University of California
Los Angeles

Applications of the Level Set Method to Geometrical Optics, Transmission Tomography, Image Processing and Crystal Growth Modeling

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Shing-Yu Leung

2006

The dissertation of Shing-Yu Leung is approved.

Jiun-Shyan Chen

Russel Caflisch

Andrea Bertozzi

Stanley Osher, Committee Chair

University of California, Los Angeles

2006

To my parents.

Table of Contents

1 Introduction **1**

2 A Level Set Based Eulerian Method for Paraxial Multivalued Traveltimes **3**

2.1 Introduction 3

2.2 Ray Tracing Formulation 6

2.2.1 Isotropic Paraxial Eikonal Equation 6

2.2.2 Anisotropic Paraxial Eikonal Equation 9

2.3 Level Set Formulation 11

2.3.1 Wavefronts 11

2.3.2 Traveltime 12

2.3.3 Amplitude 14

2.3.4 Caustics 15

2.4 Implementation 15

2.4.1 Boundary Conditions and an Algorithm 16

2.4.2 Regularization 18

2.4.3 Amplitude 21

2.4.4 Detecting Caustics 23

2.5 Numerical Experiments 24

2.5.1 Constant Model 24

2.5.2 Waveguide Model 28

2.5.3 Synthetic Marmousi Model 28
2.5.4 An anisotropic model 34

3 A Local Level Set Method for Paraxial Geometrical Optics . . 38

3.1 Introduction 38
3.2 Local Level Set Method 38
3.2.1 Implementation 39
3.2.2 Algorithm 40
3.3 Numerical Examples 42
3.3.1 Waveguide Model 43
3.3.2 Sinusoidal Model 46
3.3.3 Synthetic Marmousi Model 46

4 A Level Set Method for Three-dimensional Paraxial Geometrical Optics with Multiple Point Sources 50

4.1 Introduction 50
4.2 3D Paraxial Formulation for Eikonal Equation 51
4.3 Level Set Formulation 54
4.3.1 Representation of a Point Source 54
4.3.2 Traveltime 55
4.3.3 Representation of Multiple Sources 56
4.3.4 Amplitude 57
4.4 Numerical Method 58
4.4.1 Level Set Equations 58

4.4.2 Traveltime Equation 60
4.4.3 Multivalued Traveltimes 61
4.4.4 Reinitialization and Intersection 62
4.4.5 Amplitude 63
4.5 Numerical Examples 65
4.5.1 Waveguide Model 65
4.5.2 Vinje's Gaussian Model 69

5 Transmission Traveltime Tomography Using First Arrivals . . 76
5.1 Introduction 76
5.2 Governing Equations 78
5.3 Algorithm and Numerical Implementations 83
5.3.1 Tomography Algorithm 83
5.3.2 Fast Sweeping Method for Equation (5.1) 84
5.3.3 Fast Sweeping Method for Equation (5.7) 86
5.3.4 L-BFGS Method 88
5.4 Two-Dimensional Numerical Examples 89
5.4.1 Constant Model 90
5.4.2 Waveguide Model 90
5.4.3 Gaussian Model 92
5.5 Three-Dimensional Numerical Examples 93
5.5.1 Constant Model 96
5.5.2 Gaussian Model 96

5.6 Synthetic Marmousi Model 97

6 Transmission Traveltime Tomography Using Multiple Arrivals 102

6.1 Introduction 102

6.2 Tomography Based on Paraxial Liouville Equations 104

6.3 An Algorithm and Some Implementation Details 113

6.4 Practical Details 114

6.5 Examples 119

6.5.1 Constant Model 120

6.5.2 Waveguide Model 123

6.5.3 Gaussian Model 130

7 Global Minimization of the Active Contour Model with TV-Inpainting and Two-Phase Denoising 137

7.1 Introduction 137

7.2 Two Recent Developments 139

7.2.1 A Two-Phase Method to Remove Impulse-Type Noise . . . 139

7.2.2 Global Minimizer of the Active Contour Model 141

7.3 The New Energy 143

7.3.1 The Energy 143

7.3.2 The Link Between Active Contour for Segmentation, Denoising and TV-Inpainting 146

7.4 Numerical Method 148

7.5 Examples 149

7.5.1 Example 1 . . . 149
7.5.2 Example 2 . . . 151
7.5.3 Example 3 . . . 157

8 An Adaptive Level Set Method for Stefan Problems . . . 158
8.1 Introduction . . . 158
8.2 The Quasi-steady Stefan Problem . . . 159
8.3 Numerical Method . . . 161
8.3.1 Grid Adaptation . . . 161
8.3.2 Coordinate Transformation . . . 163
8.3.3 Temperature Extension . . . 164
8.3.4 Updating the Level Set Function . . . 167
8.3.5 Reinitialization . . . 167
8.3.6 A Predictor-Corrector Approach . . . 168
8.4 Numerical Examples . . . 170
8.4.1 Example 1 . . . 170
8.4.2 Example 2 . . . 170
8.4.3 Example 3 . . . 178

References . . . 179

List of Figures

2.1 (Constant Model) Evolution of the zero level set in the phase space at different z=0.0, 0.2, 0.4, 0.6, 0.8 and 1.0. 25

2.2 (Waveguide Model) Multivalued traveltimes by the level set method for z=0.8, 1.2 (upper row), 1.6 and 2.0 in the wave guide model. . 29

2.3 (Waveguide Model) The zero level set overlaid on contours of time field T in the wave guide model at different z=0.0, 0.4, 0.8, 1.2, 1.6 and 2.0. 30

2.4 (Marmousi Model) (Left) traveltime at z=0.0 by the level-set method for Marmousi model on 100×200 grid and (right) Eulerian traveltimes (solid line) vs. ray-tracing traveltimes (*) 32

2.5 (Marmousi Model) The zero level set for Marmousi model on 100×200 grid; the zero level set overlaying contours of T at z=0.0. 33

2.6 (Marmousi Model) (Left) traveltime at z=0.0 by the level-set method for Marmousi model on 400×200 grid and (right) Eulerian traveltimes (solid line) vs. ray-tracing traveltimes (star). 33

2.7 (Marmousi Model) The zero level set for Marmousi model on 400×200 grid; the zero level set overlaying contours of T at z=0.0. 34

2.8 (Anisotropic Model) Anisotropic paraxial multivalued traveltimes by the level set method. qP wave traveltime: the upper-left one; qSV traveltime: the upper-right one; qSH traveltime: the lower-left one; qP-qSV-qSH: the lower-right one. 37

3.1 (Waveguide Model) Traveltime, amplitude, Δ and ϕ_θ at $z = 1.6$km using a 240-by-240 grid. 44

3.2 (Waveguide Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a 360-by-360 grid in the x-θ space. 45

3.3 (Sinusoidal Model) Traveltime, amplitude, Δ and ϕ_θ at $z = 2.0$km using a 240-by-240 grid. 47

3.4 (Sinusoidal Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a 360-by-360 grid in the x-θ space. 48

3.5 (Marmousi Model) Contours of the traveltime and the zero level set at $z = 0.0$km. 48

3.6 (Marmousi Model) (Left) Comparison between traveltimes using a ray tracing method and the local level set method and (right) the Zoom-in of the local level set solution at Receiver 310 at z=0.0km. 49

4.1 The spherical coordinates and the rotated spherical coordinates . 52

4.2 Semi-Lagrangian method to solve the advection equations. 59

4.3 Determining the intersection of level set functions. 61

4.4 (Waveguide Model) Traveltimes in the physical space. 66

4.5 (Waveguide Model) Traveltimes in the physical space using ray tracing method. 66

4.6 (Waveguide Model) Traveltimes in the physical space on the cross section $y = 0$. Solution using ray tracing method is plotted using solid line. 68

4.7 (Waveguide Model) Traveltimes in the physical space on different cross sections. 68

4.8 (Waveguide Model) Amplitudes in the physical space. 69

4.9 (Waveguide Model) Amplitudes in the physical space on the cross section $y = 0$. 70

4.10 (Waveguide Model) Amplitudes in the physical space on different cross sections. 70

4.11 (Vinje's Gaussian Model) Traveltimes in the physical space. . . . 71

4.12 (Vinje's Gaussian Model) Traveltimes in the physical space using ray tracing method. 72

4.13 (Vinje's Gaussian Model) Traveltimes in the physical space. Solution using ray tracing method is plotted using solid line. 72

4.14 (Vinje's Gaussian Model) Traveltimes in the physical space on different cross sections. 73

4.15 (Vinje's Gaussian Model) Amplitudes in the physical space. . . . 74

4.16 (Vinje's Gaussian Model) Amplitudes in the physical space on the cross section $y = 0$. 74

4.17 (Vinje's Gaussian Model) Amplitudes in the physical space on different cross sections. 75

5.1 (Constant Model. Ten Sources.) BFGS. (a): the initial guess; (b): final approximated c; (c): the relative error in the solution; (d): the convergence history of energy. 91

5.2 (Waveguide Model. Ten Sources.) (a): the relative error in the solution and (b): the convergence history of energy. 92

5.3 (Waveguide Model. Ten Sources.) Cross-sections of the solutions. (a): $z = 1$ and (b): $x = 0$. 93

5.4 (Gaussian Model. Ten Sources.) BFGS. (a): the final approximated c and (b): the convergence history of energy. 94

5.5 (Gaussian Model. Ten Sources.) BFGS. Cross-sections of the solutions. (a): $z = 1$ and (b): $x = 0$. 94

5.6 (Gaussian Model with added noise. Ten Sources.) BFGS. (a): the final approximated c; (b): the convergence history of energy. . . . 95

5.7 (Gaussian Model with added noise. Ten Sources.) BFGS. Cross-sections of the solutions: (a): $z = 1$ and (b): $x = 0$. 95

5.8 (Constant Model. 98 Sources.) 3-D case. (a): the relative error in the solution on the cross-section $z = 1$ and (b): the convergence history of energy. 96

5.9 (Gaussian Model. 98 Sources.) 3-D case. (a): the relative error in the solution on the cross-section $z = 1$ and (b): the final approximated c. 97

5.10 (Marmousi model) (a): the true velocity distribution and (b): the initial profile c^0. 98

5.11 (Marmousi model) Converged solutions. (a): $\nu = 10^4$ and $\Delta x = 24$; (b): $\nu = 10^2$ and $\Delta x = 24$; (c): $\nu = 10^6$ and $\Delta x = 24$; (d): $\nu = 10^4$ and $\Delta x = 12$. 99

5.12 (Marmousi model) The change in (I): the energy and (II): the residual. Legend: (a): $\nu = 10^4$ and $\Delta x = 24$; (b): $\nu = 10^2$ and $\Delta x = 24$; (c): $\nu = 10^6$ and $\Delta x = 24$; (d): $\nu = 10^4$ and $\Delta x = 12$. . 100

6.1 (Constant Model) The initial guess and final approximated c, the error in the solution and the convergence history of energy in semi-log scale. 121

6.2 (Constant Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 122

6.3 (Constant Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively. 122

6.4 (Constant Model) First-arrival based traveltime tomography with a single source and multiple receivers: the final approximated c and the convergence history of energy in semi-log scale. 124

6.5 (Constant Model) First-arrival based traveltime tomography with a single source and multiple receivers: Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 124

6.6 (Constant Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution. 125

6.7 (Waveguide Model) The initial guess and final approximated c, the relative error in the solution and the convergence history of energy in semi-log scale. 126

6.8 (Waveguide Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 127

6.9 (Waveguide Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively. 127

6.10 (Waveguide Model) Energy histories with different regularization ν using the method of gradient descent with (a) $\epsilon = 0.01$ and (b) $\epsilon = 0.10$, and (c) varying ν using the strategy proposed in Section 6.2 and (d) the relative error in the solution using this strategy. . 128

6.11 (Waveguide Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution. 130

6.12 (Gaussian Model) The initial guess and final approximated c, the relative error in the solution and the convergence history of energy in semi-log scale. 131

6.13 (Gaussian Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 132

6.14 (Gaussian Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively. 132

6.15 (Gaussian Model) First-arrival based traveltime tomography with a single source and multiple receivers: the final approximated c and the convergence history of energy in semi-log scale. 134

6.16 (Gaussian Model) First-arrival based traveltime tomography with a single source and multiple receivers: cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 134

6.17 (Gaussian Model with Additive Gaussian Noise) The final approximated c and the convergence history of energy in semi-log scale. . 135

6.18 (Gaussian Model with Additive Gaussian Noise) Errors and relative errors in c^∞. 135

6.19 (Gaussian Model with Additive Gaussian Noise) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$. 136

6.20 (Gaussian Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution. 136

7.1 Segmentation results using the active contour model. We show different initial configurations of the snake on the first row. The corresponding segmented results using these initial conditions are shown on the second row. 141

7.2 Problem setting. Definition of the set Ω' (domain for inpainting), $\tilde{\Omega}'$ (compliment of Ω') and Ω_C (domain bounded by the curve C). 143

7.3 The original true image and the user defined mask. 150

7.4 The original image with 75% salt-and-pepper noise, 50% random-valued impulse noise and additive Gaussian noise ($\sigma = 20$) respectively. 150

7.5 (Salt-and-Pepper) The minimizer for the energy (7.12) without an extra mask. 151

7.6 (Salt-and-Pepper) The minimizer for the energy (7.12) with an extra mask. 152

7.7 (Random-valued Impulse) The minimizer for the energy (7.12) without an extra mask. 152

7.8 (Random-valued Impulse) The minimizer for the energy (7.12) with an extra mask. 153

7.9 (Additive Gaussian) The minimizer for the energy (7.12) without an extra mask. 153

7.10 (Additive Gaussian) The minimizer for the energy (7.12) with an extra mask. 154

7.11 The original true image and the user defined mask. 154

7.12 The original image with 75% salt-and-pepper noise, 50% random-valued impulse noise and additive Gaussian noise ($\sigma = 20$) respectively. 155

7.13 (Salt-and-Pepper) The minimizer for the energy (7.12) without (left) and with (right) an extra mask. 155

7.14 (Random-valued impulse) The minimizer for the energy (7.12) without (left) and with (right) an extra mask. 156

7.15 (Additive Gaussian) The minimizer for the energy (7.12) without (left) and with (right) an extra mask. 156

7.16 Some noisy brain MRI images and their corresponding denoised MRI images. 157

8.1 (Example 1) Evolution of the interface using 150-by-150, 200-by-200 and 400-by-400 uniform rectangular grids with two different orientations of the initial profile. 171

8.2 (Example 1) Evolution of the interface using 100-by-100 and 150-by-150 adaptive grids. The second row shows the grid points at the last time step. 172

8.3 (Example 1) Evolution of the interface using 100-by-100 and 150-by-150 adaptive grids with another orientation for the initial seed. The second row shows the grid points at the last time step. 173

8.4 (Example 2a) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step. 175

8.5 (Example 2a) 100-by-100 and 200-by-200 uniform rectangular grids.175

8.6 (Example 2b) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step. 176

8.7 (Example 2b) 100-by-100 and 200-by-200 uniform rectangular grids.176

8.8 (Example 2c) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step. 177

8.9 (Example 2c) 100-by-100, 150-by-150 and 200-by-200 uniform rectangular grids. 177

8.10 (Example 3) Four four-fold initial seeds with (left to right) $\sigma = 0.0001, 0.00025$ and 0.0005 in the Gibbs-Thomson condition. The corresponding grid points at the last time step are shown on the bottom. 178

List of Tables

2.1 Accuracy and convergence order of traveltimes without either reinitialization or orthogonalization 26

2.2 Accuracy and convergence order of traveltimes without orthogonalization but 2 reinitialization pseudo steps at each z-step using the approximation (2.52). 27

2.3 Accuracy and convergence order of traveltimes without reinitialization but 2 orthogonalization pseudo steps at each z-step using the approximation (2.52). 27

2.4 Accuracy and convergence order of traveltimes with 2 reinitialization pseudo steps and 2 orthogonalization pseudo steps at each z-step using the approximation (2.52). 27

5.1 Iteration count for the fast sweeping methods. The numbers in the brackets are the errors in the corresponding iteration, $||T^{n+1} - T^n||$ or $||\lambda^{n+1} - \lambda^n||$. 101

Acknowledgments

I wish to express my deepest appreciation to my advisor, Professor Stanley Osher, for his constant encouragements and supports throughout my years at UCLA. I thank Professor Jianliang Qian for his continuous encouragements, guidance and valuable suggestions over years. Further thanks go to Professor John Lowengrub at UCI for some valuable discussions on crystal growth modeling. I would also like to thank my committee members, Professor Andrea Bertozzi, Professor Russel Caflisch and Professor Jiun-Shyan Chen, for their valuable suggestions in my oral exam and in the final thesis defense.

I thank my parents and my brother for their continuous supports in my studies here at UCLA. I thank my friends and colleagues: Youri Bae, Eric Chung, Jason Chung, Lin He, Lennon Ho, Raphael Lee, Yan Li, Lok Ming Lui, John Mui, David Shao, Henry Siu, Nang Keung Sze, Sam Tse, Jinjun Xu and many more. All of you have helped me a great deal to make it through, so thank you.

I would also like to thank the Department of Mathematics at UCLA and the IPAM, the Institute of Pure & Applied Mathematics, for providing fantastic computing facilities and a very comfortable environment for my studies and research.

