osher, S. and Y. Li, "Coordinate descent optimization for $\ell 1$ minimization with application to compressed sensing; a greedy algorithm", Inverse Problems and Imaging **3**, 3, 487–503, URL http://dx.doi.org/10.3934/ipi.2009.3.487 (2009).

Petersen, A., A. Gelb and R. Eubank, "Hypothesis testing for fourier based edge detection methods", J. Sci. Comput. **51**, 608–630 (2012).

Pipe, J. G., "Motion correction with propeller mri: application to head motion and free-breathing cardiac imaging", (1999a).

Pipe, J. G., "Motion correction with propeller mri: application to head motion and free-breathing cardiac imaging", in "Magn Reson Med.;42:963?9", (1999b).

Reddy, S. C. and L. N. Trefethen, "Stability of the method of lines", Numer. Math. **62**, 1, 235–267, URL http://dx.doi.org/10.1007/BF01396228 (1992).

Richards, M., J. Scheer, J. Scheer and W. Holm, *Principles of modern radar*, no. v. 1 in Principles of Modern Radar (SciTech Publishing, Incorporated, 2010), URL https://books.google.com/books?id=nD7tGAAACAAJ.

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

Schaeffer, H., R. Caflisch, C. D. Hauck and S. Osher, "Sparse dynamics for partial differential equations", Proceedings of the National Academy of Sciences **110**, 17, 6634–6639, URL http://dx.doi.org/10.1073/pnas.1302752110 (2013).

Schatzman, J. C., "Accuracy of the discrete Fourier transform and the fast Fourier transform", SIAM J. Sci. Comput. **17**, 5, 1150–1166, URL http://dx.doi.org/10.1137/S1064827593247023 (1996).

Setzer, S., "Operator splittings, bregman methods and frame shrinkage in image processing", International Journal of Computer Vision **92**, 3, 265–280, URL http://dx.doi.org/10.1007/s11263-010-0357-3 (2010).

Shor, N. Z., *Minimization Methods for Non-Differentiable Functions* (Springer Berlin Heidelberg, 1985), URL http://dx.doi.org/10.1007/978-3-642-82118-9.

Stefan, W., R. Renaut and A. Gelb, "Improved total variation type regularizations using higher order edge detectors", SIAM J. Imaging Sci. **3**, 2, 232–2516 (2010).

Stefan, W., A. Viswanathan, A. Gelb and R. Renaut, "Sparsity enforcing edge detection method for blurred and noisy Fourier data", J. Sci. Comput. **50**, 3, 536–556, URL http://dx.doi.org/10.1007/s10915-011-9536-9 (2012).

Tadmor, E., "Filters, mollifiers and the computation of the Gibbs phenomenon", Acta Numer. **16**, 305–378, URL http://dx.doi.org/10.1017/S0962492906320016 (2007).

Wasserman, G., R. Archibald and A. Gelb, "Image reconstruction from Fourier data using sparsity of edges", J. Sci. Comput., to appear (2015).

Yan, H., *Signal processing for magnetic resonance imaging and spectroscopy*, vol. 15 (CRC Press, 2002).

Yin, W., S. Osher, D. Goldfarb and J. Darbon, "Bregman iterative algorithms for l1-minimization with applications to compressed sensing", SIAM J. Imaging Sci. **1**, 143–168 (2008).

Zuo, J., E. G. Walsh, G. Deutsch and D. B. Twieg, "Rapid mapping of flow velocity using a new parse method", Magnetic Resonance in Medicine **55**, 1, 147–152, URL `http://dx.doi.org/10.1002/mrm.20750` (2006).

APPENDIX A

THE WAVELET BASIS

Wavelets are characterized by their ability to form an orthogonal basis with localization in space and frequency. For the continuous wavelets, individual basis elements are dilations and translations of a single mother wavelet, ψ. In the case of the Daubechies Wavelets, each family has a predetermined number of vanishing moments. We now discuss the discrete Daubechies Wavelets as used in (2.5) following the formulation in Frazier (2001).

A.1 Motivation

We assume that vectors are real of length N such that $N = 2^p, p \in \mathbb{N} \setminus \{0\}$. They will be extended periodically with period N so as to be indexed by the integers, with $\vec{v}_j = \vec{v}_{j+N}$. We introduce several operators to facilitate this explanation

Definition 5 (The conjugate reflection operator)**.** *The discrete conjugate reflection operator is defined by equation* (A.1) *and has the effect of reversing the order of elements in the vector and shifting such that the first element remains in place. This reversal is followed by the element-wise complex conjugate.*

$$\left(\mathcal{C}\vec{v}\right)_j = \overline{\vec{v}_{2-j}} \tag{A.1}$$

Definition 6 (The discrete translation operator)**.** *The discrete translation operator is defined by equation* (A.2) *and has the effect of shifting the indices of the vector by k elements. Shifting by negative values is equivalent to shifting in the opposite direction.*

$$\left(\mathcal{R}_k\vec{v}\right)_j = \vec{v}_{j-k} \tag{A.2}$$

The convolution operator can then be expressed in terms of the discrete translation operator, conjugate reflection and the inner product. I.e.

$$\left(\vec{u} * \mathcal{C}\vec{v}\right)_k = \langle \vec{u}, \mathcal{R}_{k-1}\vec{v} \rangle \tag{A.3a}$$
$$\left(\vec{u} * \vec{v}\right)_k = \langle \vec{u}, \mathcal{R}_{k-1}\mathcal{C}\vec{v} \rangle \tag{A.3b}$$

Definition 7 (The down-sampling operator)**.** *The down-sampling operator is defined by equation* (A.4) *and has the effect of producing a vector of length* $\frac{N}{2}$ *composed of the odd elements of its argument.*

$$\left(\mathcal{D}\vec{v}\right)_j = \vec{v}_{2(j-1)+1} \tag{A.4}$$

The notation $\mathcal{D}^p$ *will mean p applications of the down-sampling operator.*

Definition 8 (The up-sampling operator)**.** *The up-sampling operator is defined by equation* (A.5) *and has the effect of producing a vector of length* $2N$ *with odd elements being provided by its argument and even elements being 0.*

$$\left(\mathcal{U}\vec{v}\right)_j = \begin{cases} \vec{v}_{\frac{j-1}{2}+1} & j \text{ odd} \\ 0 & j \text{ even} \end{cases} \tag{A.5}$$

The notation $\mathcal{U}^p$ *will mean p applications of the up-sampling operator. Also the composition* $\mathcal{U}\mathcal{D}\vec{v}$ *has the effect of zeroing the even elements of* $\vec{v}$*.*

As the starting point for the wavelets, we note that an orthonormal basis formed by the translations of a single basis element will fail to be localized in frequency, leading us to a basis formed by two elements. The wavelet construction therefore seeks to form an orthonormal set by using two vectors $\vec{u}$ and $\vec{v}$ as well as their even translations.

$$W := \{\mathcal{R}_{2k}\vec{u}\}_{k=0}^{\frac{N}{2}-1} \cup \{\mathcal{R}_{2k}\vec{v}\}_{k=0}^{\frac{N}{2}-1}$$

The admissibility requirements that two vectors $\vec{u}$ and $\vec{v}$, which will be referred to as seeds, and their even translates form an orthonormal set are given by

$$|\hat{u}_k|^2 + \left|\hat{u}_{k+N/2}\right|^2 = 2 \tag{A.6a}$$

$$|\hat{v}_k|^2 + \left|\hat{v}_{k+N/2}\right|^2 = 2 \tag{A.6b}$$

$$\hat{u}_j\overline{\hat{v}_j} + \hat{u}_{j+\frac{N}{2}}\overline{\hat{v}_{j+\frac{N}{2}}} = 0. \tag{A.6c}$$

A.2 The Discrete Wavelet Basis

Assuming that one seed vector $\vec{u}$ has been provided and the mean of the elements of $\vec{u}$ is non-zero, the second seed $\vec{v}$ can be found as

$$\vec{v}_j = (-1)^{|j|}\overline{\vec{u}_{2-j}}. \tag{A.7}$$

For the sake of generality, we will define $\tilde{u}$ and $\tilde{v}$ as the duals of $\vec{u}$ and $\vec{v}$ respectively such that

$$\langle \vec{u}_j, \tilde{u}_k \rangle = \delta_j^k, \langle \vec{v}_j, \tilde{v}_k \rangle = \delta_j^k.$$

The vectors $\vec{u}$ and $\vec{v}$ as well as their even translations form an analysis basis and $\tilde{u}$ and $\tilde{v}$ with their even translations form a synthesis basis. Any vector $\vec{z}$ can be reconstructed with the following decomposition

$$\vec{z} = \sum_{k=0}^{\frac{N}{2}-1} \langle \vec{z}, \mathcal{R}_{2k}\vec{v} \rangle \, \mathcal{R}_{2k}\tilde{v} + \sum_{k=0}^{\frac{N}{2}-1} \langle \vec{z}, \mathcal{R}_{2k}\vec{u} \rangle \, \mathcal{R}_{2k}\tilde{u}. \tag{A.8}$$

In the restricted case of the orthogonal wavelets $\tilde{u} = \overline{\vec{u}}$ and $\tilde{v} = \overline{\vec{v}}$. Using (A.3a) and (A.3b), we find that (A.8) is equivalent to

$$\vec{z} = \tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right)\right). \tag{A.9}$$

A.2.1 Multi-resolution Analysis

We will call the reconstruction, (A.9), $\mathcal{S}$. Noting that $\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right)$ is a vector of length $\frac{N}{2}$, it too can be decomposed using $\mathcal{S}$ thus

$$\mathcal{S}\left(z\right) = \tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\mathcal{S}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right)\right)\right) \tag{A.10}$$

Such nesting can occur recursively with $\mathcal{S}$ operating on ever smaller vectors. We let

$$\vec{z}_1 = z$$

$$\vec{z}_j = \mathcal{D}\left(\vec{z}_{j-1} * \mathcal{C}\vec{u}\right)$$

so

$$\mathcal{S}\left(z_j\right) = \tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\vec{z}_j * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\mathcal{S}\left(\vec{z}_{j+1}\right)\right). \tag{A.11}$$

This process establishes a so-called Multi-Resolution Analysis with the multi-scale properties of the continuous wavelets. The reconstruction, (A.9) uses convolutions, which can be calculated using the FFT and the successive decompositions in (A.11) operate on ever smaller vectors forming a convergent geometric series; thus, MRAs operate with $\mathcal{O}\left(N\log_2 N\right)$ performance similar to the FFT.

A.2.2 Wavelet Basis Formation

The MRA described in (A.11) performs well, but this formulation is incompatible with software requiring an explicit linear operator and it is difficult to analyze. We therefore seek a set of basis vectors allowing for a pair of linear transformations that are equivalent to (A.11). We define the sets of vectors $\vec{\phi}_{j,k}, \vec{\psi}_{j,k}$ and associated duals $\tilde{\phi}_{j,k}, \tilde{\psi}_{j,k}$ where j represents a stage and k represents a translation such that

$$\vec{\psi}_{j,k} = \mathcal{R}_{2^{j-1}(k-1)}\vec{\psi}_{j,1} \quad k = 1 \cdots \frac{N}{2^{j-1}}$$

We let

$$\vec{\psi}_{1,1} = \vec{v},\ \tilde{\psi}_{1,1} = \tilde{v}, \vec{\phi}_{1,1} = \vec{u},\ \tilde{\phi}_{1,1} = \tilde{u}$$

To direct us to the form of the elements in the bases, we expand the recursion (A.11) by one step.

$$\vec{z} = \tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right) * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\mathcal{D}\left(\mathcal{S}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right)\right) * \mathcal{C}\vec{u}\right)\right)\right)$$

Then by linearity

$$\vec{z} = \tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{v}\right)\right) + \tilde{u} * \mathcal{U}\left(\tilde{v} * \mathcal{U}\left(\mathcal{D}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right) * \mathcal{C}\vec{v}\right)\right)\right) + \tilde{u} * \mathcal{U}\left(\tilde{u} * \mathcal{U}\left(\mathcal{D}\left(\mathcal{S}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{u}\right)\right) * \mathcal{C}\vec{u}\right)\right)\right)$$

We seek a form

$$\begin{aligned}\vec{z} = \tilde{\psi}_{1,1} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{\psi}_{1,1}\right)\right) + \tilde{\psi}_{2,1} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{\psi}_{2,1}\right)\right) + \cdots + \\ \tilde{\psi}_{\log_2 N,1} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{\psi}_{\log_2 N,1}\right)\right) + \tilde{\phi}_{\log_2 N,1} * \mathcal{U}\left(\mathcal{D}\left(\vec{z} * \mathcal{C}\vec{\phi}_{\log_2 N,1}\right)\right)\end{aligned} \tag{A.12}$$

This is accomplished by letting

$$\vec{\psi}_{j+1,1} = \vec{\phi}_{j,1} * \mathcal{U}^j\left(\vec{\psi}_{1,1}\right), \vec{\phi}_{j+1,1} = \vec{\phi}_{j,1} * \mathcal{U}^j\left(\vec{\phi}_{1,1}\right), \tilde{\psi}_{j+1,1} = \tilde{\phi}_{j,1} * \mathcal{U}^j\left(\tilde{\psi}_{1,1}\right), \tilde{\phi}_{j+1,1} = \tilde{\phi}_{j,1} * \mathcal{U}^j\left(\tilde{\phi}_{1,k}\right) \tag{A.13}$$

A.3 The Daubechies Wavelets

To satisfy the admissibility requirements (A.6a), the Daubechies Wavelets employ powers of the well known trigonometric identity.