Chapter 2, 3, 5 and 6 are co-authored with Jianliang Qian and are versions of [86], [87], [67] and [66, 68], respectively. Chapter 4 is a version of [69] and is co-authored with Jianliang Qian and Stanley Osher. Chapter 7 is based on [65] and is co-authored with Stanley Osher. Chapters 2-6 were supported by the ONR Grant #N00014-02-1-0720. Chapter 7 was supported through the NSF Grant DMS-0312222 and the NIH Grant U54 RR021813.

Vita

1977	Born, Hong Kong, China.
1999	B.Sc. (Mathematics), Hong Kong University of Science and Technology, Hong Kong, China.
1999–2001	Teaching Assistant, Department of Mathematics, Hong Kong University of Science and Technology, Hong Kong, China.
2001	M.Phil. (Mathematics), Hong Kong University of Science and Technology, Hong Kong, China.
2001–2003	Teaching Assistant, Department of Mathematics, UCLA.
2003	M.A. (Mathematics), UCLA.
2004	C.Phil. (Mathematics), UCLA.
2003–present	Research Assistant, Department of Mathematics, UCLA.

Publications

Jianliang Qian and Shingyu Leung, *A Level Set Based Eulerian Method for Paraxial Multivalued Traveltimes*, Journal of Computational Physics, Volume 197, Issue 2, Pages 711-736.

Jianliang Qian and Shingyu Leung, *A Local Level Set Method for Paraxial Geometrical Optics*, SIAM. J. Sci. Comp., Volume 28, Issue 1, Pages 206-223.

Shingyu Leung, Jianliang Qian and Stanley Osher, *A Level Set Method for Three-dimensional Paraxial Geometrical Optics with Multiple Sources*, Commun. Math. Sci., Volume 2, Issue 4, December 2004, Pages 643-672.

Shingyu Leung and Stanley Osher, *Fast Global Minimization of the Active Contour Model with TV-Inpainting and Two-phase Denoising*, Proceeding of the 3rd IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision, 2005, Pages 149-160.

Shingyu Leung and Jianliang Qian, *A Transmission Tomography Problem Based on Multiple Arrivals from Paraxial Liouville Equations*, In Expanded Abstract for the SEG 75th Annual Meeting, Houston, USA, 2005.

Shingyu Leung and Jianliang Qian, *An Adjoint State Method for 3D Transmission Traveltime Tomography Using First Arrival*. Commun. Math. Sci., Volume 4, Number 1, March 2006. Pages 249-266.

Abstract of the Dissertation

Applications of the Level Set Method to Geometrical Optics, Transmission Tomography, Image Processing and Crystal Growth Modeling

by

Shing-Yu Leung

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2006

Professor Stanley Osher, Chair

The level set method is a successful numerical method. This dissertation focuses on its applications to geometrical optics, transmission tomography, image processing, and crystal growth modeling. In geometrical optics, we have proposed a level set framework to compute multivalued solutions of the paraxial eikonal equation. To further improve the computational efficiency and the memory requirement of this algorithm, we combine it with a local level set method and a semi-Lagrangian method. Using these tools, we then solve an inverse problem of geometrical optics, the so-called transmission traveltime tomography. Namely, we are inverting the velocity in a medium using possibly multivalued traveltime measurements. We have also studied two other applications of the level set method. In image processing, we have developed a variational method for image segmentation which determines the global minimizer of the active contour model. In crystal growth modeling, we have proposed an adaptive level set method. All these examples show that the level set method is an effective numerical method and can be easily applied to various fields in applied mathematics.

CHAPTER 1

Introduction

The level set method, invented by Osher and Sethian [81], is a numerical method originally proposed to capture a moving interface. The main idea is to represent the interface by a level set of a function embedded in a higher dimension. This implicit and Eulerian approach has been successfully generalized and applied to various areas in applied mathematics [80]. In this dissertation, we focus on applications to geometrical optics, transmission tomography, image processing, and crystal growth modeling.

In Chapter 2, we apply the level set method to compute multivalued solutions of the two dimensional paraxial eikonal equation. With N the number of grid points along each spatial direction, the complexity of this algorithm is $O(N^3 LogN)$, rather than $O(N^4)$ as seen in typical ray-tracing methods.

In Chapter 3 and Chapter 4, we improve both the computational efficiency and the memory requirement of the algorithm in Chapter 2 by two numerical schemes. We first apply a local level set method in Chapter 3. This reduces the computational complexity from $O(N^3 LogN)$ to $O(N^2 LogN)$. In Chapter 4, we first extend the formulation in Chapter 2 to three-dimensional paraxial geometrical optics. Then we introduce a global semi-Lagrangian method to solve all level set equations. This numerical method drops the computational memory requirement from $O(N^4)$ to $O(N^2)$. Although the whole algorithm has the computational complexity $O(MN^4)$, where M is the number of steps in the ODE

solver for the semi-Lagrangian scheme, it can handle up to N^2 multiple point sources simultaneously.

In Chapter 2 to Chapter 4, we assume one knows the velocity, or the slowness, of the medium in the computational domain. The problem then is to determine the multivalued traveltimes. In Chapter 5 and Chapter 6, we are solving the inverse of this problem. Namely, we are approximating the velocity in the domain using traveltimes measured by some receivers on the boundaries. In Chapter 5, we invert the velocity using only first arrivals which provides a computationally fast estimation of the velocity. To improve the quality of this approximation, we apply in Chapter 6 the techniques we developed in Chapter 2 to transmission traveltime tomography using multiple arrivals.

The last two chapters contain two other applications of the level set method. In Chapter 7, we study variational methods for image segmentation. One of the most well-known methods is the active contour model. In a recent paper by Bresson et al., a link between the active contour model and the variational denoising model of Rudin-Osher-Fatemi (ROF) was demonstrated. This relation provides a method to determine the global minimizer of the active contour model. In that chapter, we propose a variation of this method to determine the global minimizer of the active contour model in the case when there are missing regions in the observed image. This new approach unifies image denoising, image segmentation and image inpainting.

In Chapter 8, we develop an adaptive level set method to solve the quasi-steady state approximation of the Stefan problem. Since this adaptive approach is still based on a structured quadrilateral mesh, we can simply use any standard finite difference method for all computations. We also implement several numerical examples to demonstrate the computational efficiency of this approach.

CHAPTER 2

A Level Set Based Eulerian Method for Paraxial Multivalued Traveltimes

2.1 Introduction

Consider the linear acoustic wave equation. According to the Debye procedure, we first insert the high frequency asymptotic ansatz into the wave equation. Among all these terms in the ansatz, the most important one is the zeroth order term, the so-called geometrical optics term [61]. This geometrical optics term consists of two functions, a phase function satisfying the eikonal equation, a first-order nonlinear Hamilton-Jacobi equation, and an amplitude function satisfying a linear transport equation with the phase gradient as coefficients. Thus to construct the geometrical optics term, one first solves the eikonal equation for the phase function and then integrates the transport equation for amplitude afterwards.

Conventionally, both the eikonal equation and the amplitude equation were solved by the method of characteristics, a.k.a. the ray tracing method in the seismology and optics. Thus the method inherits the intrinsic shortcoming of the Lagrangian method, i.e. the non-uniform resolution of the solution in the desired computational domain [17]. Certainly it is possible to overcome this drawback by introducing interpolation and bookkeeping data structures [118]. However, these

procedures could be fairly complicated in practise.

In the late 80's, Vidale [115], and van Trier and Symes [114] introduced different direct discretizations of the eikonal equation based on finite difference schemes. These methods are Eulerian approaches which yield uniform resolutions of the solution in the computational domain. As pointed out in [114] however, these obtained solutions should be understood as the minimum phase or the first-arrival traveltime in the viscosity solution sense for Hamilton-Jacobi equations [70, 29]. Once the traveltime is obtained, the transport equation for amplitude might be integrated [116, 121, 106, 88]. But since the viscosity solution usually develops kinks, the gradient of the phase function is discontinuous. Therefore, the resulting solution for the transport equation has to be understood in the measure sense [40]. The difference between the ray tracing solution and the finite difference solution of the eikonal equation is that one is multiple-valued and the other single-valued [7, 37]. Although the viscosity concept and those related numerical methods provide a natural Eulerian framework for geometrical optics, the drawback is that it provides only first arrivals while some applications may require all traveltimes. Moreover, White [120] proved that there is a high probability for a so-called transmission caustic to occur in an inhomogeneous medium. Beyond transmission caustics, more than one ray pass over each point in space so that the phase is multivalued. Therefore, it is important to design an Eulerian method for multivalued solutions of the eikonal equation, or in general Hamilton-Jacobi equations.

In the last decade or so, there are a lot of effort in this direction: geometrical domain decomposition type methods [35, 5, 6], slowness matching method [104, 105], dynamic surface extension method [100, 93], segment projection method [38], level set method [79, 85, 57, 24, 86], phase space method [43], moment-

based methods [36, 48, 56, 49], and etc. See [7, 37] for up-to-date reviews of the above methods.

Among these schemes, one of the most successful methods is the level set method. It was first used to compute multivalued phases in the high frequency asymptotics for acoustic wave equations in [79]. Later it was extended to compute multivalued phases (*traveltimes*) in the high frequency asymptotics for anisotropic elastic wave equations in [85, 25], where the multivalued solutions for a class of steady Hamilton-Jacobi equations were computed. Recently it was extended to compute multivalued wavefronts and multivalued phases in the high frequency asymptotics for the Schrödinger equation in [24], where multivalued solutions for time dependent Hamilton-Jacobi equations were constructed in a general level set framework, and it was also extended to compute multivalued wavefront locations of Hamilton-Jacobi equations in [57].

In this chapter, we propose another level-set based Eulerian method for computing multivalued solutions of the paraxial eikonal equation in both isotropic and anisotropic metrics. This level-set framework provides a natural link from a Lagrangian formulation to an Eulerian formulation. We first derive the ray tracing equation using one of the spatial directions as the running parameter, which corresponds to the paraxial eikonal equation. This ray tracing equation is then embedded into a level-set motion equation to define a passive motion for level sets. The corresponding multivalued traveltime is obtained by solving another advection equation with a non-homogeneous source term.

2.2 Ray Tracing Formulation

2.2.1 Isotropic Paraxial Eikonal Equation

Consider the eikonal equation with a point source condition in an isotropic medium which occupies an open, bounded domain $\Omega \subset \mathbb{R}^2$. By isotropy here we mean the wave velocity has no directional dependence. The equation is as follows,

$$|\nabla_{\mathbf{X}} \tau(\mathbf{x}, \mathbf{x}_s)| = \frac{1}{c(\mathbf{x})}. \tag{2.1}$$

$$\lim_{\mathbf{X}\to\mathbf{x}_s} \frac{\tau(\mathbf{x},\, \mathbf{x}_s)}{\|\mathbf{x} - \mathbf{x}_s\|} = \frac{1}{c(\mathbf{x}_s)},\ \tau \geq 0, \tag{2.2}$$

where $\mathbf{x}_s$ is the given source point, $c \in C^1(\Omega)$ is the positive velocity. Here $\tau(\mathbf{x}, \mathbf{x}_s)$ denotes the time (*traveltime*) taken by a particle moving at velocity $c(\mathbf{x})$ to travel from the source point $\mathbf{x}_s$ to a target point $\mathbf{x} \in \Omega$. For $\mathbf{x} \neq \mathbf{x}_s$ near $\mathbf{x}_s$, τ is a differentiable function of both arguments and satisfies the *eikonal equation* (2.1). However, if the velocity $c(\mathbf{x})$ is inhomogeneous with spatial position $\mathbf{x}$ and $\mathbf{x}$ is sufficiently distant from $\mathbf{x}_s$, τ is generally a multivalued function of both variables. This implies that it is very likely to have cusps and caustics in the solution [120].

As mentioned above, the concept of viscosity solution can be used to extract a globally single valued solution for the eikonal equation [28]; this solution assigns to each point $\mathbf{x}$ the least (possibly many) traveltimes from $\mathbf{x}_s$ to $\mathbf{x}$. Relying on this concept, we may use finite difference schemes to stably and efficiently compute this least traveltime [30, 81, 82, 91, 88, 89]. Although the viscosity concept and these related numerical methods provide a natural Eulerian framework for geometrical optics, the drawback is that it provides only first-arrivals while some

applications may require all arrival times.

By the method of characteristics for the eikonal equation (2.1) with the point source condition (2.2), we have a ray tracing system,

$$\frac{dx}{dt} = c\sin\theta \tag{2.3}$$
$$\frac{dz}{dt} = c\cos\theta \tag{2.4}$$
$$\frac{d\theta}{dt} = \sin\theta\frac{\partial c}{\partial z} - \cos\theta\frac{\partial c}{\partial x} \tag{2.5}$$

with initial conditions

$$x|_{t=0} = x_s \tag{2.6}$$
$$z|_{t=0} = z_s \tag{2.7}$$
$$\theta|_{t=0} = \theta_s \tag{2.8}$$

where $\mathbf{x} = (x, z)$, $\mathbf{x}_s = (x_s, z_s)$ and θ_s varies from $-\pi$ to π. This is a multivalued Lagrangian formulation because even though the rays in the phase space (x, z, θ) may never intersect, the projected rays in the physical space (x, z) may intersect.

In some applications, for example in reflection seismics [26] or in quantum mechanics [84], the traveltimes of interest are carried by the so-called sub-horizontal rays [50, 103, 88], where sub-horizontal means *oriented in the positive z-direction.* A convenient characterization for sub-horizontal rays is that

$$\frac{dz}{dt} \geq c\cos\theta_{\max} > 0 \tag{2.9}$$

for some $0 < \theta_{\max} < \frac{\pi}{2}$. This inequality holds for rays making an angle θ with the vertical satisfying $|\theta| \leq \theta_{\max} < \frac{\pi}{2}$.

To be specific, consider $\Omega = \{(x,z) : x_{\min} \leq x \leq x_{\max}, 0 \leq z \leq z_{\max}\}$ and assume that the source is located on the surface: $x_{\min} \leq x_s \leq x_{\max}$ and $z_s = 0$. Using the sub-horizontal condition, we can use depth z as the running parameter so that we have a reduced system

$$\frac{dx}{dz} = \tan\theta \tag{2.10}$$

$$\frac{d\theta}{dz} = \frac{1}{c}\left(\frac{\partial c}{\partial z}\tan\theta - \frac{\partial c}{\partial x}\right) \tag{2.11}$$

with

$$x|_{z=0} = x_s \tag{2.12}$$

$$\theta|_{z=0} = \theta_s \tag{2.13}$$

where now θ_s varies from $-\theta_{\max} \leq \theta \leq \theta_{\max} < \pi/2$. In addition, the traveltime can be computed by integrating

$$\frac{dt}{dz} = \frac{1}{c\cos\theta} \tag{2.14}$$

with

$$t|_{z=0} = 0. \tag{2.15}$$

This ray tracing system (2.10-2.13) is a multivalued Lagrangian formulation defined in the reduced phase space $(z; x, \theta)$. In fact, the same system can be directly obtained by applying the method of characteristics to the paraxial eikonal equa-

tion

$$\frac{\partial \tau}{\partial z} = H\left(x, z, \frac{\partial \tau}{\partial x}\right) \quad = \quad \sqrt{\max\left(\frac{1}{c^2} - \left(\frac{\partial \tau}{\partial x}\right)^2, \frac{\cos^2 \theta_{\max}}{c^2}\right)}. \qquad (2.16)$$

A theoretical justification can be found in [105].

As we will see, since the ray tracing system (2.10-2.11) is formulated in a reduced phase space, we may use a 2-dimensional level set motion equation to move the initial curve deduced from the initial condition.

2.2.2 Anisotropic Paraxial Eikonal Equation

For a general anisotropic medium in which wave propagation velocity has both spatial and directional dependence, we may also formulate the paraxial eikonal equation by enforcing the sub-horizontal condition.

To illustrate the idea behind our approach, we consider only the two-dimensional anisotropic eikonal equation

$$F(x, z, p_1, p_3) \quad = \quad 0, \qquad (2.17)$$

where F is a function depending on the anisotropic medium under consideration, $p_1 = \tau_x$ and $p_3 = \tau_z$ components of the slowness vector $\nabla\tau$ with τ being the traveltime. Parameterize the slowness vector by

$$p_1 = \frac{\sin\theta}{V(x, z, \theta)} \quad , \quad p_3 = \frac{\cos\theta}{V(x, z, \theta)}, \qquad (2.18)$$

where θ is known as the phase angle, varying from $-\pi$ to π, and V as the phase velocity solving an eigenvalue problem [85] and selecting different wave modes.