$$\cos^2\theta + \sin^2\theta = 1$$

The Daubechies Wavelets form families, 'DK', where K represents the number of vanishing moments in the corresponding wavelet, Daubechies (1992, 1993). Also, $2K$ is the number of non-zero entries in the seed vector $\vec{u}$. We let $k = 2K - 1$. Consider the polynomial

$$\left(\cos^2\frac{j\pi}{N} + \sin^2\frac{j\pi}{N}\right)^k = 1$$

evaluated at N points $\frac{j\pi}{N}, j = 0 \cdots N-1$, where N is the defined as in Section A.1. We expand the polynomial using the binomial theorem

$$\left(\cos^2\frac{j\pi}{N} + \sin^2\frac{j\pi}{N}\right)^k = \sum_{m=0}^{k} \left(\begin{array}{c} k \\ m \end{array} \right) \left(\cos\left(\frac{j\pi}{N}\right)\right)^{2m} \left(\sin\left(\frac{j\pi}{N}\right)\right)^{2(k-m)} = 1 \quad \text{(A.14)}$$

With k being odd, the number of terms in (A.14) is even. We isolate and label the first half of the terms (A.14). Let

$$b\left(j\right) = \sum_{m=0}^{\frac{k-1}{2}} \left(\begin{array}{c} k \\ m \end{array} \right) \left(\cos\left(\frac{j\pi}{N}\right)\right)^{2m} \left(\sin\left(\frac{j\pi}{N}\right)\right)^{2(k-m)}$$

We use the fact that there is $\frac{\pi}{2}$ phase difference between sin and cos.

$$\cos\left(\frac{\left(j+\frac{N}{2}\right)\pi}{N}\right) = -\sin\left(\frac{j\pi}{N}\right), \sin\left(\frac{\left(j+\frac{N}{2}\right)\pi}{N}\right) = \cos\left(\frac{j\pi}{N}\right)$$

Thus, the second half of the terms in (A.14) can be expressed as

$$b\left(j+\frac{N}{2}\right) = \sum_{\frac{k-1}{2}=0}^{k} \left(\begin{array}{c} k \\ m \end{array} \right) \left(\cos\left(\frac{j\pi}{N}\right)\right)^{2m} \left(\sin\left(\frac{j\pi}{N}\right)\right)^{2(k-m)}$$

Thus,

$$b\left(j\right) + b\left(j+\frac{N}{2}\right) = 1$$

If we choose

$$\left|\hat{u}\left(j\right)\right|^2 = 2b\left(j\right) \quad \text{(A.15)}$$

then

$$\left|\hat{u}\left(j+\frac{N}{2}\right)\right|^2 = 2b\left(j+\frac{N}{2}\right) \text{ and } \left|\hat{u}\left(j\right)\right|^2 + \left|\hat{u}\left(j+\frac{N}{2}\right)\right|^2 = 2$$

as required by the admissibility condition (A.6a).

With $\left|e^{i\phi}\hat{u}\left(j\right)\right| = \left|\hat{u}\left(j\right)\right|$, we are left with free phase parameters for all of the Fourier coefficients and need to choose these phase parameters such that the physical space support of the wavelets is minimized. This can be found analytically. Alternatively,

we use knowledge of the support location to establish a system of equations using the discrete Fourier transform.

$$\hat{u}_j = \sum_{m=1}^{k+1} u_m e^{\frac{-2\pi i(m-1)(j-1)}{N}} \tag{A.16}$$

Thus the scaling function seed, $\vec{u}$, has been generated and $\vec{v}$ can be found by (A.7).

APPENDIX B

THE SPLIT BREGMAN ALGORITHM

B.1 Introduction

We examine the Split-Bregman algorithm for solving regularization problems of the form:

$$\underset{\vec{u}}{\operatorname{argmin}} \left(\left\| \mathbf{A}\vec{u} - \vec{b} \right\|_2 + \mu \|\mathbf{E}\vec{u}\|_1 \right) \tag{B.1}$$

This algorithm is able to solve a more general class of problem:

$$\underset{\vec{u}}{\operatorname{argmin}} \|\Phi(\vec{u})\|_1 + H(\vec{u}) \tag{B.2}$$

where $\|\Phi(u)\|_1, H(u)$ are convex, $\min_u H(u) = 0$, and $H(u), \Phi(u)$ are differentiable.

Below are outlines of the Generalized Split-Bregman Algorithm, Algorithm 7, and its antecedent, Algorithm 8. Each major component will be examined in detail following discussions in Goldstein and Osher (2009); Yin *et al.* (2008).

Algorithm 7 Generalized Split Bregman Algorithm

1: **while** convergence criterion not met **do**
2: **for** $n = 1$ to N **do** ▷ A small number of interior iterations
3: $u^{k+1} = \underset{u}{\operatorname{argmin}} H(u) + \frac{\lambda}{2} \left\| d^k - \Phi(u) - b^k \right\|_2^2$
4: $d^{k+1} = \underset{d}{\operatorname{argmin}} \|d\|_1 + \frac{\lambda}{2} \left\| d - \Phi\left(u^{k+1}\right) - b^k \right\|_2^2$
5: $b^{k+1} = b^k + \left(\Phi\left(u^{k+1}\right) - d^{k+1} \right)$
6: $k = k + 1$

Algorithm 8 Split Bregman without Feedback

1: **while** $\left\| u^k - u^{k-1} \right\|_2 > tol$ **do**
2: $u^{k+1} := \underset{u}{\operatorname{argmin}} E_u\left(u, d^k\right) - p_u^k\left(u^k, d^k\right) \cdot \left(u - u^k\right) + \frac{\lambda}{2} \left\| d^k - \Phi(u) \right\|_2^2$
3: $d^{k+1} := \underset{d}{\operatorname{argmin}} E_d\left(u^k, d\right) - p_d^k\left(u^k, d^k\right) \cdot \left(d - d^k\right) + \frac{\lambda}{2} \left\| d - \Phi(u_k) \right\|_2^2$
4: $p_u^{k+1} = p_u^k + \lambda \nabla \Phi\left(d^{k+1} - \Phi\left(u^{k+1}\right)\right)$
5: $p_d^{k+1} = p_d^k - \lambda\left(d^{k+1} - \Phi\left(u^{k+1}\right)\right)$
6: $k = k + 1$

B.2 Splitting

First, we examine the "Split" in Split-Bregman which leads to the variable, d, in lines 3 and 4 in Algorithm 7. Problems such as (B.2) have a mixture of terms using the 1-norm and the 2-norm. For example, H may represent a least squares fidelity term, as in $\left\| \mathbf{A}\vec{u} - \vec{b} \right\|_2$ from (B.1); and Φ might represent a gradient leading to a total variation regularization term. The mixed norms and the lack of differentiability for the 1-norm make the general problem, (B.2), difficult to solve. Nonetheless, there are

known fast techniques for solving simpler problems, namely the conjugate gradient method for solving convex problems involving strictly 2-norms and shrinkage or soft thresholding for solving a class of problem typically referred to as Basis Pursuit, Chen *et al.* (2001), and is given by,

$$\underset{\vec{u}}{\operatorname{argmin}} \|\vec{u}\|_1 \text{ subject to } \mathbf{A}\vec{u} = \vec{b} \tag{B.3}$$

We discuss the solution to this sub-problem later, but the first step is to transform (B.2) so that the Basis Pursuit solution can be used.

We simplify the 1-norm term and convert (B.2) into a constrained optimization problem by letting $\vec{d} = \Phi(\vec{u})$.

$$\underset{\vec{u},\vec{d}}{\operatorname{argmin}} \left\|\vec{d}\right\|_1 + H(\vec{u}) \text{ subject to } \vec{d} = \Phi(\vec{u}) \tag{B.4}$$

This constrained optimization problem is difficult as well, so we attempt to solve it as an unconstrained problem, penalizing any departure from the constraint. We create a new convex and differentiable term representing the error in the constraint

$$\frac{\lambda}{2} \|d - \Phi(\vec{u})\|_2^2 \tag{B.5}$$

This is then added to the objective function to remove the constraint, yielding

$$\underset{\vec{u},\vec{d}}{\operatorname{argmin}} \left\|\vec{d}\right\|_1 + H(\vec{u}) + \frac{\lambda}{2} \|d - \Phi(\vec{u})\|_2^2 \tag{B.6}$$

Remark. *It is important to note for the explanation that follows, in this regularized form, d no longer has an explicit dependence on u and is allowed to move freely.*

The left two terms of (B.6) represent the original objective function that we are trying to minimize. We let $E\left(\vec{u},\vec{d}\right) = \left\|\vec{d}\right\|_1 + H(\vec{u})$, so (B.6) becomes

$$\underset{\vec{u},\vec{d}}{\operatorname{argmin}} E\left(\vec{u},\vec{d}\right) + \frac{\lambda}{2} \left\|\vec{d} - \Phi(\vec{u})\right\|_2^2 \tag{B.7}$$

We now introduce the Bregman distance as a means to solve (B.7).

B.2.1 The Bregman Distance

We begin by defining the Bregman distance for differentiable functions and then generalize it to non-differentiable functions.

Definition 9 (The Bregman distance for differentiable functions)**.** *Let f be a convex, differentiable function. The Bregman distance is defined as*

$$D_f(\vec{v},\vec{u}) = f(\vec{v}) - (f(\vec{u}) + (\nabla f)(\vec{u}) \cdot (\vec{v} - \vec{u})) \tag{B.8}$$

Interpreting (B.8), we have $f(\vec{v})$ minus the linear approximation to $f(\vec{v})$ using the value and gradient at $\vec{u}$. This is not a true metric, since it violates symmetry and the triangle inequality; but it is positive semi-definite for convex functions. Also, the Bregman distance will decrease monotonically as $\vec{v}$ approaches $\vec{u}$ along the line between them.

We now introduce the sub-gradient and sub-differential.

Definition 10 (The subgradient)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex. A subgradient at a point $\vec{x}_0$ is a vector $\vec{v}$ such that:*

$$f(\vec{x}) - f(\vec{x}_0) \geqslant \vec{v} \cdot (\vec{x} - \vec{x}_0), \quad \forall \vec{x} \in \mathbb{R}^N$$

Following the convention standard in the imaging community we will use $\vec{p}_x(\vec{x}_0)$ to denote the subgradient at $\vec{x}_0$. If f is differentiable then the subgradient takes on a single value at all points $\vec{x}_0$. On the other hand, if f is not differentiable then it is possible to have a set of subgradients at points lacking differentiability.

Definition 11 (The subdifferential)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex. The subdifferential of f at $\vec{x}_0$ is a set valued function:*

$$(\partial f)(\vec{x}_0) := \{v : f(\vec{x}) - f(\vec{x}_0) \geqslant \vec{v} \cdot (\vec{x} - \vec{x}_0)\}$$

In one dimension, the subdifferential of f at a point x_0, $\partial f(x_0)$, is the set of all slopes such that no line passing through $\vec{x}_0$ crosses f, although tangency is allowed. If f is convex then $0 \in \partial f(\vec{x}_0)$ if f has a minimum at x_0. This is illustrated in Figure B.1.

Since $\|d\|_1$ is not differentiable in (B.7), we use the subgradient with the following Bregman distance:

Definition 12 (The Bregman distance for non-differentiable functions)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex and let $\vec{p}_u$ be the subgradient as defined in Definition 10, then the Bregman Distance is defined as*

$$D_f^p(\vec{v}, \vec{u}_0) = f(\vec{v}) - (f(\vec{u}_0) + \vec{p}_u(\vec{u}_0) \cdot (\vec{v} - \vec{u}_0)) \tag{B.9}$$

B.2.2 The Iterative Solution

In order to solve the mixed norm problem, (B.6), iteratively, we split (B.7) into a minimization for u and d and set up an iteration by defining two separate functionals:

$$E_u(\vec{u}) = H(\vec{u}) \tag{B.10a}$$

$$E_d\left(\vec{d}\right) = \left\|\vec{d}\right\|_1 \tag{B.10b}$$

We solve these functionals independently while still respecting the constraint. A single iteration, k, of the solution to (B.7) is therefore

$$\vec{u}^{k+1} := \underset{\vec{u}}{\operatorname{argmin}} E_u(\vec{u}) + \frac{\lambda}{2}\left\|\vec{d}^k - \Phi(\vec{u})\right\|_2^2 \tag{B.11a}$$

$$\vec{d}^{k+1} := \underset{\vec{d}}{\operatorname{argmin}} E_d\left(\vec{d}\right) + \frac{\lambda}{2}\left\|\vec{d} - \Phi\left(\vec{u}^{k+1}\right)\right\|_2^2 \tag{B.11b}$$

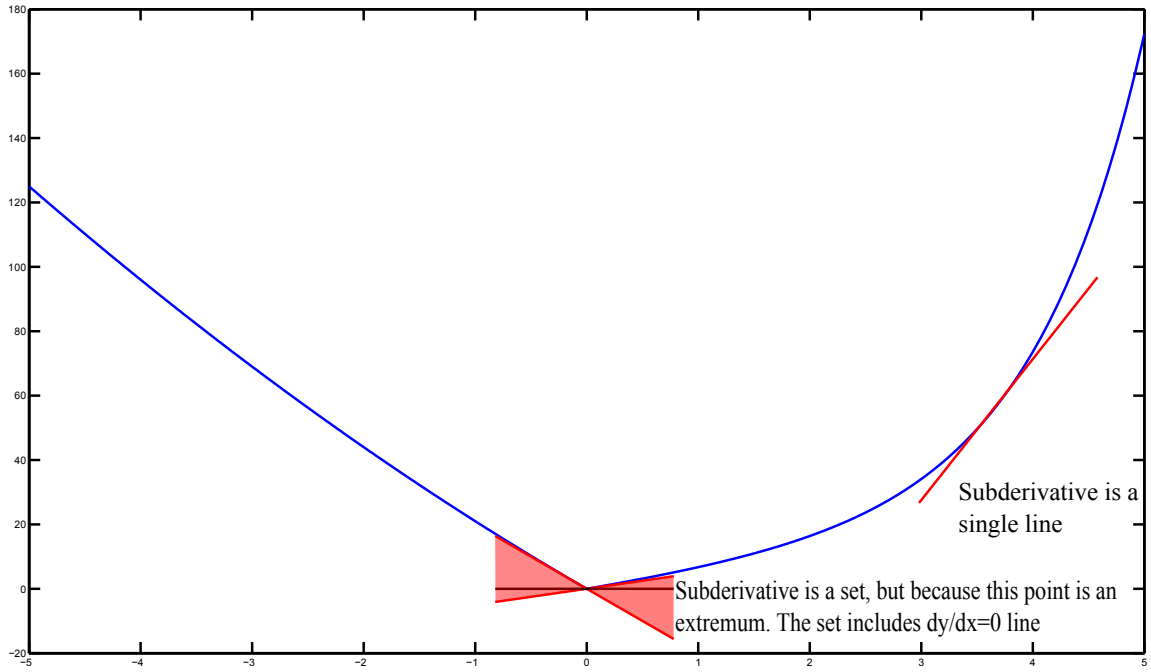


Figure B.1: The Subdifferentials at Differentiable and Non-differentiable Points in One Dimension.