Applying the method of characteristics to equation (2.17), we have

$$\frac{dx}{dt} = \left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\frac{\partial F}{\partial p_1}, \tag{2.19}$$

$$\frac{dz}{dt} = \left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\frac{\partial F}{\partial p_3}, \tag{2.20}$$

$$\frac{dp_1}{dt} = -\left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\frac{\partial F}{\partial x}, \tag{2.21}$$

$$\frac{dp_3}{dt} = -\left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\frac{\partial F}{\partial z}, \tag{2.22}$$

where the normalization is made so that the evolution parameter t has the dimension of time and is identical to τ. To obtain an equation for $\frac{d\theta}{dt}$, we differentiate both equations in (2.18),

$$\frac{dp_1}{dt} = \frac{V\cos\theta - \frac{\partial V}{\partial\theta}\sin\theta}{V^2}\frac{d\theta}{dt} - \frac{\sin\theta}{V^2}\left(\frac{\partial V}{\partial x}\frac{dx}{dt} + \frac{\partial V}{\partial z}\frac{dz}{dt}\right), \tag{2.23}$$

$$\frac{dp_3}{dt} = \frac{-V\sin\theta - \frac{\partial V}{\partial\theta}\cos\theta}{V^2}\frac{d\theta}{dt} - \frac{\cos\theta}{V^2}\left(\frac{\partial V}{\partial x}\frac{dx}{dt} + \frac{\partial V}{\partial z}\frac{dz}{dt}\right). \tag{2.24}$$

Thus, solving the above equations for $\frac{d\theta}{dt}$ and substituting in (2.21) and (2.22), we have

$$\frac{d\theta}{dt} = \left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\left(V\frac{\partial F}{\partial x}\cos\theta - V\frac{\partial F}{\partial z}\sin\theta\right). \tag{2.25}$$

Equations (2.19), (2.20) and (2.25) give us the ray tracing system which may be solved with suitable initial conditions as (2.6), (2.7) and (2.8).

The condition for sub-horizontal rays is

$$\frac{dz}{dt} = \left(p_1\frac{\partial F}{\partial p_1} + p_3\frac{\partial F}{\partial p_3}\right)^{-1}\frac{\partial F}{\partial p_3} > 0, \tag{2.26}$$

which may be easily enforced in the reduced phase space for different wave modes. This implies that we may use the depth variable as the running parameter along the ray so that we have

$$\frac{dx}{dz} = \frac{\partial F}{\partial p_1}\left(\frac{\partial F}{\partial p_3}\right)^{-1} \tag{2.27}$$

$$\frac{d\theta}{dz} = \left(\frac{\partial F}{\partial p_3}\right)^{-1}\left(V\frac{\partial F}{\partial x}\cos\theta - V\frac{\partial F}{\partial z}\sin\theta\right), \tag{2.28}$$

augmented with initial conditions (2.12) and (2.13).

Similar to the isotropic case, the traveltime $T = t$ is computed by integrating

$$\frac{dT}{dz} = p_3 + p_1\frac{dx}{dz}, \tag{2.29}$$

with the initial condition (2.15).

2.3 Level Set Formulation

2.3.1 Wavefronts

As we mentioned above, we treat z as an artificial time variable. Now, if we define $\phi = \phi(z, x, \theta)$ such that the zero level set, $\{(x(z), \theta(z)) : \phi(z, x(z), \theta(z)) = 0\}$, gives the location of the reduced bicharacteristic strip $(x(z), \theta(z))$ at z, then we may differentiate the zero level set equation with respect to z to obtain

$$\phi_z + u\phi_x + v\phi_\theta = 0\,, \tag{2.30}$$

with

$$u = \frac{dx}{dz} \text{ and } v = \frac{d\theta}{dz}\,, \tag{2.31}$$

which are given by the ray equations (2.10-2.11) or (2.27-2.28). In essence, we embed the ray tracing equations as the velocity field, $\mathbf{u} = (u, v)$, into the level set equation which governs the motion of the bicharacteristic strips in the phase space.

The initial condition for the level set motion equation (2.30) is taken to be

$$\phi|_{z=0} = \phi(0, x, \theta) = x - x_s \,, \tag{2.32}$$

which is obtained from initial conditions (2.12) and (2.13). This is a signed distance function, satisfying $|\nabla_{x,\theta}\phi| = 1$, to the initial phase space curve

$$\{(x, \theta) : x = x_s, -\theta_{\max} \leq \theta \leq \theta_{\max}\} \tag{2.33}$$

in the reduced phase space $\Omega_p = \{(x, \theta) : x_{\min} \leq x \leq x_{\max}, -\theta_{\max} \leq \theta \leq \theta_{\max}\}$. The initial curve partitions Ω_p into two sub-domains represented by $\{(x, \theta) : \phi(0, \cdot, \cdot) < 0\}$ and $\{(x, \theta) : \phi(0, \cdot, \cdot) > 0\}$. Afterwards, the level set motion equation takes over and moves this initial curve as z varies. Since the initial curve defines an implicit function between x and θ, where θ is a multivalued function of x, the new curve shares the same property. Therefore, for a fixed z, for some x^*'s we may have more than one θ^* such that $\phi(z, x^*, \theta^*) = 0$. This essentially tells us where the solutions are multivalued.

2.3.2 Traveltime

To determine the traveltime of the ray from the above level set equation, we now derive a corresponding equation governing the evolution of traveltime. By the sub-horizontal condition in the paraxial formulation and the ray equation (2.14) or (2.29), let $F_{\mathbf{u}}(z; x, \theta)$ be the flow generated by the velocity field $\mathbf{u} = (u, v)$ in

the phase space (x, θ) along the z-direction. Then we can write

$$\frac{dT}{dz}(z, F_{\mathbf{u}}(z; x, \theta)) = \frac{1}{c\cos\theta} \tag{2.34}$$

in the isotropic case and

$$\frac{dT}{dz}(z, F_{\mathbf{u}}(z; x, \theta)) = p_3 + p_1\frac{dx}{dz} \tag{2.35}$$

in the anisotropic case. Therefore, having $t = T(z, x, \theta)$, we get the following advection equation

$$\frac{dt}{dz} = \frac{dT}{dz} = T_z + uT_x + vT_\theta = \frac{1}{c\cos\theta} \tag{2.36}$$

for isotropic traveltime and

$$\frac{dt}{dz} = \frac{dT}{dz} = T_z + uT_x + vT_\theta = p_3 + p_1\frac{dx}{dz} \tag{2.37}$$

for anisotropic traveltime.

The initial condition for T is specified according to the initial condition (2.15):

$$T|_{z=0} = T(0, x, \theta) = 0\,, \tag{2.38}$$

which is consistent with the initial condition (2.32).

By solving the level set equation (2.30), we have the locations of wavefronts; by solving the traveltime equation (2.36) or (2.37), we have the traveltime values at the corresponding locations. Therefore, the multivalued traveltime at a specific location in the physical space can be obtained by first computing wavefront locations and then interpolating the traveltime at that location from gridded

traveltimes.

2.3.3 Amplitude

The amplitude of the ray can be computed according to the formula given in [44, 121, 88]:

$$\tilde{A}(x,z;x_s,z_s) = \frac{1}{2\pi}\sqrt{\frac{c}{2}}\sqrt{|\nabla\tilde{T}\times\nabla\tilde{\psi}|}\ , \tag{2.39}$$

where $\tilde{T}$ and $\tilde{\psi}$ are, respectively, the traveltime and the take-off angle of a ray reaching (x,z) from (x_s,z_s). Traveltimes and takeoff angles are well defined on each solution branch in the physical space (x,z). To compute this quantity in the reduced phase space, we consider T as the extension of $\tilde{T}$ to the phase space; furthermore, we may also extend $\tilde{\psi}$ and $\tilde{A}$ to ψ and A in the (z,x,θ) space, respectively. Since the takeoff angle is constant along a given ray in the phase space, we have

$$\psi_z + u\psi_x + v\psi_\theta = 0\,. \tag{2.40}$$

Moreover, we have

$$\begin{aligned}
\frac{\partial\tilde{T}}{\partial x} &= T_x + T_\theta\frac{\partial\theta}{\partial x}\,,\\
\frac{\partial\tilde{T}}{\partial z} &= T_z + T_\theta\frac{\partial\theta}{\partial z}\,,\\
\frac{\partial\tilde{\psi}}{\partial x} &= \psi_x + \psi_\theta\frac{\partial\theta}{\partial x}\,,\\
\frac{\partial\tilde{\psi}}{\partial z} &= \psi_x + \psi_\theta\frac{\partial\theta}{\partial z}\,.
\end{aligned} \tag{2.41}$$

It follows that

$$\left|\nabla\tilde{T}\times\nabla\tilde{\psi}\right| = \left|(T_x\psi_\theta - T_\theta\psi_x)\left(-v + u\frac{\partial\theta}{dx} + \frac{\partial\theta}{\partial z}\right) - \frac{1}{c\cos\theta}\left(\psi_x + \psi_\theta\frac{\partial\theta}{\partial x}\right)\right| . \tag{2.42}$$

Since $\phi(z, x, \theta(x, z)) = 0$ on the zero level set and equation (2.11), we have

$$\frac{\partial\theta}{\partial x} = -\frac{\partial\phi/\partial x}{\partial\phi/\partial\theta} \text{ and } \frac{\partial\theta}{\partial z} = v - u\frac{\partial\theta}{\partial x} . \tag{2.43}$$

Finally we have

$$A(z; x, \theta) = \frac{1}{2\pi}\sqrt{\frac{c}{2\cos\theta}}\sqrt{\left|\frac{\psi_x\phi_\theta - \psi_\theta\phi_x}{\phi_\theta}\right|} . \tag{2.44}$$

2.3.4 Caustics

Mathematically, caustic surfaces are envelops of the family of rays. In the geometrical optics, at a caustic the amplitude of the asymptotic expansion becomes infinite, so that the usual asymptotic expansion is no longer valid at caustics, and some special expansions have been introduced to construct wave fields near the caustics [72, 73, 15].

In the current level set formulation, the caustic curves correspond to

$$\{(z, x) : \phi(z, x, \theta(x, z)) = 0 \text{ and } \phi_\theta(z, x, \theta(x, z)) = 0\}.$$

2.4 Implementation

We will give full details on implementing the level set Eulerian method for isotropic eikonal equations only.

2.4.1 Boundary Conditions and an Algorithm

We impose a non-reflective boundary condition for the level set equation, $\partial\phi/\partial\mathbf{n} = 0$. This ensures the information outside the domain Ω_p will not interfere with the zero level set inside the computational domain.

For the boundary conditions for the traveltime equation, to take care of the upwind property of the equation, we will split the boundaries into two types, one with information going into the domain and one with no information coming from outside the domain. The second type can be easily treated using non-reflective boundary condition. For the first type, the boundary conditions for the traveltime equation are determined using the local information from the characteristics system. We first locally invert equation (2.3) and (2.5) and get

$$\frac{\partial T}{\partial x} = \frac{1}{c\sin\theta}, \tag{2.45}$$

which is used for boundaries $x = x_{\min}$ and $x = x_{\max}$, and

$$\frac{\partial T}{\partial \theta} = \left(\sin\theta\frac{\partial c}{\partial z} - \cos\theta\frac{\partial c}{\partial x}\right)^{-1}, \tag{2.46}$$

which is used for boundaries $\theta = -\theta_{\max}$ and $\theta = \theta_{\max}$.

The above conditions essentially specify the normal derivatives of traveltime along the boundaries. Then the values of T on the boundaries will be obtained by applying the Adams' Extrapolation formula to equation (2.45), where θ is considered as fixed, and to equation (2.46), where x is considered as fixed.

In equation (2.45), it seems that there is a singularity when $\theta = 0$. However, in that case, $u = \tan\theta = 0$ and the information on the boundary will not pass into the computational domain and this case can actually be handled by the

non-reflective boundary condition too.

Summarizing all above ingredients, we propose determining the multivalued traveltimes for all x at some depth z^* using the following algorithm.

Algorithm 1:

I. Solve the level set equation (2.30) and the traveltime equation (2.36) up to z^* with the velocity field generated by the ray equations (2.10), (2.11).

II. For all x,

   i. determine **all** θ_i such that $\phi(z^*, x, \theta_i) = 0$ $(i = 1, \cdots)$ by root finding;

   ii. determine $T(z^*, x, \theta_i)$ $(i = 1, \cdots)$ by interpolation.

In Step I, the level set equation and the traveltime equation are decoupled and can be solved separately. The spacial derivatives are approximated by the fifth order WENO-Godunov scheme [55] while the time derivatives are solved by the third order TVD-RK method [82]. Both the level set equation (2.30) and the traveltime equation (2.36) are linear, hence the CFL step Δz can be chosen by

$$\Delta z \;\leq\; C\frac{\min(\Delta x, \Delta\theta)}{\max(\sqrt{u^2+v^2})}\,, \tag{2.47}$$

where Δx and $\Delta\theta$ are mesh sizes along the x- and θ-direction respectively, and C is the CFL number taken to be 0.6. For the root-finding and the interpolation in Step II, we can simply use any non-oscillatory interpolation scheme like linear interpolation or ENO reconstruction. For z^* fixed, the root finding is conducted by checking the values of ϕ on each line of x and θ varying. Interpolation for traveltime is performed where a root is present by the Intermediate Value Theorem.

2.4.2 Regularization

Initially at $z = 0$, we have a signed distance function satisfying $|\nabla\phi| = 1$, so that the level sets of ϕ are equally spaced. However, as z varies the level set function is no longer equally spaced in general. This implies that ϕ may develop steep or flat gradients near the zero level set, making the computed curve locations and further computations inaccurate, which does happen in Algorithm 1. Therefore, we propose the following regularization procedure which consists of reinitialization and orthogonalization.

To restore the equally spaced property for the level sets, the usual way is to make ϕ a signed distance function without moving the zero level set of ϕ appreciably. This can be achieved through the so-called reinitialization. The most-used way is to solve the following equation to steady state $\tilde{\phi}_\infty$ [102, 52, 83, 80]:

$$\begin{aligned} \frac{\partial\tilde{\phi}}{\partial\xi} + S(\phi)(|\nabla\tilde{\phi}| - 1) &= 0 && (2.48)\\ \tilde{\phi}|_{\xi=0} &= \phi(z,\cdot,\cdot) && (2.49)\\ \left.\frac{\partial\tilde{\phi}}{\partial\mathbf{n}}\right|_{\partial\Omega_p} &= 0\,, && (2.50) \end{aligned}$$

where $S(\phi)$ is a smoothed signum function which can be approximated by

$$S(\phi) = \frac{\phi}{\sqrt{\phi^2 + \Delta x\Delta\theta}}. \qquad (2.51)$$

Other choice of approximation is also possible, for example

$$S(\phi) = \frac{2}{\pi}\tan^{-1}(\phi)\,. \qquad (2.52)$$

A general principles of picking the function is, $S(\phi)$ is a bounded function with the following properties

1. $xS(x) > 0$ for all $x \neq 0$.

2. $S(0) = 0$.

The steady state $\tilde{\phi}_\infty$ has the same zero level set as $\phi(z, \cdot, \cdot)$ within a certain accuracy since $\tilde{\phi}$ does not move on the zero level set of ϕ. Moreover, at the steady state $\tilde{\phi}_\infty$ is a signed distance function since $|\nabla\tilde{\phi}_\infty| = 1$. The reinitialization step is to use $\tilde{\phi}_\infty$ instead of $\phi(z, \cdot, \cdot)$ as the initial condition at z for solving the level set equation to the next stage. Because we are interested only in the zero level set, it is necessary to evolve equation (2.48) for only a few pseudo-time steps. How often we should invoke the reinitialization step is a subtle issue; see [83, 80] for some discussions. In our implementation, we invoke the reinitialization at every z step so that we have a better-behaved function for determining the values θ_i in Step II of Algorithm 1.

Even with careful implementation of the above reinitialization procedure, the location of the zero level set may still be shifted by an amount less than one grid cell. This is harmless for the visualization purpose of the location of the ray. However, because the solution from the traveltime equation (2.36) would typically vary a lot near the corresponding location of the zero level set of ϕ, this shift makes the results from the interpolation in Step II highly inaccurate.

Because we are interested only in the value of T where $\phi = 0$, we propose the following orthogonalization procedure

$$\frac{\partial\tilde{T}}{\partial\xi} + \mathrm{sgn}(\phi)\left(\frac{\nabla\phi}{|\nabla\phi|}\cdot\nabla\tilde{T}\right) = 0\,, \tag{2.53}$$

$$\tilde{T}|_{\xi=0} = T(z, \cdot, \cdot) \tag{2.54}$$

$$\left.\frac{\partial \tilde{T}}{\partial \mathbf{n}}\right|_{\partial\Omega_p} = 0\,, \tag{2.55}$$

which, theoretically, preserves the values of T on where $\phi = 0$ but changes them elsewhere such that the new T would not vary too much near the desired region. At the steady state, we have $\nabla\phi\cdot\nabla\tilde{T} = 0$. Equation (2.53) may also be viewed as an extension procedure; namely, we extend the values of T on the zero level set of ϕ along the normal direction of the zero level set of ϕ; see [83, 80]. This generally makes T discontinuous since lines normal to the zero level set will eventually intersect somewhere away from the zero level set. Even if the location of the zero level set may be shifted, the effect to the interpolation will still be acceptable.

Similar to the reinitialization in ϕ, we only need to apply several iterations, instead of solving it until reaching the steady state solution. This makes the regularization procedure efficient, simple to implement and robust.

Incorporating those regularization into Algorithm 1, we have an improved algorithm.

Algorithm 2:

I. Initialization: given N_z, N_x and N_θ: $\Delta z = z_{\max}/(N_z - 1)$, $\Delta x = (x_{\max} - x_{\min})/(N_z - 1)$ and $\Delta\theta = 2\theta_{\max}/(N_\theta - 1)$; initialize ϕ and T at $z = 0$.