We justify this splitting by showing that an iteration which is contractive for both (B.11a) and (B.11b) will be contractive for (B.7) as well. Note that (B.10a) has no dependence on $\vec{d}$ and as noted earlier, $\vec{d}$ has no direct dependence on $\vec{u}$. Thus, after an application of (B.11a) we have

$$H\left(\vec{u}^{k+1}\right)+\frac{\lambda}{2}\left\|\vec{d}^{k}-\Phi\left(\vec{u}^{k+1}\right)\right\|_{2}^{2} \leqslant H\left(\vec{u}^{k}\right)+\frac{\lambda}{2}\left\|\vec{d}^{k}-\Phi\left(\vec{u}^{k}\right)\right\|_{2}^{2}$$

for all possible $\vec{u}^{k}$. Moreover, since $\vec{d}^{k}$ is held fixed, $\left\|\vec{d}^{k}\right\|_{1}$ is unchanged and thus the functional in (B.6) cannot grow.

$$\left\|\vec{d}^{k}\right\|_{1}+H\left(\vec{u}^{k+1}\right)+\frac{\lambda}{2}\left\|d^{k}-\Phi\left(\vec{u}^{k+1}\right)\right\|_{2}^{2} \leqslant \left\|\vec{d}^{k}\right\|_{1}+H\left(\vec{u}^{k}\right)+\frac{\lambda}{2}\left\|d^{k}-\Phi\left(\vec{u}^{k}\right)\right\|_{2}^{2}$$

The identical argument holds for (B.11b). Thus we have a contraction for (B.6).

While (B.11a) and (B.11b) could be solved by gradient descent using the subgradient, such techniques are slow to converge with convergence rate $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$, Shor (1985). A more efficient method is to use the Bregman distance as defined in Definition 12 to establish a proximal minimization algorithm Byrne (2015). Thus we replace E_u and E_d in (B.11a) and (B.11b) with their associated Bregman distances to obtain

$$D_{E_u}^{p_u}\left(\vec{u},\vec{u}^{k}\right)=E_u\left(\vec{u}\right)-\left(E_u\left(\vec{u}^{k}\right)+\vec{p}_u^{k}\left(\vec{u}^{k}\right)\cdot\left(\vec{u}-\vec{u}^{k}\right)\right) \tag{B.12a}$$

$$D_{E_d}^{p_d}\left(\vec{d},\vec{d}^{k}\right)=E_d\left(\vec{d}\right)-\left(E_d\left(\vec{d}^{k}\right)+\vec{p}_d^{k}\left(\vec{d}^{k}\right)\cdot\left(\vec{d}-\vec{d}^{k}\right)\right) \tag{B.12b}$$

In (B.12a) we evaluate along the $\vec{u}$, $\vec{u}^k$ line holding $\vec{d}^k$ fixed and conversely for (B.12b). Since we are solving a minimization problem, $E_u\left(\vec{u}^k\right), E_d\left(\vec{d}^k\right)$ represent uniform vertical shifts in the surfaces and are irrelevant to the minimization. Combining (B.12a), (B.12b), (B.11a) and (B.11b) therefore yields

$$\vec{u}^{k+1} := \underset{\vec{u}}{\operatorname{argmin}} E_u\left(\vec{u}\right) - \vec{p}_u^k\left(\vec{u}^k\right) \cdot \left(\vec{u} - \vec{u}^k\right) + \frac{\lambda}{2}\left\|\vec{d}^k - \Phi\left(\vec{u}\right)\right\|_2^2 \tag{B.13a}$$

$$\vec{d}^{k+1} := \underset{\vec{d}}{\operatorname{argmin}} E_d\left(\vec{d}\right) - \vec{p}_d^k\left(\vec{d}^k\right) \cdot \left(\vec{d} - \vec{d}^k\right) + \frac{\lambda}{2}\left\|\vec{d} - \Phi\left(\vec{u}_k\right)\right\|_2^2 \tag{B.13b}$$

Determining the Subgradients

In (B.13a) and (B.13b) we assume the subgradients $\vec{p}_u^k, \vec{p}_d^k$ are known. We now develop a method for determining them. Recall from Definition 11 that at an extremum, 0 is a member of the subdifferential. Thus, since $\vec{u}^{k+1}$ solves (B.13a), 0 is a member of the subdifferential of the objective function in (B.13a) when evaluated at $\vec{u}^{k+1}$. Similarly, 0 is a member of the subdifferential of the objective function in (B.13b) when evaluated at $\vec{d}^{k+1}$.

Finally, by (B.12a), (B.12b) and Definition 12 we have

$$\vec{p}_u^{k+1} \in \left(\partial E_u\right)\left(\vec{u}^{k+1}\right) \text{ and } \quad \vec{p}_d^{k+1} \in \left(\partial E_d\right)\left(\vec{d}^{k+1}\right) \tag{B.14}$$

Differentiating (B.13a) and applying the argument above implies

$$0 \in \partial_{\vec{u}}\left(E_u\left(\vec{u}\right) - \vec{p}_u^k\left(\vec{u}^k\right) \cdot \left(\vec{u} - \vec{u}^k\right) + \frac{\lambda}{2}\left\|\vec{d}^k - \Phi\left(\vec{u}\right)\right\|_2^2\right) \Bigg| \; \vec{u} = \vec{u}^{k+1}$$

which leads to

$$0 = \vec{p}_u^{k+1}\left(\vec{u}^{k+1}\right) + \partial_{\vec{u}}\left(-\vec{p}_u^k\left(\vec{u}^k\right) \cdot \left(\vec{u} - \vec{u}^k\right) + \frac{\lambda}{2}\left\|\vec{d}^k - \Phi\left(\vec{u}\right)\right\|_2^2\right) \Bigg| \; \vec{u} = \vec{u}^{k+1}$$

when solving for 0 and using (B.14). Note that ∂ represents a sub-gradient in this context. Next, observing that $\vec{u}^k$ and $\vec{d}^k$ are constants and $\partial\left(\vec{p}_u^k\left(\vec{u}^k\right) \cdot \vec{u}\right) = \sum_j \frac{\partial}{\partial \vec{u}_j}\left(\vec{p}_u^k\left(\vec{u}^k\right)\right)_j \vec{u}_j = \vec{p}_u^k\left(\vec{u}^k\right)$, we have

$$0 = \vec{p}_u^{k+1}\left(\vec{u}^{k+1}\right) - \vec{p}_u^k\left(\vec{u}^k\right) + \lambda\left(\nabla\Phi\right)\left(\vec{u}^{k+1}\right)\left(\vec{d}^k - \Phi\left(\vec{u}^{k+1}\right)\right)$$

or equivalently

$$\vec{p}_u^{k+1}\left(\vec{u}^{k+1}\right) = \vec{p}_u^k\left(\vec{u}^k\right) - \lambda\left(\nabla\Phi\right)\left(\vec{u}^{k+1}\right)\left(\vec{d}^k - \Phi\left(\vec{u}^{k+1}\right)\right) \tag{B.15}$$