II. For $k = 1$ to N_z:

   1. March one Δz step from $(k-1)\Delta z$ to $k\Delta z$ by solving the level set equation (2.30) and reinitializing the level set motion by solving (2.48) at every intermediate z-step.

   2. March one Δz step from $(k-1)\Delta z$ to $k\Delta z$ by solving the traveltime equation (2.36).

3. Orthogonalize T and ϕ by solving equation (2.53).

4. For $x = (j-1)\Delta x, j = 1, \cdots, N_x$,

   i. determine **all** θ_i such that $\phi(k\Delta z, x, \theta_i) = 0$ $(i = 1, \cdots)$ by root finding;

   ii. determine $T(k\Delta z, x, \theta_i)$ $(i = 1, \cdots)$ by interpolation.

Since the reinitialization procedure is usually invoked for a fixed number of steps (from 1 to 4 pseudo-steps in our numerical examples presented below), the above algorithm in the average case has the complexity $O(N^3)$ where we assume $N_z = N_x = N_\theta = N$. In the worst case, if the reinitialization procedure is invoked until convergence, the above algorithm has the complexity $O(N^3 LogN)$.

2.4.3 Amplitude

To compute the amplitude, we need the derivatives of the level set function at different z's. However, as discussed above, we have regularized the level set function using reinitialization and unfortunately, this process would theoretically fix the location of the zero level set but alter all other level sets. This means the derivatives of ϕ cannot be computed directly from the level set function by differentiating the function ϕ itself. Therefore, to compute the derivatives of the level set function on the zero level set, we need to advect those derivatives as well. We first let $\xi = \phi_x$ and $\eta = \phi_\theta$. Differentiating the advection equation for ϕ with respect to x and θ respectively, we have

$$\begin{aligned} \xi_z + u\xi_x + v\xi_\theta + u_x\xi + v_x\eta &= 0\,, \\ \eta_z + u\eta_x + v\eta_\theta + u_\theta\xi + v_\theta\eta &= 0\,. \end{aligned} \tag{2.56}$$

We might apply the same idea to the advection equation for takeoff angles

as well so that the derivatives of takeoff angles could be determined. Once those ingredients are in place, the amplitude could be obtained. However, we will not use this approach because it is not computationally efficient and also numerically difficult to reconcile the accuracies of four different derivatives.

Instead, defining

$$\Delta = \psi_x \phi_\theta - \psi_\theta \phi_x \tag{2.57}$$

and differentiating it with respect to z, we have the advection equation

$$\Delta_z + \nabla_{x,\theta} \cdot (\mathbf{u}\Delta) = 0\,, \tag{2.58}$$

and the amplitude can then be computed by

$$A(z; x, \theta) = \frac{1}{2\pi} \sqrt{\left| \frac{c}{2\cos\theta} \frac{\Delta}{\phi_\theta} \right|}\,. \tag{2.59}$$

Therefore in order to get the amplitude, we can simply solve the advection equations for ϕ_x, ϕ_θ and Δ. We need the advection equation for ϕ_x because it is coupled with the one for ϕ_θ. In general, Δ is a bounded quantity and ϕ_θ may approach zero. When ϕ_θ goes to zero, A goes to infinity and we are approaching caustics.

Next we have to initialize all quantities that we are going to advect. At $z = 0$, we can set ϕ_x and ϕ_θ equal to 1 and 0 respectively. However, ψ_x is a delta-type like function at the source and it is better to start computing ψ_x and ψ_θ at some $z = dz > 0$ close to zero. Assuming that the velocity c can be approximated by a constant near the source, we have

$$\psi_x(z, x, \theta)|_{z=dz} \;=\; \frac{\cos^2\theta}{dz}\,,$$

$$\psi_\theta(z,x,\theta)|_{z=dz} \;=\; 1\,. \tag{2.60}$$

Thus at small $z = dz > 0$, $\phi(z,x,\theta)|_{z=dz} = x - dz\tan\theta$ and $\Delta(dz,x,\theta) \equiv -2$, which is independent of the dz as long as the velocity can be well approximated by a constant near the source. This makes the computation of amplitude stable.

2.4.4 Detecting Caustics

Because passing through a caustic implies overturning of the zero level set in the x-θ space, the number of θ's such that $\phi(z,x,\theta(x)) = 0$ will increase or decrease by two when x varies monotonically. Therefore, a simple way to detect caustics is first enumerating the number of roots θ_k's for every x_i, then checking where those numbers have sudden jumps, and finally approximating locations of caustics by taking the mid-point of two adjacent x_i's which have different number of θ_k. The resulting approximation of the caustics detection would be in first order. We let x^* be the exact location of a caustic at some fixed z^*. Assuming we can compute exactly, let x_n be the location of the caustic using our level set formulation in a grid level n. Let x'_n be the location computed using our mid-point approximation. We have at least $x_n = x^* + O(\Delta x^2)$. Because the distance between x_n and x'_n would be less than $\Delta x/2$, we have $x'_n = x_n + O(\Delta x)$ and, therefore, $x'_n = x^* + O(\Delta x)$.

In terms of multivalued traveltimes, passing through a caustic implies that the number of traveltimes increases or decreases by two as we will see in numerical examples.

2.5 Numerical Experiments

For the first two examples here, we put a point source at the origin and the velocity functions $c(x, z)$ are all C^∞. The third example, the synthetic Marmousi model, is a more challenging one where the velocity function is given only as a sampled function.

In all the examples the computational domain is chosen to be

$$\Omega_p \quad = \quad \{(x, \theta) : -1 \leq x \leq 1, -\frac{9\pi}{20} \leq \theta \leq \frac{9\pi}{20}\}. \tag{2.61}$$

Accordingly, the Marmousi velocity will be rescaled to the above computational domain. The last example is an anisotropic model which consists of three different wave modes; one of the wave modes has the so-called instantaneous singularity.

2.5.1 Constant Model

When the velocity c is constant, the analytic solution for the traveltime is known so that we can study the accuracy and the convergence order of the proposed Eulerian method. We compute the traveltime up to z=1.0 with different options of regularization procedures to see how reinitialization and orthogonalization affect the accuracy and the convergence order of the method.

Figure 2.1 shows the evolution of the zero level set as the depth z increases. Noticed that initially we have a vertical line. Under the influence of the velocity field $\mathbf{u} = (\tan\theta, 0)$, the upper part of the vertical line is advected to the right, and the lower part of the line is advected to the left. However, since this traveltime is single valued in this case, the zero level set always defines a single valued implicit function between x and θ for every fixed z.

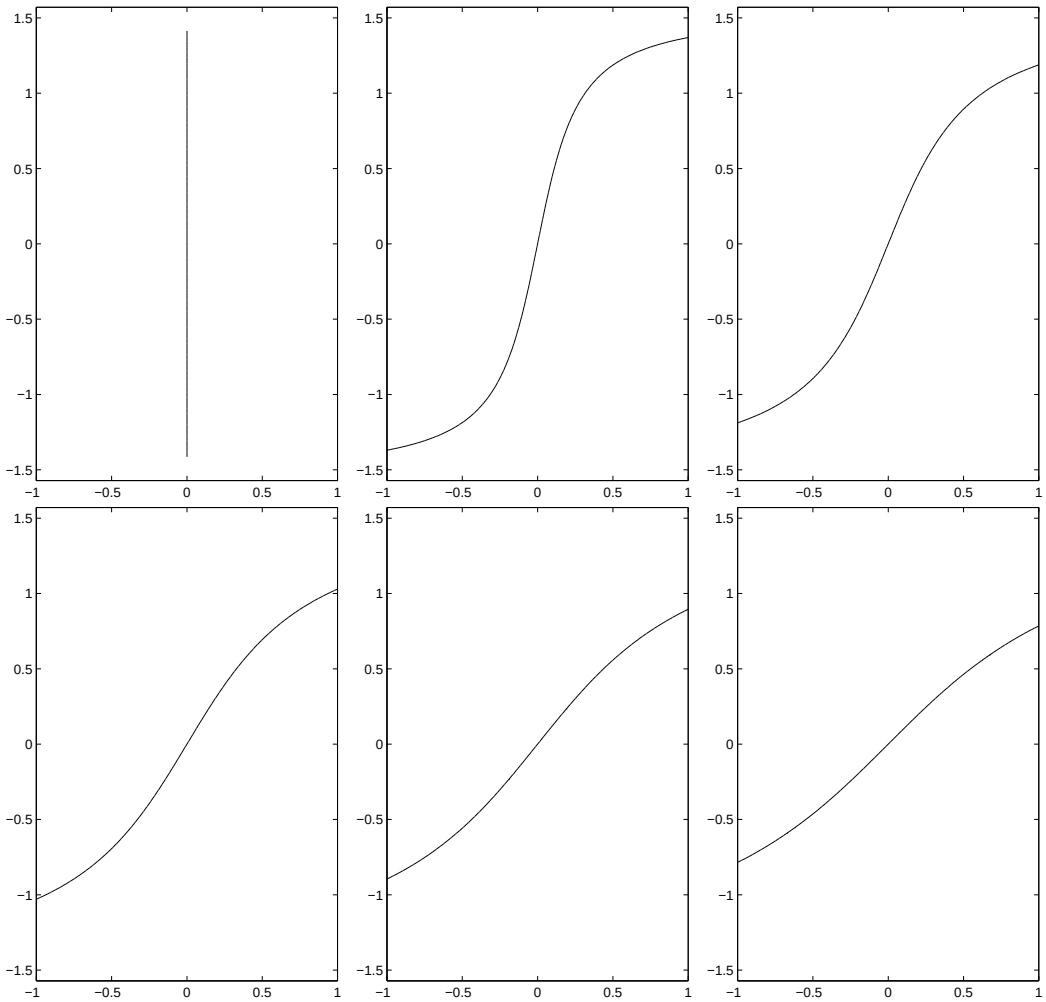


Figure 2.1: (Constant Model) Evolution of the zero level set in the phase space at different z=0.0, 0.2, 0.4, 0.6, 0.8 and 1.0.

Δx	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
0.20000	0.01374575		0.01153032		0.01213886	
0.10000	0.00366580	1.9067	0.00283586	2.0235	0.00265964	2.1903
0.05000	0.00092272	1.9901	0.00073505	1.9478	0.00079357	1.7447
0.02500	0.00024266	1.9269	0.00018498	1.9904	0.00021538	1.8814
0.01250	0.00005863	2.0491	0.00004600	2.0074	0.00005231	2.0416
0.00625	0.00001497	1.9692	0.00001165	1.9814	0.00001369	1.9336

Table 2.1: Accuracy and convergence order of traveltimes without either reinitialization or orthogonalization

Table 2.1 shows the clean second-order accuracy and convergence of traveltimes in l_1, l_2 and l_∞ norms without either reinitialization or orthogonalization. This is expected because the linear interpolation is used to extract traveltimes gives us second-order accuracy only, even though the level set equation and the traveltime equation are solved to third order accuracy. Table 2.2 also shows second order convergence of traveltimes in different norms without orthogonalization but with two reinitialization pseudo steps at each z-step. This shows that the reinitialization procedure alone does not move the zero level set too much so that the accuracy is not affected appreciably. Table 2.3 shows second order convergence of traveltimes in different norms without reinitialization but with two orthogonalization pseudo steps at each z-step. This shows that the orthogonalization procedure improves the behavior of the traveltime field near the zero level set so that the traveltime accuracy is enhanced greatly. Table 2.4 shows second order convergence of traveltimes in different norms when both reinitialization and orthogonalization are invoked. This indicates that combining reinitialization and orthogonalization procedures together does enhance the algorithmic behavior and improve the accuracy of computed traveltimes significantly.

Δx	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
0.20000	0.02029779		0.01874370		0.02519635	
0.10000	0.00532929	1.9293	0.00430244	2.1231	0.00484257	2.3793
0.05000	0.00135341	1.9773	0.00116012	1.8908	0.00173133	1.4838
0.02500	0.00035766	1.9199	0.00029215	1.9894	0.00050259	1.7844
0.01250	0.00008633	2.0506	0.00007299	2.0008	0.00011961	2.0710
0.00625	0.00002233	1.9506	0.00001873	1.9617	0.00003397	1.8159

Table 2.2: Accuracy and convergence order of traveltimes without orthogonalization but 2 reinitialization pseudo steps at each z-step using the approximation (2.52).

Δx	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
0.20000	0.00727231		0.00770262		0.01124069	
0.10000	0.00143178	2.3445	0.00131120	2.5544	0.00181632	2.6296
0.05000	0.00040647	1.8165	0.00038993	1.7495	0.00065443	1.4726
0.02500	0.00010576	1.9422	0.00009684	2.0095	0.00020036	1.7076
0.01250	0.00002598	2.0252	0.00002437	1.9899	0.00004782	2.0667
0.00625	0.00000671	1.9517	0.00000633	1.9446	0.00001484	1.6880

Table 2.3: Accuracy and convergence order of traveltimes without reinitialization but 2 orthogonalization pseudo steps at each z-step using the approximation (2.52).

Δx	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
0.20000	0.00393867		0.00383178		0.00440058	
0.10000	0.00109580	1.8457	0.00097230	1.9785	0.00122641	1.8432
0.05000	0.00027266	2.0067	0.00024258	2.0029	0.00032516	1.9152
0.02500	0.00007131	1.9349	0.00006096	1.9924	0.00008355	1.9603
0.01250	0.00001745	2.0304	0.00001520	2.0031	0.00002115	1.9814
0.00625	0.00000434	2.0053	0.00000380	1.9996	0.00000530	1.9955

Table 2.4: Accuracy and convergence order of traveltimes with 2 reinitialization pseudo steps and 2 orthogonalization pseudo steps at each z-step using the approximation (2.52).

2.5.2 Waveguide Model

The velocity function is

$$c(x,z) \;=\; 1.1 - \exp(-0.5x^2)\,. \tag{2.62}$$

The function is symmetric with respect to $x = 0$, and we also expect the same type of symmetry in the traveltime.

Figure 2.2 shows the traveltime we obtained using only 40×40 grids in the x-θ space. The solutions here show the traveltimes at z equals 0.8, 1.2, 1.6 and 2.0. The solutions are symmetric as expected. Figure 2.3 shows the zero level set overlaying the traveltime field at z=0.0, 0.4, 0.8, 1.2, 1.6 and 2.0 respectively. The dashed line is the location of the zero level set and the solid lines are the contour plot of the traveltime function T. There are discontinuities in the traveltime field coming from the orthogonalization procedure, where the normals of the zero level set intersect. However, it is reminded that we only use the information near the dashed line and the jump in T will not hurt our interpolation computations if the grids are fine enough, meaning that the discontinuity is greater than one grid point away from the zero level set. Figure 2.3 also shows that the contours are perpendicular to the zero level set as designed. As z varies, the zero level set is advected so that it has more turnarounds and the number of traveltime arrivals increases from 1 to 3.

2.5.3 Synthetic Marmousi Model

This example is the Marmousi model from the 1996 INRIA Workshop on Multi-arrival Traveltimes. The calibration data used here were computed by Dr. Klimes and can be found at http://www.caam.rice.edu/∼benamou/traveltimes.html.

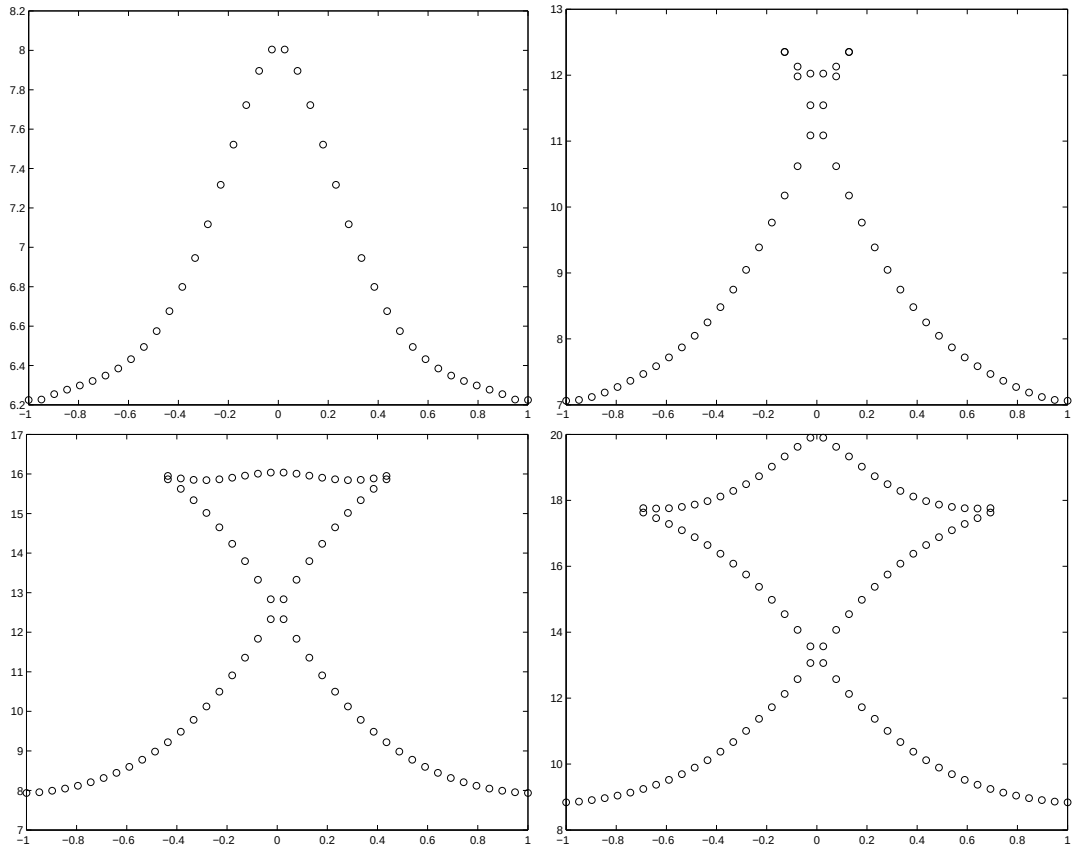


Figure 2.2: (Waveguide Model) Multivalued traveltimes by the level set method for z=0.8, 1.2 (upper row), 1.6 and 2.0 in the wave guide model.