A similar construction for the (B.13b) yields

$$\vec{p}_d^{k+1}\left(\vec{d}^{k+1}\right) = \vec{p}_d^k\left(\vec{d}^k\right) - \lambda\left(\vec{d}^{k+1} - \Phi\left(\vec{u}^{k+1}\right)\right) \tag{B.16}$$

The basis for Algorithm 8 can then be obtained by combining (B.13a), (B.13b), (B.15), and (B.16) as

$$u^{k+1} := \underset{u}{\operatorname{argmin}}\, E_u\left(u, d^k\right) - p_u^k\left(u^k, d^k\right) \cdot \left(u - u^k\right) + \frac{\lambda}{2}\left\|d^k - \Phi\left(u\right)\right\|_2^2 \tag{B.17a}$$

$$d^{k+1} := \underset{d}{\operatorname{argmin}}\, E_d\left(u^k, d\right) - p_d^k\left(u^k, d^k\right) \cdot \left(d - d^k\right) + \frac{\lambda}{2}\left\|d - \Phi\left(u_k\right)\right\|_2^2 \tag{B.17b}$$

$$p_u^{k+1} = p_u^k\left(\vec{u}^{k+1}\right) + \lambda\left(\nabla\Phi\right)\left(d^{k+1} - \Phi\left(u^{k+1}\right)\right) \tag{B.17c}$$

$$p_d^{k+1} = p_d^k\left(\vec{d}^{k+1}\right) - \lambda\left(d^{k+1} - \Phi\left(u^{k+1}\right)\right) \tag{B.17d}$$

B.3 Adding Feedback to Simplify the Iteration

We now examine the evolution of Algorithm 8 to Algorithm 7. The idea of modifying b in line 5 of Algorithm 7 is often referred to as "adding the noise back into the iteration". This characterization was first used when solving a very specific problem, Total Variation Denoising or Rudin-Osher-Fatemi (ROF) Denoising given by (B.18), Osher *et al.* (2005). In the more general context of solving (B.2), the technique facilitates convergence in an analogous way, Goldstein and Osher (2009), but the b term may or may not represent noise in the solution. In the context of (B.2) b is representative of errors in the constraint term (B.5) during prior iterations.

To understand the source of the phrase "adding the noise back into the iteration" we look at the ROF algorithm in detail. The minimization problem for ROF denoising is given by

$$c = \underset{\vec{u}}{\operatorname{argmin}} \left\|u\right\|_{BV} + \mu\left\|m - u\right\|_2^2 \tag{B.18}$$

where c is a noise reduced image, m is a measured, noisy source image, μ is a regularization parameter and BV is the bounded variation,

$$\left\|u\right\|_{BV} = \int \left|\nabla u\right| \tag{B.19}$$

The one dimensional discrete form of (B.19) is $\left\|D\vec{u}\right\|_1$, where D is a differentiation matrix.

Changing μ in (B.18) affects the relative dominance of the BV term, thus the amount of smoothing that occurs in the solution.

The technique described in Osher *et al.* (2005) is given in Algorithm 9.

Algorithm 9 Total Variation Denoising with Feedback

1: $u^0 = 0$
2: $b^0 = 0$
3: **while** not stopping criterion met **do**
4: $\quad u^{k+1} = \underset{u}{\operatorname{argmin}} \left\|Du\right\|_1 + \mu\left\|m + b^k - u\right\|_2^2$
5: $\quad b^{k+1} = b^k + m - u^{k+1}$
6: $\quad k = k + 1$

Consider the first iteration of Algorithm 9 for the one dimensional ROF problem. After the first iteration, $\vec{u}^1$ represents a smoothed version of the original signal m.

Thus, in line 5, $\vec{m} - \vec{u}^1$ represents noise and texture, which is then added into the fidelity term on line 4 during the next iteration. Hence, we are led to the concept of adding the noise back into the iteration.

The literature provides various perspectives on why adding the noise back into the iteration is effective, but most intuitive explanations seem to break down after the first few iterations. In particular, some explanations make note that during the initial step the minimization extracts the smooth portion of the signal, which then leads to b representing noise and texture. This then leads $m + b$ being more noisy on the next iteration. The explanations then assume that over time more and more of the original noise appears in the result of each iteration. Numerical tests show that some iterations have a result that is smoother locally than their predecessors or show fluctuations completely unrelated to the initial noise. For example, a gradient descent step can lead to a solution which is better in an l_2 sense than its predecessor, yet has individual elements which are worse. Nonetheless, Algorithm 9 has been shown to be equivalent to iterations using the Bregman distance as in Algorithm 8, Goldstein and Osher (2009); Osher *et al.* (2005).

For simplicity in notation we let $J(\vec{u}) = \|\vec{u}\|_{BV}$. Now consider the Bregman distance for $J(\vec{u})$ with respect to the prior iteration, $\vec{u}^k$, given by

$$D_J^p\left(\vec{u}, \vec{u}^k\right) = J\left(\vec{u}\right) - \left(J\left(\vec{u}^k\right) + \vec{p}_u\left(\vec{u}^k\right) \cdot \left(\vec{u} - \vec{u}^k\right)\right) \tag{B.20}$$

Using (B.20) we create a new functional to replace the functional in (B.18), yielding

$$J\left(\vec{u}\right) - \left(J\left(\vec{u}^k\right) + \vec{p}_u\left(\vec{u}^k\right) \cdot \left(\vec{u} - \vec{u}^k\right)\right) + \mu \left\|m - u\right\|_2^2 \tag{B.21}$$

Analogous to the process used to obtain (B.15),

$$\vec{p}_u^{k+1}\left(\vec{u}^{k+1}\right) = \vec{p}_u^k\left(\vec{u}^k\right) + 2\mu\left(\vec{m} - \vec{u}^k\right) \tag{B.22}$$

As described in Osher *et al.* (2005), defining $\vec{b}^k = \frac{\vec{p}^k\left(\vec{u}^k\right)}{2\mu}$ and making the replacement in (B.22) yields

$$\vec{b}^{k+1} = \vec{b}^k + \left(\vec{m} - \vec{u}^k\right)$$

i.e., line 5 of Algorithm 9.