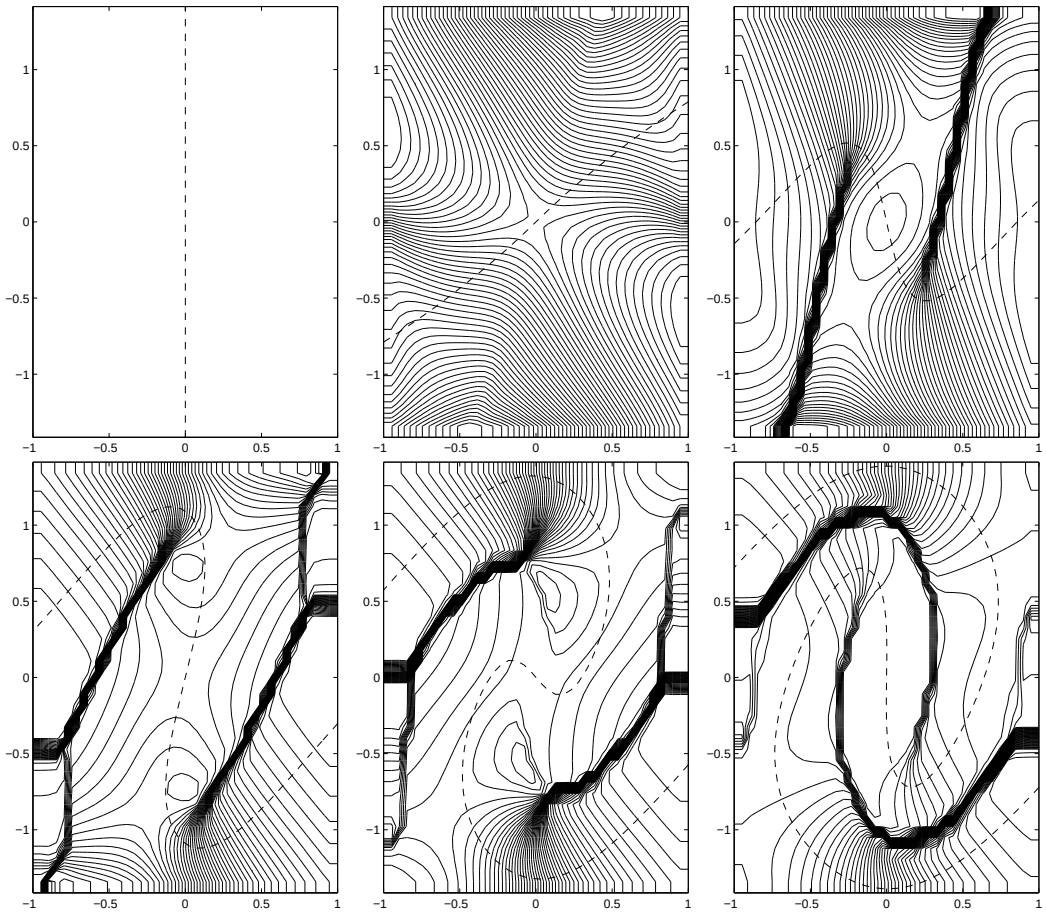


Figure 2.3: (Waveguide Model) The zero level set overlaid on contours of time field T in the wave guide model at different z=0.0, 0.4, 0.8, 1.2, 1.6 and 2.0.

The original Marmousi model is sampled on a 24m by 24m grid, consisting of 384 samples in the x-direction and 122 samples in the z-direction; therefore the model dimension is 9.192km long in the x-direction and 2.904km deep in the z-direction. In the computational results presented here, we use a portion of Marmousi model, i.e., a window from 4.8km to 7.2km in the x-direction and from 0km to 2.904km in the z-direction. The source is located at x=6.0km and z=2.8km. The purpose is to compute (possibly multivalued) traveltimes for those sampling points, i.e. the receivers from 200 to 300 on the surface z=0.0.

In the first run, we used a 100 by 200 grid in x-θ space with Δx=24m and $\theta_{\max} = 9\pi/20$. The computed traveltime at z=0.0 and the comparison with the ray tracing solution are shown in Figure 2.4. The ray tracing data used to calibrate the computed Eulerian solutions are presumably accurate. As we can see from Figure 2.4, the computed Eulerian solution is consistent with the ray tracing solution, being able to capture most of the structure of the multivalued solution, but it failed to resolve some fine details, especially the two traveltime branches located from receivers 260 to 280, where those two branches are very close to each other. Figure 2.5 explains why the level set method failed in that region. Figure 2.5 shows that the zero level set and its overlay on the traveltime field in the reduced phase space. From the ray tracing solution we know that receivers 260 to 280 should have three arrivals, but from the zero level set, receivers from 260 to 280 are single-valued functions of θ, and they correspond to first-arrival traveltimes. Apparently, the tip of the zero level set near receiver 260 should be more elongated, but somehow the level set failed to elongate that tip. This is partly due to the dissipation of the finite difference scheme used here and partly due to the resolution capability of the level set method which can resolve the zero level set up to only one grid-cell width. Computationally, if the segments of the zero level curve get too close to each other, then they will merge and this is

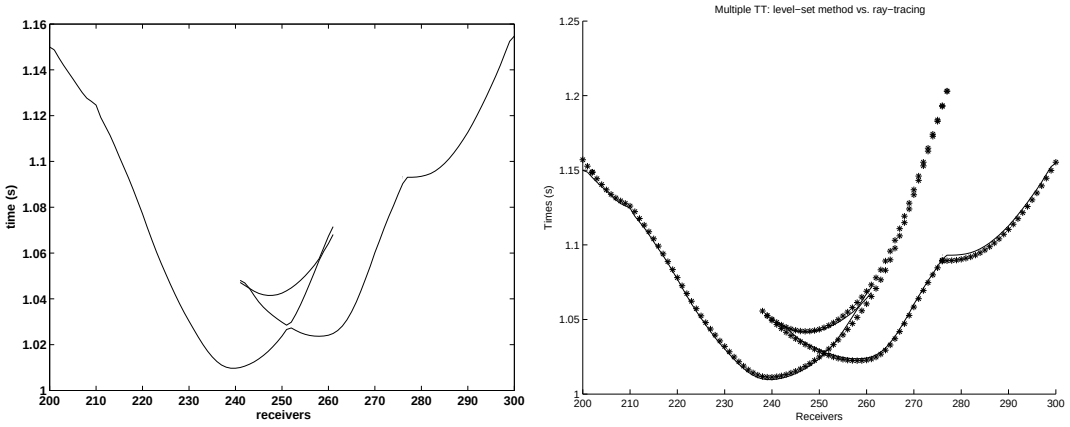


Figure 2.4: (Marmousi Model) (Left) traveltime at z=0.0 by the level-set method for Marmousi model on 100×200 grid and (right) Eulerian traveltimes (solid line) vs. ray-tracing traveltimes (*)

happening to the tip that we are interested in.

Therefore, to resolve the fine tip, we have to use a finer grid 400×200 in the x-θ space. Since the original velocity model is given on the discretized points, we use interpolation to obtain a velocity model for the finer computational mesh. The computational results are shown on Figure 2.6 and 2.7. Using this finer grid, the level-set Eulerian method yields multivalued traveltimes which match with the ray tracing solution remarkably. From Figure 2.7, the zero level set does have an elongated tip from receiver 260 to 280, and the segment near the tip are indeed very close to each other. Without a finer computational mesh, the level set method is unable to capture the tip and the related multivalued traveltimes. This demonstrates that given a discretized mesh, the resolution is fixed and the method can capture only a finite number of traveltimes.

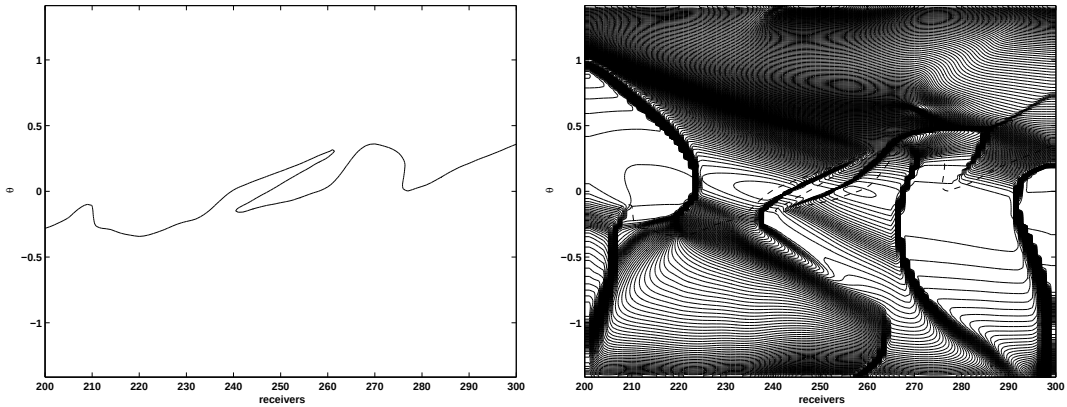


Figure 2.5: (Marmousi Model) The zero level set for Marmousi model on 100×200 grid; the zero level set overlaying contours of T at z=0.0.

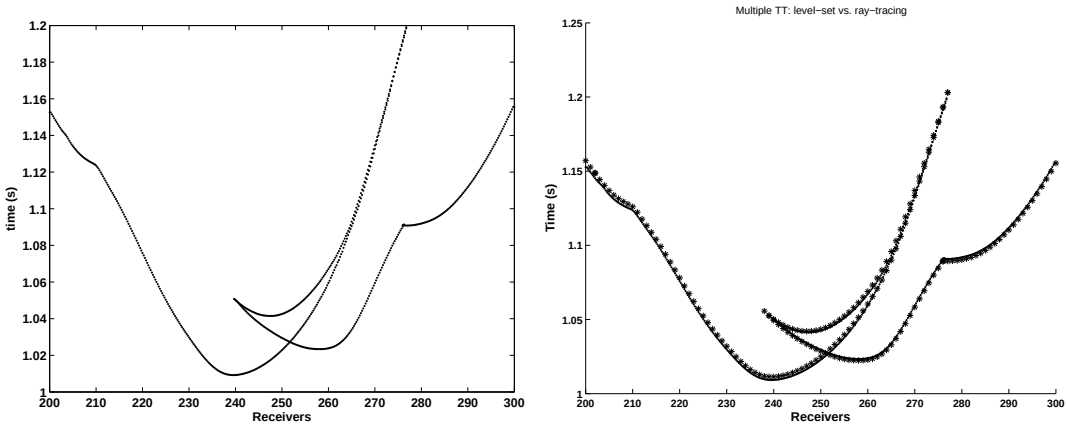


Figure 2.6: (Marmousi Model) (Left) traveltime at z=0.0 by the level-set method for Marmousi model on 400×200 grid and (right) Eulerian traveltimes (solid line) vs. ray-tracing traveltimes (star).

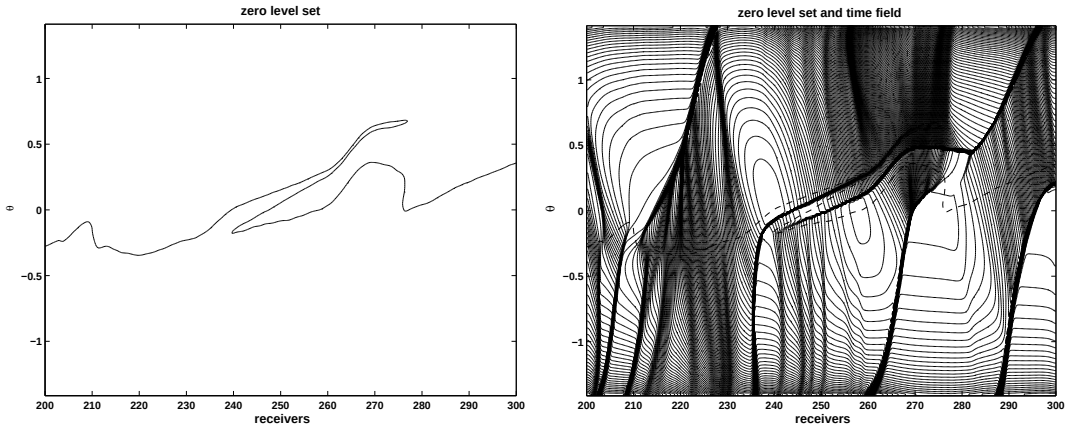


Figure 2.7: (Marmousi Model) The zero level set for Marmousi model on 400×200 grid; the zero level set overlaying contours of T at z=0.0.

2.5.4 An anisotropic model

Although a general anisotropic solid has 21 independent elastic parameters, the transversely isotropic, or TI, solid has only five. It nevertheless retains the essential features of the anisotropic case that we are interested in. Therefore, it is convenient to use TI solids as models to illustrate the advantages of our approach. We consider the simplest case for TI solids, those with vertical symmetry axes, known as VTI solids.

The elastic modulus matrix for transversely isotropic media with vertical symmetry axes has five independent components among a total of twelve nonzero components (see, e.g., [78]). A closed form solution exists in this case for the eigenvalue problem of so-called phase velocities. The quasi-P and quasi-SV slowness surfaces for VTI can be represented as a quartic polynomial equation and the quasi-SH slowness surface can be decoupled from this, leading to the equations [85]

$$c_1 p_1^4 + c_2 p_1^2 p_3^2 + c_3 p_3^4 + c_4 p_1^2 + c_5 p_3^2 + 1 \quad = \quad 0, \tag{2.63}$$

and

$$\frac{1}{2}(a_{11}-a_{12})p_1^2+a_{44}p_3^2 \;=\; 1, \tag{2.64}$$

where

$$\begin{aligned}
c_1 &\equiv a_{11}a_{44},\\
c_2 &\equiv a_{11}a_{33}+a_{44}^2-(a_{13}+a_{44})^2,\\
c_3 &\equiv a_{33}a_{44},\\
c_4 &\equiv -(a_{11}+a_{44}),\\
c_5 &\equiv -(a_{33}+a_{44}).
\end{aligned}$$

In the above equations, a_{ij} are independent elastic parameters of VTI media [78].

Thus, the phase velocities for the three different waves take the form

$$\begin{aligned}
V_{qP}^2 &= \frac{1}{2}\left(-Y_1+\sqrt{Y_1^2-4Y_2}\right),\\
V_{qSV}^2 &= \frac{1}{2}\left(-Y_1-\sqrt{Y_1^2-4Y_2}\right),\\
V_{SH}^2 &= \frac{1}{2}(a_{11}-a_{12})\sin^2\theta+a_{44}\cos^2\theta,
\end{aligned}$$

where

$$\begin{aligned}
Y_1 &= c_4\sin^2\theta+c_5\cos^2\theta,\\
Y_2 &= c_1\sin^4\theta+c_2\cos^2\theta\sin^2\theta+c_3\cos^4\theta.
\end{aligned}$$

As an example, we compute the three waves for Greenriver shale, which is a typical VTI medium [109]. The five elastic parameters are $a_{11} = 15.0638$,

$a_{33} = 10.8373$, $a_{13} = 1.6381$, $a_{44} = 3.1258$, and $a_{12} = 6.5616$.

Figure 2.8 shows traveltimes for the three different waves at depth z=0.5 computed by the level set approach plotted in circles and the ray tracing method plotted in solid line. These traveltimes are excited by a point source located at the origin. The upper-left sub-figure in Figure 2.8 shows the qP wave traveltime which is the fastest of the three waves. The lower-left sub-figure in Figure 11 shows the qSH wave traveltime. In particular, the qSV wave (the upper-right sub-figure in Figure 2.8) has cusps which imply the multivaluedness at some locations; those multivalued solutions are captured very well by the level set method. The lower-right sub-figure shows the three waves together.