We assume that μ is chosen so as to produce a smoothed result in (B.18). It is shown in Osher *et al.* (2005) that asymptotically $\vec{u}^k \to \vec{m}$. Thus, as the iterations progress, $\vec{u}^k$ evolves from a smooth signal to the original noisy signal. A frequently chosen stopping criterion is the discrepancy principle – stopping when the noise level in the iteration is of the scale of some pre-determined noise variance.

We compare Algorithm 9 to a Split Bregman implementation of the same technique, Algorithm 10. In Algorithm 10 on line 6, b^k is no longer adjusted by $m - u^k$, the difference between the smooth image and the measured image, but instead by $Du^{k+1} - d^{k+1}$, the error in the iteration's ability to conform to the constraint (B.5). For the algorithm to properly solve the regularization problem $Du^k - d^k \to 0$ for large k, but the expression "adding back the noise" does not really describe this procedural step. The Split Bregman algorithm has been shown to be equivalent to other optimization algorithms, in particular the Bregman method, an Augmented Lagrangian method, and a Douglas-Rachford splitting method, Setzer (2010).

Algorithm 10 Split Bregman ROF Iteration with Feedback

1: $b^0 = 0$
2: **while** not stopping criterion met **do**
3: **for** $n = 1$ to N **do** ▷ A small number of interior iterations
4: $u^{k+1} = \underset{u}{\operatorname{argmin}} \frac{1}{2} \|m - u\|_2^2 + \frac{\lambda^2}{2} \left\|Du - \left(d^k - b^k\right)\right\|_2^2$
5: $d^{k+1} = \underset{d}{\operatorname{argmin}} \mu_s \|d\|_1 + \frac{\lambda^2}{2} \left\|d - \left(Du^{k+1} + b^k\right)\right\|_2^2$
6: $b^{k+1} = b^k + Du^{k+1} - d^{k+1}$
7: $k = k + 1$

B.4 Solving the Sub-problems

B.4.1 The Fidelity Term

Line 3 of Algorithm 7 involves finding the minimizer of the fidelity term of the original problem as well as the regularization term linking the two sub-problems. All the components are differentiable and therefore can be solved directly or through well known optimization techniques. It has been recommended that in the context of Split-Bregman with feedback, iterative minimizers only be run for a few steps as the regularization term will change with each step in the outer loop. In spite of solving the sub-problems inaccurately, as long as the solutions are contractive, overall convergence still occurs as the outer loop progresses. This property is known as "error forgetting".

B.4.2 The Basis Pursuit Problem

Line (4) of Algorithm 7 involves finding the solution to a Basis Pursuit problem, (B.3). Following comments in Osher and Li (2009), the explanation below describes how the shrink operator solves the Basis Pursuit problem in one dimension. Let us consider

$$\underset{u}{argmin} \left(\|\vec{u}\|_1 + \lambda \|A\vec{u} - f\|_2^2\right)$$

and its scalar equivalent

$$\underset{u}{argmin} \left(|u| + \lambda(cu - b)^2\right) \tag{B.23}$$

For simplicity, let us also assume that $\lambda = 1, c = 1$, with b being the only free parameter. Figure B.2 demonstrates a simple Basis Pursuit problem.

To demonstrate the effect of the shrink operator we examine various cases:

- **Case 1:** Assume that the minimum occurs at a point where the functional in (B.23) is differentiable, i.e. at any point other than $u = 0$. It will be shown that this occurs when $|2\lambda bc| > 1$. We will further assume that $bc > 0$. This forces the minimum of $(cu - b)^2$ to occur on $0 < u < \infty$. These assumptions will lead to the minimum of (B.23) occurring on $0 < u < \infty$. This case is shown in Figure B.3.

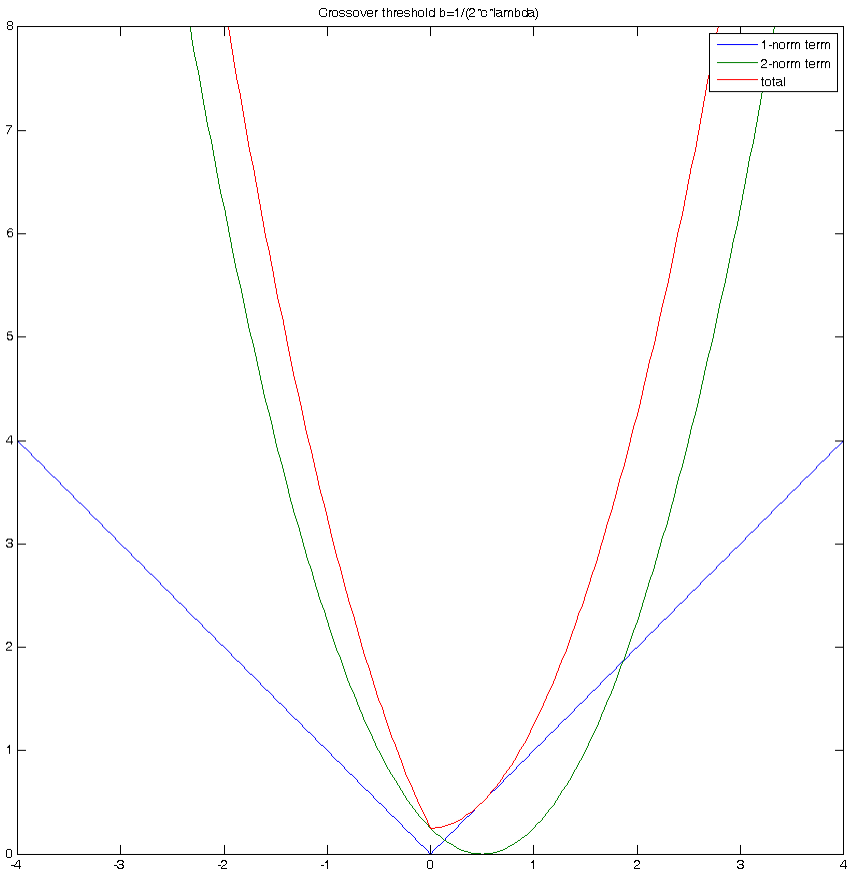


Figure B.2: A Simple Basis Pursuit Problem Showing the Combined Functional As Well As the 1-norm and 2-norm Components.

We replace $|u|$ with u in (B.23) and expand to obtain

$$\lambda c^2 u^2 + (1 - 2\lambda cb)\, u + \lambda b^2$$

We differentiate and seek the requirement that this minimum occurs in $0 < u < \infty$

$$2\lambda c^2 u + (1 - 2\lambda cb) = 0$$

The minimum occurs in $0 < u < \infty$ if $bc > 0$ and $2\lambda bc > 1$

$$u = \frac{b}{c} - \frac{1}{2\lambda c^2} = \frac{2\lambda bc - 1}{2\lambda c^2} \tag{B.24}$$

If instead $bc < 0$, we replace $|u|$ with $-u$ and seek a solution in $-\infty < u < 0$. By a similar process we obtain

$$u = \frac{b}{c} + \frac{1}{2\lambda c^2} = \frac{2\lambda bc + 1}{2\lambda c^2} \tag{B.25}$$

Thus, the minimum occurs at a differentiable point as long as $|2\lambda bc| \geqslant 1$.