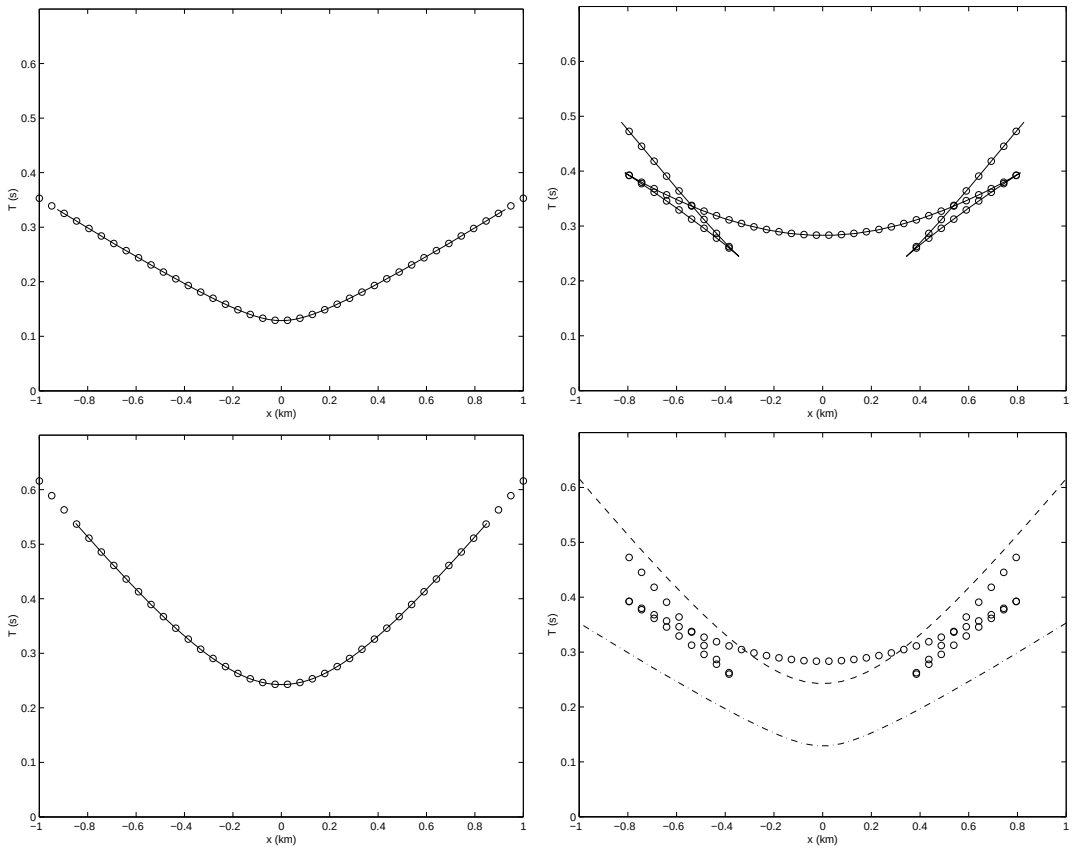


Figure 2.8: (Anisotropic Model) Anisotropic paraxial multivalued traveltimes by the level set method. qP wave traveltime: the upper-left one; qSV traveltime: the upper-right one; qSH traveltime: the lower-left one; qP-qSV-qSH: the lower-right one.

CHAPTER 3

A Local Level Set Method for Paraxial Geometrical Optics

3.1 Introduction

In the previous chapter, we have presented a global level set method for the paraxial eikonal equation in the 2-dimensional case, and the computed multivalued traveltime matches the ray tracing solution very well even in the difficult synthetic Marmousi model. In this chapter, we proceed along the same line and design a fast local level set method for both multivalued traveltimes and amplitudes, thus construct the geometrical optics term. The overall complexity of our level set method is $O(N^2 LogN)$ rather than $O(N^4)$ as typically seen in the ray tracing formulation.

3.2 Local Level Set Method

The computational complexity of the algorithm in the previous chapter can be improved dramatically by implementing a narrow banding [1] or a PDE based version [83] of local level set method. In this chapter, we adopt a version similar to the one proposed in [83].

3.2.1 Implementation

Because we are interested in only the zero level set, all the updates can actually be done in a tube centered at $\phi = 0$. The radius of this tube, γ, is picked to be $5\Delta x$, due to the fact that 5 grid values are needed for the fifth order WENO scheme when solving the advection equations. Therefore, by only considering the grid points within this tube, the complexity of the algorithm in the previous chapter can be reduced by a factor of N from $O(N^3 LogN)$ to $O(N^2 LogN)$.

Reinitialization and orthogonalization are the two core ingredients in the local level set method as proposed in [83]. Reinitialization restores the equally spaced property for the level sets. The usual way is to make ϕ a signed distance function without moving the zero level set of ϕ appreciably. To achieve this, one can solve the following equation to steady state $\tilde{\phi}_\infty$ [102, 123, 42, 83, 80]:

$$\frac{\partial \tilde{\phi}}{\partial \xi} + S(\phi)(|\nabla \tilde{\phi}| - 1) = 0 \tag{3.1}$$

$$\tilde{\phi}|_{\xi=0} = \phi(z, \cdot, \cdot)\,. \tag{3.2}$$

Here $S(\phi)$ could be the smoothed signum function [83]

$$S(\phi) = \frac{\phi}{\sqrt{\phi^2 + |\nabla \phi|^2 \Delta x \Delta \theta}}\,, \tag{3.3}$$

where Δx and $\Delta \theta$ are the mesh sizes along x- and θ- directions, respectively. However, we usually only need to evolve equation (3.1) for a few steps only. Because we are only interested in the value of T where $\phi = 0$, we apply the following orthogonalization procedure [83, 80]

$$\frac{\partial \hat{T}}{\partial \xi} + \mathrm{sgn}(\phi)\left(\frac{\nabla \phi}{|\nabla \phi|} \cdot \nabla \hat{T}\right) = 0\,, \tag{3.4}$$

$$\hat{T}|_{\xi=0} \;=\; T(z,\cdot,\cdot) \tag{3.5}$$

which, theoretically, preserves the values of T where $\phi = 0$ but changes them elsewhere such that the new T would not vary too much near the desired region. At the steady state, $\nabla\phi \cdot \nabla\hat{T} = 0$.

3.2.2 Algorithm

Hence we have the following algorithm for computing the geometrical optics terms.

Algorithm:

I. Initialization.

1. Given N_x and N_θ, determine $\Delta x = (x_{\max} - x_{\min})/(N_z - 1)$ and $\Delta\theta = 2\theta_{\max}/(N_\theta - 1)$.
2. Initialize ϕ, T, ϕ_x and ϕ_θ at $z = 0$.
3. For each (x_i, θ_j), where $i = 1, \ldots, N_x$ and $j = 1, \ldots, N_\theta$, check if any of $|\phi(x_i, \theta_j)|$, $|\phi(x_{i-1}, \theta_j)|$, $|\phi(x_{i+1}, \theta_j)|$, $|\phi(x_i, \theta_{j-1})|$ or $|\phi(x_i, \theta_{j+1})|$ is less than γ. Collect all these points and call the set Γ.

II. March in z until $z = z_{\max}$.

1. Determine Δz from the CFL condition.
2. March one Δz step by solving the level set equation (2.30) in Γ.
3. Reinitialize the level set function in Γ by solving (3.1).
4. March one Δz step by solving the traveltime equation (2.36) in Γ.
5. If $z \neq \Delta z$, update Δ by solving equation (2.58).

6. Orthogonalize T, ϕ_x and ϕ_θ to ϕ respectively in Γ by solving (3.4).
7. If $z \neq \Delta z$, orthogonalize Δ to ϕ in Γ. Otherwise, initialize Δ.
8. Update the tube Γ.
9. Detect caustics by checking if there is a change in the number of θ_k's which gives $\phi(z; x_i, \theta_k) = 0$ for two adjacent x_i.

III. Output. For each x_i with $i = 1, \ldots, N_x$,

1. Determine **all** root θ_k such that $\phi(z_{\max}; x_i, \theta_k) = 0$ $(k = 1, \cdots)$.
2. Determine $T(z_{\max}; x_i, \theta_k)$ $(k = 1, \cdots)$ by interpolation.
3. Determine $\phi_x(z_{\max}, x_i, \theta_k)$, $\phi_\theta(z_{\max}; x_i, \theta_k)$ and $\Delta(z_{\max}; x_i, \theta_k)$ $(k = 1, \cdots)$ by interpolation, and then compute $A(z_{\max}; x_i, \theta_k)$.

To determine Δz in Step II.1, we only need to scan through the grid points in the computational tube Γ in order to determine the maxima of velocity fields u and v, and this takes $O(NLogN)$ steps.

In Step II, the level set equation and the traveltime equation are decoupled and can be solved separately. The spacial derivatives are approximated by the fifth-order WENO-Godunov scheme [55] while the time derivatives are solved by the third-order TVD-RK method [82].

Unlike the global level set method, the reinitialization step in Step II.3 is used not only to regularize the level set function but also to ensure the location of the tube in Step II.8 more accurate. This step is necessary here and the number of iterations is to be determined so that the information of the location of the zero level set is propagated by a distance larger than γ, the radius of the tube. Numerically, one or two steps per iteration in z would be enough to get a reasonably good solution. However, formally, let β be the CFL number

used in the reinitialization and $m_{\min}$ be the minimum number of iterations of reinitialization; then we have

$$m_{\min} = \frac{\gamma}{\beta \min(\Delta x, \Delta\theta)} \,. \tag{3.6}$$

Because $\gamma = O(\Delta x)$, we have $m_{\min} = O(1)$. Overall the complexity of the reinitialization step for each Δz advancement would be equal to the number of grid points within the tube and is given by $O(NLogN)$.

For the root-finding and the interpolation in Step III, we can simply use any non-oscillatory interpolation scheme like linear interpolation or ENO reconstruction.

Next issue is how to update the computational tube Γ. A simple way is to scan all grid points in the domain and to apply the same procedure as that in Step I.3. The complexity of the resulting method will be $O(N^2)$. However, because the motion of the zero level set is purely advective, zeros will not be generated outside the tube. We can therefore update the tube by only scanning through the boundary of Γ, and this requires only $O(NLogN)$ operations.

As a result, for each iteration in z-direction, the complexity is $O(NLogN)$. Because of the CFL condition, the number of iterations in z is of $O(N)$. Overall the complexity of this algorithm is only $O(N^2LogN)$. Comparing to $O(N^4)$ as typically seen in the Lagrangian ray tracing method, this Eulerian method is highly efficient and attractive.

3.3 Numerical Examples

For the first two examples, we put a point source at the origin and velocity functions $c(x, z)$ are all C^∞. The third example is the synthetic Marmousi model

as described in the previous chapter. Unless specified, the computational domain we use in the following examples is chosen to be

$$\Omega_p = \{(x, \theta) : -1 \leq x \leq 1, \theta_{\max} \leq \theta \leq \theta_{\max}\} \ . \tag{3.7}$$

where $\theta_{\max} = 9\pi/20$. Accordingly, the Marmousi velocity will be rescaled to the above computational domain.

3.3.1 Waveguide Model

The velocity function is given by

$$c(x, z) \ = \ 1.1 - \exp(-0.5x^2) \, . \tag{3.8}$$

The function is symmetric with respect to $x = 0$, and we also expect the same type of symmetry in both the traveltime and the amplitude.

The solutions in Figure 3.1 show traveltimes, amplitudes and some intermediate quantities at z =1.6 using 240-by-240 grid points, respectively. The solid lines in the traveltime and amplitude plots are obtained using a ray tracing method. The solutions are symmetric as expected. Δ in this velocity model should be equal to -2 everywhere. As we mentioned earlier, the velocity model is approximated by a constant near the source and this gives the initial condition $\Delta = -2$ at $z = dz$. For this waveguide model, because $u_x = v_\theta = 0$, the equation for Δ is purely advective; therefore the exact solution is $\Delta(z; x, \theta) = -2$, which is independent of z. The variations in the subplot of Δ in Figure 3.1 are due to numerical errors.

The singularities in the amplitudes, shown in upper right subfigure in Figure 3.1, come from the vanishing of ϕ_θ on the zero level set of ϕ at around $x =$

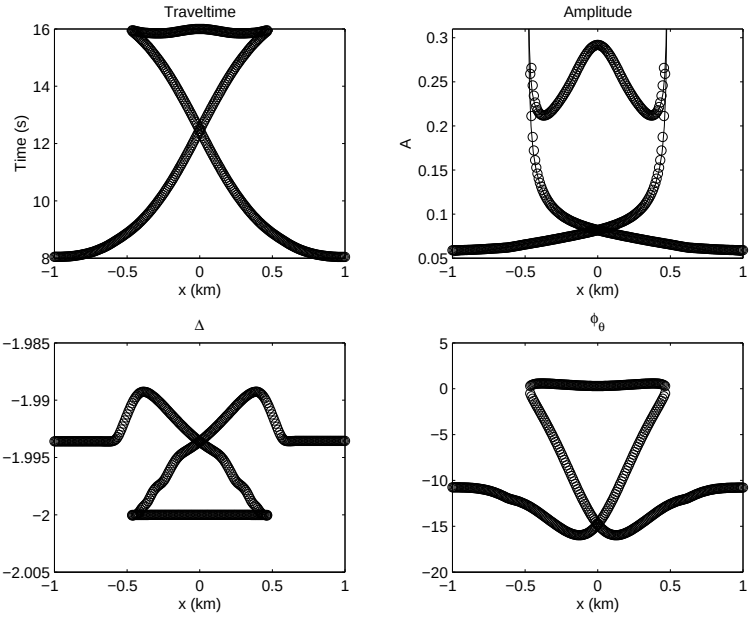


Figure 3.1: (Waveguide Model) Traveltime, amplitude, Δ and ϕ_θ at $z = 1.6$km using a 240-by-240 grid.

± 0.45km, shown in lower right subfigure in Figure 3.1.

The caustic curves detected by the local level set method are shown in Figure 3.2: circles are computed locations of the caustics in the waveguide model, and the solid lines are the rays emanating from the source computed by a ray tracing method. The caustic locations are exactly those places where rays form an envelop as seen in the figure.

To demonstrate the improvement in the computational efficiency of the current formulation, we computed the solutions of this test case with different number of grids and recorded the computational time. Using the full level set formulation, it took approximately 72 mins and 641 mins to reach the level $z = 1.6$ using 120-by-120 and 240-by-240 grids, respectively, in the phase space. With the current algorithm, the computational times reduced to 8 mins and 19 mins. If we further double the grids in each direction, the new formulation needed only

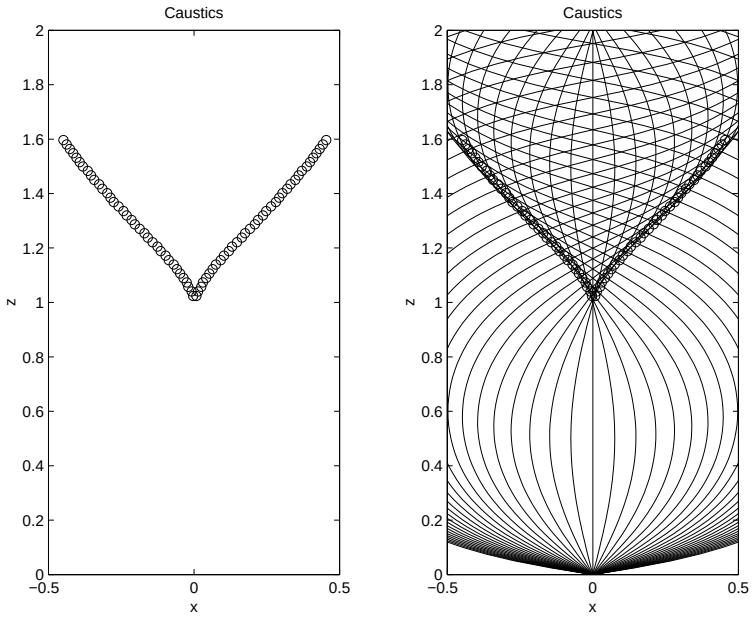


Figure 3.2: (Waveguide Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a 360-by-360 grid in the x-θ space.

58 mins, while it took more than a week to reach the same z-level using the full level set method.

3.3.2 Sinusoidal Model

This example is adapted from the sinusoidal waveguide model used in [104, 105], and the velocity function is given by

$$c(x, z) = 1 + 0.2 \sin(0.5\pi z) \sin[3\pi(x + 0.55)] \,. \tag{3.9}$$

Figures 3.3 show traveltimes, amplitudes and some intermediate quantities at $z = 2.0$ using 240-by-240 grids. The subplot for traveltimes in Figure 3.3 show that the triplications in the traveltime developed at $z = 2.0$ are clearly captured by the level set Eulerian method. Singularities in the amplitude come from the overturning of the zero level set in the phase space, i.e. $\phi_\theta = 0$, which is shown in the lower right subfigure in Figure 3.3.

Figure 3.4 shows locations of the caustics detected by the proposed method, and the results match with that from the ray tracing method.

3.3.3 Synthetic Marmousi Model

As illustrated in the previous chapter, to resolve a complicated wavefront like the one generated by the Marmousi model, we have to use a relatively fine computational mesh which requires huge memory requirement in the global level set setup. With the local level set method developed in this chapter, we are able to tackle a larger portion of the original velocity and a higher resolution with reasonable memory and computational cost. Figure 3.5 shows the zero level set and its overlay on the traveltime field by using a refinement of the original model

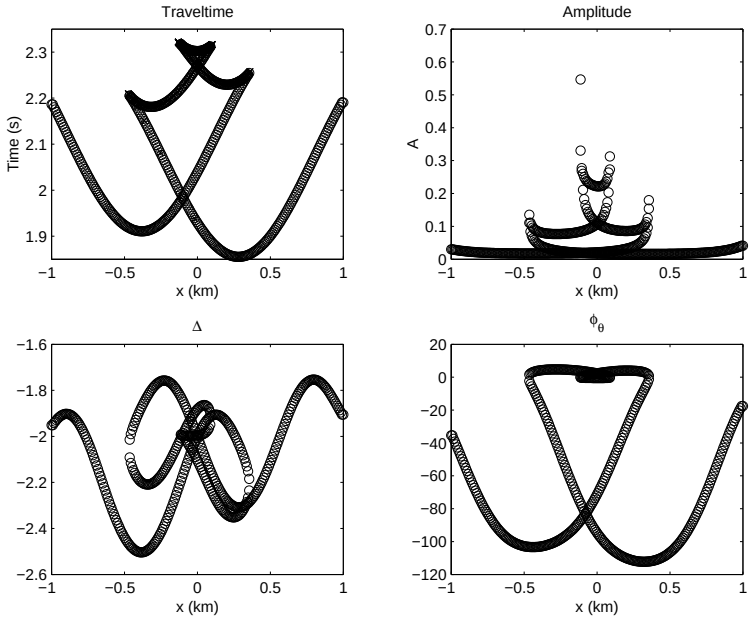


Figure 3.3: (Sinusoidal Model) Traveltime, amplitude, Δ and ϕ_θ at $z = 2.0$km using a 240-by-240 grid.

with a refinement ratio equals 8, i.e. we are using 1440 grids in the x-direction to resolve the arrivals between the Receiver 160 to 340. As we can see from the plot, the zero level set has lots of overturnings and tiny tips. Therefore, there are lots of caustics developed in the wave propagation. The resulting traveltimes at the surface along with the ray tracing solutions are shown in Figure 3.6; the two solutions match with each other reasonably well. The development of a complicated caustic can be seen near Receiver 310 as shown in the right hand side of the Figure 3.6.

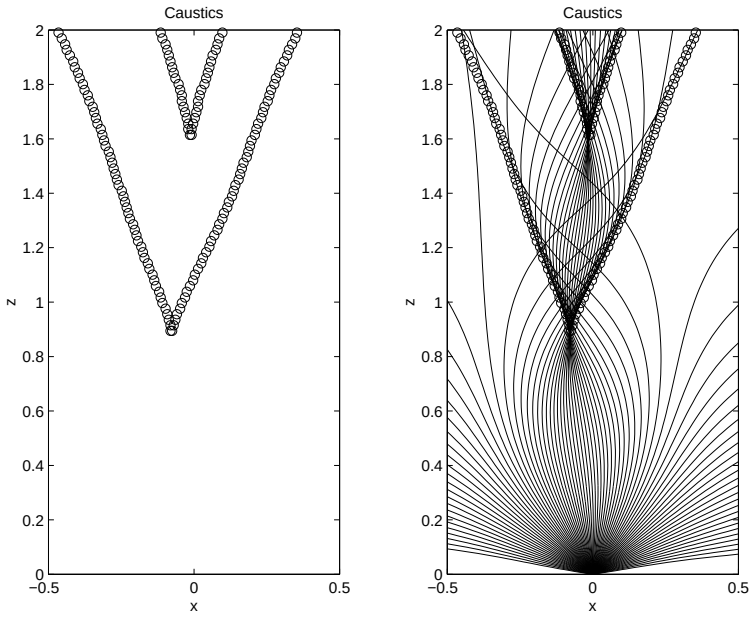


Figure 3.4: (Sinusoidal Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a 360-by-360 grid in the x-θ space.

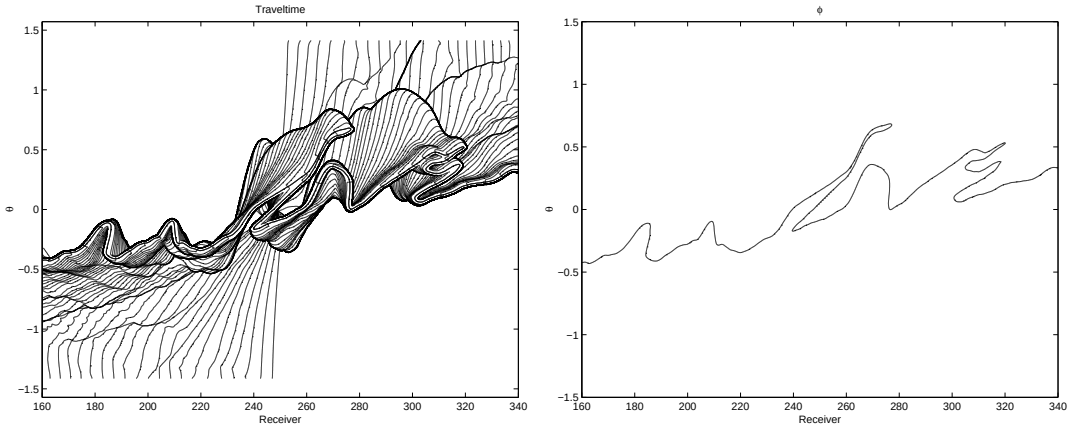


Figure 3.5: (Marmousi Model) Contours of the traveltime and the zero level set at $z = 0.0$km.

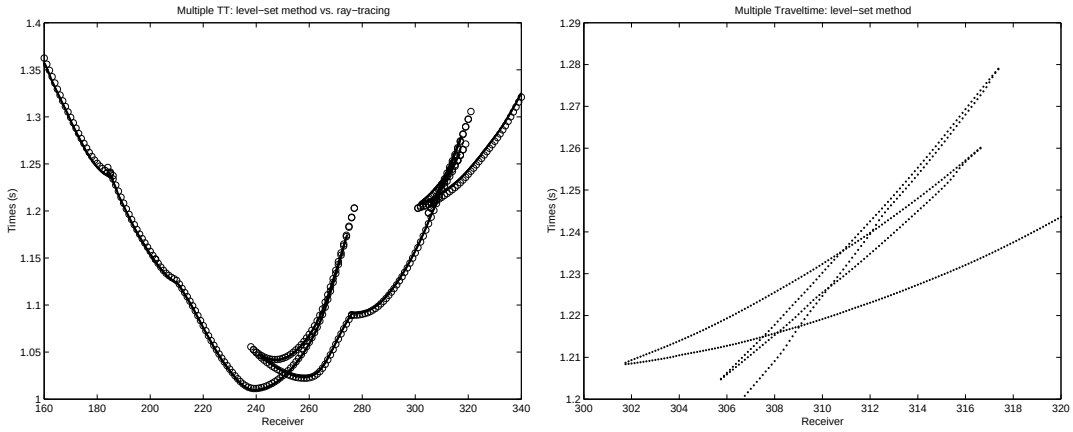


Figure 3.6: (Marmousi Model) (Left) Comparison between traveltimes using a ray tracing method and the local level set method and (right) the Zoom-in of the local level set solution at Receiver 310 at z=0.0km.

CHAPTER 4

A Level Set Method for Three-dimensional Paraxial Geometrical Optics with Multiple Point Sources

4.1 Introduction

To use the Liouville equation based phase space formulation more efficiently, we have developed the local level set method for the two-dimensional paraxial multivalued geometrical optics in Chapter 3. In this chapter, we continue to develop efficient level set methods for three dimensional multivalued geometrical optics in the paraxial formulation.

Our framework provides multivalued geometrical optics terms for multiple point sources simultaneously. The fundamental idea of this approach is that we are able to make use of the information from not only the zero level set but also all the nonzero level sets. By using a global semi-Lagrangian method to solve all level set equations, the computational memory requirement is reduced from $O(N^4)$ to $O(N^2)$, where N is the number of mesh points in each direction. Comparing to the usual finite difference discretization of the level set equations which requires $O(N^4)$ memory storage, this saving is very significant in the computational space of five dimensions. Although the method proposed here has the computational complexity $O(MN^4)$, where M is the number of steps in the ODE solver for

the semi-Lagrangian scheme, this approach can simultaneously handle up to N^2 multiple point sources.

4.2 3D Paraxial Formulation for Eikonal Equation

To apply the method of characteristics to the three-dimensional eikonal equation, we first parameterize the 3-D unit vectors by spherical coordinates. Points on a unit sphere away from the x-axis can be uniquely represented by the following rotated spherical coordinates,

$$\begin{aligned} x &= \cos\theta \\ y &= \sin\theta\sin\psi \\ z &= \sin\theta\cos\psi\,, \end{aligned} \tag{4.1}$$

where $\theta \in (0, \pi)$ is the angle between the point and the positive x-axis, and $\psi \in [-\pi, \pi)$ is the angle between the positive z-axis and the projection of the point onto the y-z plane; Figure 4.1 shows the standard and rotated spherical coordinates. With this rotated spherical coordinates, the slowness vector $\nabla\tau = \mathbf{p}$ can be represented by

$$p_1 = \frac{\cos\theta}{c} \tag{4.2}$$

$$p_2 = \frac{\sin\theta\sin\psi}{c} \tag{4.3}$$

$$p_3 = \frac{\sin\theta\cos\psi}{c} \tag{4.4}$$

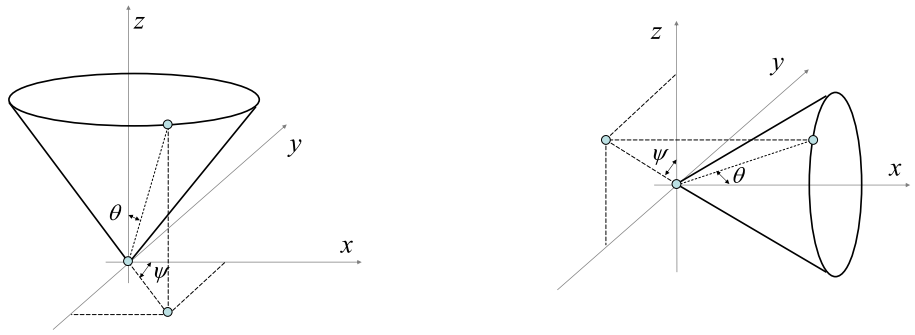


Figure 4.1: The spherical coordinates and the rotated spherical coordinates

Using the above parameterization, we have the ray tracing system

$$\begin{aligned}
\frac{dx}{dt} &= c\cos\theta \\
\frac{dy}{dt} &= c\sin\theta\sin\psi \\
\frac{dz}{dt} &= c\sin\theta\cos\psi \\
\frac{d\theta}{dt} &= \sin\theta\frac{\partial c}{\partial x} - \cos\theta\left(\cos\psi\frac{\partial c}{\partial z} + \sin\psi\frac{\partial c}{\partial y}\right) \\
\frac{d\psi}{dt} &= \frac{1}{\sin\theta}\left(\sin\psi\frac{\partial c}{\partial z} - \cos\psi\frac{\partial c}{\partial y}\right)
\end{aligned} \tag{4.5}$$

with initial conditions

$$\begin{aligned}
x|_{t=0} &= x_s \\
y|_{t=0} &= y_s \\
z|_{t=0} &= z_s \\
\theta|_{t=0} &= \theta_s \\
\psi|_{t=0} &= \psi_s
\end{aligned} \tag{4.6}$$

where $\mathbf{x}_s = (x_s, y_s, z_s)$ and θ_s and ψ_s vary from 0 to π and from $-\pi$ to π respectively. One can compare this formulation with the one from [58, 79]. The reason why we use this rotated spherical coordinates rather than the standard spherical coordinates system is that we can now uniquely represent points on the z-axis. Although points on the x-axis in this rotated coordinate system may still seem to cause problem, these points are actually out of our computational domain according to the paraxial assumption which will be discussed below.

Next we extend the traveltime function $\tau(x, y, z)$ to the reduced phase space $\{(x, y, z, \theta, \psi)\}$, denoted by $T(x, y, z, \theta, \psi)$, and consider the t-wavefront expanding from the source point:

$$T(x, y, z, \theta, \psi) \;=\; t. \tag{4.7}$$

Differentiating this identity with respect to t, we have

$$\frac{dx}{dt}T_x + \frac{dy}{dt}T_y + \frac{dz}{dt}T_z + \frac{d\theta}{dt}T_\theta + \frac{d\psi}{dt}T_\psi = 1 \tag{4.8}$$

with the boundary condition

$$T(x_s, y_s, z_s, \theta_s, \psi_s) = 0\,, \tag{4.9}$$

for $0 \le \theta_s \le \pi$ and $-\pi \le \psi_s \le \pi$.

Since equation (4.8) is a linear advection equation, one might try to solve it directly with the condition (4.9). However, for a given $(x, y, z) \neq (x_s, y_s, z_s)$, $T(x, y, z, \cdot, \cdot)$ is well defined for at least one (θ, ψ) corresponding to the first arrival but is not necessarily well defined for all θ and ψ, which implies that the solution surface T in the phase space is extremely singular. In other words, equations (4.8)

and (4.9) are not well-posed. To obtain a well-posed problem, we will assume the paraxial condition and use the level set formulation.

Now, we consider the domain

$$\Omega \;=\; \{(x,y,z) : x_{\min} \le x \le x_{\max}, y_{\min} \le y \le y_{\max}, 0 \le z \le z_{\max}\} \quad (4.10)$$

and assume that the point source is located on the surface: $x_{\min} \le x_s \le x_{\max}$, $y_{\min} \le y \le y_{\max}$ and $z_s = 0$. By the sub-horizontal condition we can use depth z as the running parameter so that we have the following reduced system

$$\begin{aligned} x_z &= \frac{1}{\cos\psi \tan\theta} \\ y_z &= \tan\psi \\ \theta_z &= \frac{c_x}{c\cos\psi} - \frac{c_z + c_y \tan\psi}{c\tan\theta} \\ \psi_z &= \frac{c_z \tan\psi - c_y}{c\sin^2\theta}, \end{aligned} \quad (4.11)$$

with $x \in [x_{\min}, x_{\max}]$, $y \in [y_{\min}, y_{\max}]$, $\theta \in [\epsilon_\theta, \pi - \epsilon_\theta]$ and $\psi \in [\epsilon_\psi - \pi/2, \pi/2 - \epsilon_\psi]$.

4.3 Level Set Formulation

4.3.1 Representation of a Point Source

We first assume that a single point source is located at the origin. Therefore, rays from this point source can now be represented as the intersection of zero level sets of two level set functions, $\phi^1(z; x, y, \theta, \psi)$ and $\phi^2(z; x, y, \theta, \psi)$.

Differentiating the zero level set of these functions with respect to z, we get the following level set equations which govern the motion of the corresponding

zero level sets,

$$\phi_z^m + \frac{dx}{dz}\phi_x^m + \frac{dy}{dz}\phi_y^m + \frac{d\theta}{dz}\phi_\theta^m + \frac{d\psi}{dz}\phi_\psi^m = 0 \tag{4.12}$$

for $m = 1, 2$. Or, these equations can be rewritten as

$$\phi_z^m + \mathbf{u} \cdot \nabla_{x,y,\theta,\psi}\phi^m = 0 \tag{4.13}$$

for $m = 1, 2$ where the velocity field $\mathbf{u} = (u^1, u^2, u^3, u^4)$ is given by the ray tracing system (4.11).

On $z = 0$, we initialize the level set functions by

$$\phi^1(0; x, y, \theta, \psi) = x \text{ and } \phi^2(0; x, y, \theta, \psi) = y\,. \tag{4.14}$$

4.3.2 Traveltime

Traveltime can be computed by inverting the third equation in (4.5) locally. This gives

$$T_z + \mathbf{u} \cdot \nabla_{x,y,\theta,\psi} T = \frac{1}{c \sin\theta \cos\psi}\,, \tag{4.15}$$

where $T = T(z; x, y, \theta, \psi)$; this equation will be solved along with (4.13).

To obtain the multivalued traveltime on $z = z^*$, we first solve equations (4.13) and (4.15) up to $z = z^*$. We then compute the intersection of the zero level sets, denoted by

$$\Sigma_{\mathbf{0}} = \{(x, y, \theta, \psi) : \phi^1(z^*; x, y, \theta, \psi) = \phi^2(z^*; x, y, \theta, \psi) = 0\} \subset \mathbb{R}^4\,. \tag{4.16}$$

The traveltimes at $(z^*; x, y)$ can be determined by projecting $T(z^*; \Sigma_{\mathbf{0}})$ onto the x-y plane.

4.3.3 Representation of Multiple Sources

However, we have noticed that the level set functions contain much more information than what we have used in the above algorithm since not only the zero level set but also the non-zero level sets are also useful. Rays emanating from a point source in the phase space are not necessarily represented by the intersection of two *zero* level sets. We can define rays from a point source at a location (x_s, y_s) by the intersection of $\{\phi^1 = x_s\}$ and $\{\phi^1 = y_s\}$. Under the same velocity field, given by $\mathbf{u}$, to find all rays on $z = z^*$ from this point source (x_s, y_s), one only needs to determine the set

$$\Sigma_{\mathbf{x}_s} = \{(x, y, \theta, \psi) : \phi^1(z^*; x, y, \theta, \psi) - x_s = \phi^2(z^*; x, y, \theta, \psi) - y_s = 0\}, \quad (4.17)$$

rather than the one defined by the set (4.16). In other words, using the initial conditions (4.14), we can connect a point $(x_i, y_j, \theta_k, \psi_l)$ on $z = z^*$ to the point

$$(\phi^1(z^*; x_i, y_j, \theta_k, \psi_l), \phi^2(z^*; x_i, y_j, \theta_k, \psi_l), \theta_0, \psi_0)$$

on $z = 0$ through the characteristic curve of (4.13) for some initial conditions (θ_0, ψ_0). More importantly, the initial conditions (4.14) can be reinterpreted as follows. All points, not only (x_i, y_j), but also $\forall x \in [x_{\min}, x_{\max}]$ and $\forall y \in [y_{\min}, y_{\max}]$, can be treated as locations of point sources. Therefore, we are able to determine multi-arrival rays from multiple point sources.