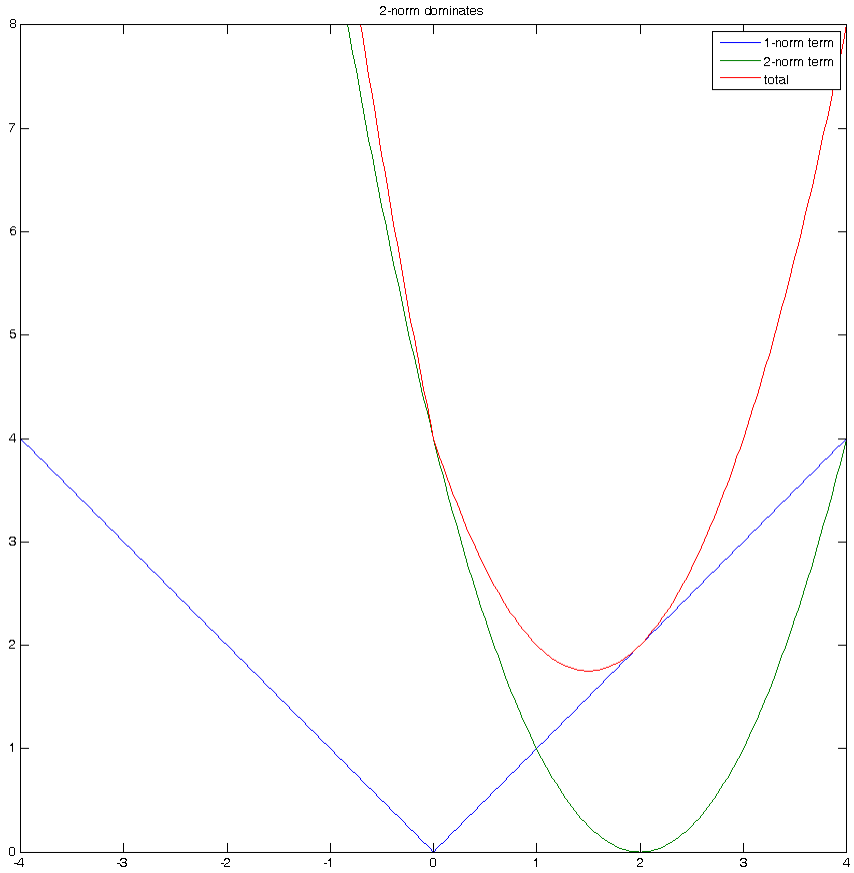


Figure B.3: When $|2\lambda bc| > 1$ and $bc > 0$, the Minimum Occurs in the Differentiable Region $0 < u < \infty$.

- **Case 2:** When $|2\lambda bc| < 1$ (B.23) is solved when $u = 0$. This is shown in Figure B.4.

The general solution to the Basis Pursuit problem comes from combining these cases and is given by

$$u = \begin{cases} \frac{b}{c} + \frac{1}{2\lambda c^2} & |2\lambda bc| \geqslant 1, bc < 0 \\ 0 & |2\lambda bc| < 1, bc > 0 \\ \frac{b}{c} - \frac{1}{2\lambda c^2} & |2\lambda bc| \geqslant 1, bc > 0 \end{cases} \tag{B.26}$$

We now define the shrink operator as

$$\text{shrink}\,(x, y) = \begin{cases} x + |y| & x < -|y| \\ 0 & |x| \leqslant |y| \\ x - |y| & x > |y| \end{cases} \tag{B.27}$$

Thus (B.26) can be expressed in terms of the shrink operator, (B.27), as

$$u = \frac{1}{c^2}\,\text{shrink}\left(bc, \frac{1}{2\lambda}\right) \tag{B.28}$$

We have now completed the motivation and numerical implementation procedure for solving (B.1) using the Split Bregman Algorithm.

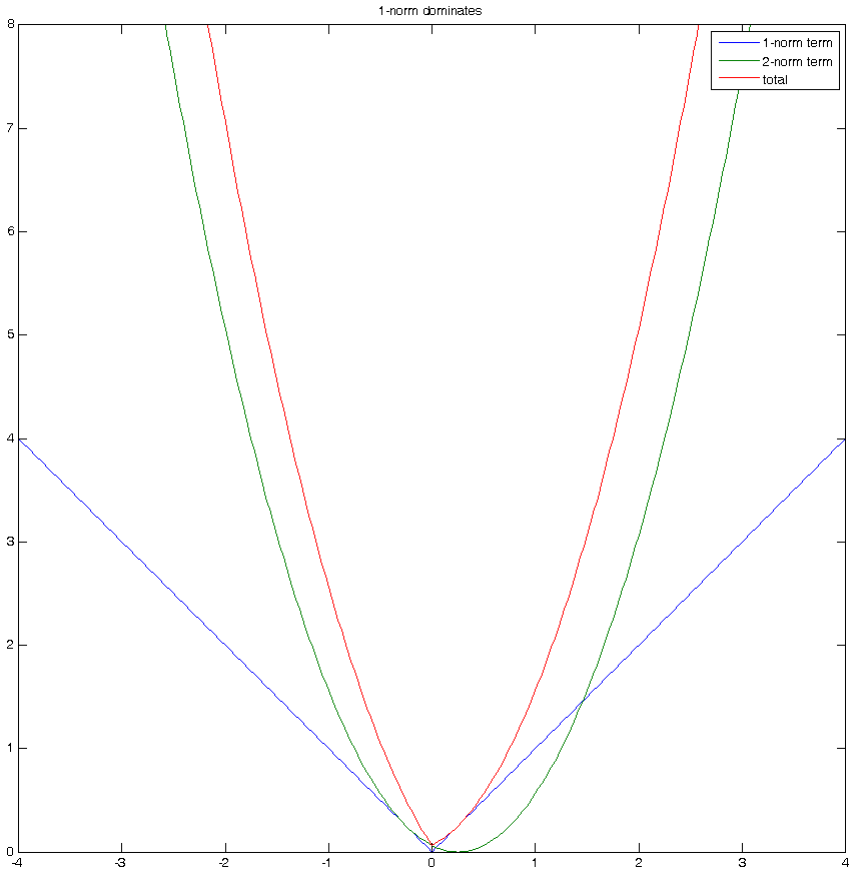


Figure B.4: When $|2\lambda bc| < 1$, the Minimum Occurs at a Point Lacking Differentiability.

APPENDIX C

PERMISSION TO USE PUBLISHED PAPERS

Chapter 3 of this dissertation was submitted for publication to the SIAM Journal on Scientific Computing in March 2016 under the title "Edge Detection of Piecewise Smooth Functions from Under-sampled Fourier Data". It was co-authored with Prof. Anne Gelb.

Chapter 4 of this dissertation has been published in Kirby *et al.* (2015), pages 157-166. It was co-authored with Dr. Rick Archibald and Prof. Anne Gelb.

I have obtained permission from the co-authors to include these papers in the dissertation.

Dennis Denker