The representation of point sources discussed above is of course not the only way that we can represent rays from multiple point sources. One can actually define $\Sigma_{\mathbf{0}}$ to denote all rays from *multiple* point sources. The only thing we need to modify is the initial conditions. $\phi^1(0; x, y, \theta, \psi)$ and $\phi^2(0; x, y, \theta, \psi)$ need to be independent of the angles θ and ψ, and $\{\phi^1(0; x, y) = 0\} \cap \{\phi^2(0; x, y) = 0\}$

at locations of those point sources. With this formulation, one can still get all the multi-arrival rays from multiple point sources. But, on the level $z = z^*$, one cannot distinguish rays from different point sources because all rays are essentially represented by the same intersection of zero level sets. However, using the formulation proposed here, we can separate the rays from different point sources by using only one set of level set functions and making use of all the available level sets.

4.3.4 Amplitude

The amplitude of a ray can also be computed using the current formulation. Defining $\tilde{A} = \tilde{A}(z; x, y)$, we have

$$\tilde{A}(z; x, y) = \frac{c}{4\pi\sqrt{c_0}} \sqrt{\sin\tilde{\Theta} \left| \frac{\partial(\tilde{T}, \tilde{\Theta}, \tilde{\Psi})}{\partial(x, y, z)} \right|} \tag{4.18}$$

where $\tilde{T} = \tilde{T}(z; x, y)$, $\tilde{\Theta} = \tilde{\Theta}(z; x, y)$ and $\tilde{\Psi} = \tilde{\Psi}(z; x, y)$ are the traveltime, take-off angles of θ and ψ from the point source located at $\mathbf{x} = \mathbf{x}_s$, respectively. Following the approach in Chapter 2, we first extend all these functions into the phase space, denoted as T, Θ and Ψ, respectively. Using (4.13) and (4.15), we obtain

$$A(z; x, y, \theta, \psi) = \frac{1}{4\pi} \sqrt{\frac{c \sin\Theta}{c_0 \sin\theta \cos\psi}} \sqrt{\frac{\Delta_1}{\Delta_2}}, \tag{4.19}$$

where Θ and θ are the takeoff angle and the arrival angle respectively, and Δ_1 and Δ_2 are the Jacobians of the transformation given by

$$\Delta_1 = \begin{vmatrix} \phi_x^1 & \phi_x^2 & \Theta_x & \Psi_x \\ \phi_y^1 & \phi_y^2 & \Theta_y & \Psi_y \\ \phi_\theta^1 & \phi_\theta^2 & \Theta_\theta & \Psi_\theta \\ \phi_\psi^1 & \phi_\psi^2 & \Theta_\psi & \Psi_\psi \end{vmatrix}, \text{ and } \Delta_2 = \begin{vmatrix} \phi_\theta^1 & \phi_\theta^2 \\ \phi_\psi^1 & \phi_\psi^2 \end{vmatrix}. \tag{4.20}$$

4.4 Numerical Method

4.4.1 Level Set Equations

One way to solve both equations (4.13) and (4.15) is to use, for example, RK3 in the z-direction and WENO5 upwind scheme in the x-y-θ-ψ space [71, 98]. This is a typical Eulerian approach. Computational complexity of this method is therefore $O(N^5 LogN)$. This implies that the Eulerian approach is not computationally efficient in a high dimensional space as in the current application. One reason is that, if we want to compute all multivalued arrivals from multiple point sources, we probably do not want to use any localized level set methods, like the one proposed in [83] or the one we have proposed in the previous chapter. In this case, keep tracking of multiple layers of level set functions with reinitializations and extensions will take a large portion of computational time.

Another potential difficulty is the limitation from the CFL condition when solving these hyperbolic equations. For this Eulerian approach, each z-direction marching is of $O(\min(\Delta x, \Delta y, \Delta\theta, \Delta\psi))$. This is acceptable for lower dimensional computations, like those in Chapter 2 and Chapter 3 where the dimension involved is only 1+2 (time-like direction plus space directions). For the current problem however, it is unreasonable to spend days of computations in solving

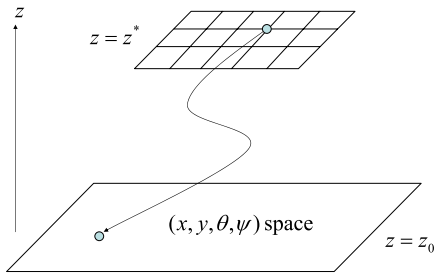


Figure 4.2: Semi-Lagrangian method to solve the advection equations.

these linear advection equations.

In this chapter, we implement a semi-Lagrangian method [101, 41] to determine the values of ϕ^m at $z = z^*$ for $m = 1, 2$. Solving the level set equations with the method of characteristics, we have $\phi^m =$ const for $m = 1, 2$ along the characteristics given by (4.11). Following the idea of semi-Lagrangian methods, we trace back in the z-direction until $z = z_0$, i.e. we solve

$$\frac{d(\hat{x}, \hat{y}, \hat{\theta}, \hat{\psi})}{dz} = \mathbf{u} \tag{4.21}$$

for $(\hat{x}, \hat{y}, \hat{\theta}, \hat{\psi})|_{z=z_0}$ with *initial* conditions $(\hat{x}, \hat{y}, \hat{\theta}, \hat{\psi})|_{z=z^*} = (x, y, \theta, \psi)$ at the current point, as shown in Figure 4.2. Then the level set values are assigned as

$$\begin{aligned} \phi^1(z^*; x, y, \theta, \psi) &= \hat{x}|_{z=z_0} \\ \phi^2(z^*; x, y, \theta, \psi) &= \hat{y}|_{z=z_0}\,. \end{aligned} \tag{4.22}$$

Numerically, the above ODE system is solved using RK3, and the step size is independent of the number of grid points used in the computational domain. By using this semi-Lagrangian approach, the computational complexity drops

to $O(MN^4)$, where M is the number of iterations in the z-direction and N is the number of grid points in each of x-y-θ-ψ direction. Different from the finite difference Eulerian approach, the factor M is independent of N and is chosen mainly for the purpose of *accuracy*. In the current implementation, we choose M large enough so that the errors from the RK3 ODE solver are negligible comparing to the errors from the linear interpolation on the uniform mesh used in finding intersections of level set functions.

A simpler case is to consider only one point source on $z = z_0$. Then the order of complexity of using the global level set method approach as in Chapter 2 can still be reduced by a factor of N^2 to $O(N^3 LogN)$ if we apply the local level set method we discussed in Chapter 3. However, the memory requirement, which will be addressed later in Section 4.4.3, may still make the local level set method difficult to implement. Moreover, to compute arrival solutions from N^2 point sources individually, one may need to solve N^2 times localized level set equations which makes the overall computational complexity back to $O(N^5 LogN)$.

4.4.2 Traveltime Equation

For the traveltime equation, we have

$$\frac{DT}{Dz} = \frac{1}{c \sin\theta \cos\psi}\,, \tag{4.23}$$

where $\frac{D}{Dz}$ is the material derivative given by

$$\frac{D}{Dz} = \frac{\partial}{\partial z} + \mathbf{u}\cdot\nabla = \frac{\partial}{\partial z} + x_z\frac{\partial}{\partial x} + y_z\frac{\partial}{\partial y} + \theta_z\frac{\partial}{\partial \theta} + \psi_z\frac{\partial}{\partial \psi}\,. \tag{4.24}$$

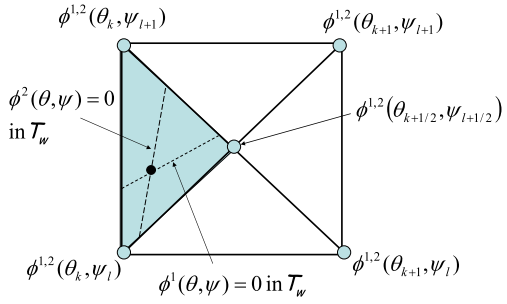


Figure 4.3: Determining the intersection of level set functions.

Therefore, we get

$$T = \int_\Gamma \frac{ds}{c \sin\theta \cos\psi} \tag{4.25}$$

where $T(0; x, y, \theta, \psi) = 0$ by the reciprocity and Γ is the characteristic given by the system (4.21). Again, RK3 is used to integrate the traveltime of rays.

4.4.3 Multivalued Traveltimes

After we solve for ϕ^m (m=1,2) and T on the grid points at the time level $z = z^*$, we need to compute the intersection of the level surfaces $\{\phi^1 = x_s\} \cap \{\phi^2 = y_s\}$. To simplify this computation, we discretize the $\theta - \psi$ space for each point (x_i, y_j) in the following way. We first use rectangular grids in the $\theta - \psi$ space, giving (θ_k, ψ_l). One more grid point, denoted by $(\theta_{k+1/2}, \psi_{l+1/2})$, will then be added to the center of each cell, as seen in Figure 4.3. Therefore, each grid cell, vertices at (θ_k, ψ_l), (θ_{k+1}, ψ_l), (θ_k, ψ_{l+1}) and $(\theta_{k+1}, \psi_{l+1})$, will be sub-divided into 4 triangles, denoted by $\mathcal{T}_N$, $\mathcal{T}_E$, $\mathcal{T}_S$ and $\mathcal{T}_W$. ϕ^m (m=1,2) and T will be computed at all points $(x_i, y_j, \theta_k, \psi_l)$ and $(x_i, y_j, \theta_{k+1/2}, \psi_{l+1/2})$. On each triangle $\mathcal{T}_{(.)}$, we interpolate ϕ^1 and ϕ^2 linearly. Intersection of the level curves of ϕ^1 and ϕ^2 in each of the triangles $\mathcal{T}_{(.)}$, $\{\phi^1 = x_s\}|_{\mathcal{T}_{(.)}} \cap \{\phi^2 = y_s\}|_{\mathcal{T}_{(.)}}$, is computed, if any. The traveltime

at this intersection point will be interpolated linearly using the values of T at the vertices of the triangle $\mathcal{T}_{(.)}$.

We emphasize that we solve each of the level set equations (4.13) and the traveltime equation (4.15) only once, even if we care about more than one point source on $z = z_0$. It is the intersection of the level surfaces that we need to repeat for each of the point sources.

In the Lagrangian or the semi-Lagrangian approach, on the other hand, grid points in the $x - y$ space are independent of each other throughout all processes above. Level set function values at any two points $(x_i, y_j, \cdot, \cdot)$ and $(x_{i'}, y_{j'}, \cdot, \cdot)$, with $i \neq i'$ or $j \neq j'$, are determined independently through solving a system of ODE's with different initial conditions. Although points in the θ-ψ space are dependent upon each other through the processes in determining $\Sigma_{\mathbf{x}_s}$ and $T(z^*; \Sigma_{\mathbf{x}_s})$, numerically, the memory allocation can still be reduced to $O(N^2)$ as long as the multiple point source locations are given at the beginning, no matter how many there are. However, if there are up to $O(N^2)$ point sources given, the computational complexity will be $O(MN^4)$.

This reduction in the memory requirement may not be significant in the two dimensional paraxial geometrical optics formulation. However, it is important in the current calculations in the 1+4 dimensions (time-like direction plus $x - y - \theta - \psi$ directions).

4.4.4 Reinitialization and Intersection

In our current formulation, all the values of level set functions are used rather than only the zero value. The usual reinitialization is a process of reconstructing a signed distance function to the zero level set so that the only useful information from the original level set function is concentrated at/near the zero level set.

Therefore, the information at other places is no longer meaningful. Hence, it is impossible to apply such a technique in our setup.

On the other hand, we have to find the intersections of the level sets eventually, i.e. to determine (4.17). Since we use the semi-Lagrangian method to solve the level set equations, the accuracy of the level set functions solely depends on the ODE solver rather than on the number of grid points in the (x, y, θ, ψ) space. However, to obtain accurate intersections of both zero level sets and non-zero level sets, we may refine the grids in the (θ, ψ) space, as illustrated in Figure 4.3.

4.4.5 Amplitude

We notice that the takeoff angles Θ and Ψ are the by-products in the above computations of ϕ^1 and ϕ^2. We can simply set

$$\begin{aligned} \Theta(z^*; x, y, \theta, \psi) &= \hat{\theta}(z = z_0) \\ \Psi(z^*; x, y, \theta, \psi) &= \hat{\psi}(z = z_0)\,. \end{aligned} \tag{4.26}$$

However, in computing Δ_1, instead of numerically differentiating these four functions with respect to x, y, θ and ψ respectively, we notice that this quantity satisfies the equation

$$\frac{D\Delta_1}{Dz} = -(\nabla \cdot \mathbf{u})\Delta_1 \tag{4.27}$$

where $\Delta_1(0; x, y, \theta, \psi) = 1$ by the reciprocity principle and

$$\nabla \cdot \mathbf{u} = \frac{(1 + \cos^2\psi)c_z + (\tan\psi \cos^2\psi)c_y}{c \sin^2\theta \cos^2\psi}\,. \tag{4.28}$$

In turn we solve this equation by introducing $\tilde{\Delta}_1 = \log(\Delta_1)$. Then, similar to

the computation of the traveltime, we integrate

$$\tilde{\Delta}_1 = \int_\Gamma -(\nabla \cdot \mathbf{u})\, ds \tag{4.29}$$

with $\tilde{\Delta}_1(0; x, y, \theta, \psi) = 0$ using RK3. Finally, we have

$$\Delta_1 = \exp(\tilde{\Delta}_1)\,. \tag{4.30}$$

For the quantity Δ_2, we can simply use the linear interpolants of ϕ^1 and ϕ^2 in the triangle $\mathcal{T}_{(.)}$ when we determining $\Sigma_{\mathbf{X}_s}$. Assuming that those linear interpolants are

$$\begin{aligned} \phi^1_{\mathcal{T}} &= a_{11}\theta + a_{12}\psi + b_1 \\ \phi^2_{\mathcal{T}} &= a_{21}\theta + a_{22}\psi + b_2\,, \end{aligned} \tag{4.31}$$

we have $\Delta_2 = |a_{11}a_{22} - a_{12}a_{21}|$ defined on the triangle $\mathcal{T}_{(.)}$.

To compute $A(z^*; \Sigma_{\mathbf{X}_s})$, we first determine the quantity

$$\alpha = \frac{\Delta_1 \sin\Theta}{\cos\psi \sin\theta} \tag{4.32}$$

at each grid point. Then, we use linear interpolation to find its value at the intersection $\{\phi^1 = x_s\}|_{\mathcal{T}_{(.)}} \cap \{\phi^2 = y_s\}|_{\mathcal{T}_{(.)}}$. Finally, we have

$$A(z^*; \Sigma_{\mathbf{X}_s}) = \frac{1}{4\pi}\sqrt{\frac{c}{c_s}}\sqrt{\frac{(\alpha)_{\Sigma_{\mathbf{X}_s}}}{(\Delta_2)_{\mathcal{T}_{(.)}}}}\,. \tag{4.33}$$

4.5 Numerical Examples

In the following numerical examples, we use the computational domain

$$\Omega = \{(x, y, \theta, \psi) : x \in [-1, 1], y \in [-1, 1], \theta \in [\pi/20, 19\pi/20], \psi \in [-9\pi/20, 9\pi/20]\} . \tag{4.34}$$

Multiple point sources are located on $z = 0$. Their coordinates (x_s, y_s) (s=1, $\cdots$, 4) are given by $(0, 0)$, $(0.3, 0.4)$, $(-0.1, 0.2)$ and $(0.2, -0.3)$. It should be emphasized that the number of point sources can be up to $O(N^2)$ and their locations can be arbitrary.

4.5.1 Waveguide Model

This velocity model is the same as the one in previous chapters where

$$c(x, y, z) = 1.1 - \exp(-0.5x^2) . \tag{4.35}$$

Figure 4.4 shows the computed multivalued traveltimes at $z = 1.2$ with point sources located at (x_s, y_s) (s =1,$\cdots$,4), respectively using $M = 24$, $1 \le i, j \le 41$ and $1 \le k, l \le 201$. As the point source varies, the traveltime varies for a ray to reach a specific location. Solutions using ray tracing method are given in Figure 4.5. For these ray tracing solutions, rays are emitted from the point sources located at (x_s, y_s) with initial angles (θ, ψ) given by uniformly partition the angle space into 50 by 50 grids. The paraxial ray tracing system (4.11) is then solved using RK45 until $z = 1.2$. As expected, we can see from the figures, we do not have uniform resolutions in the solution. Rays are more concentrated on the later arrivals. Solutions from the first arrivals, on the other hand, are relatively poorly resolved.

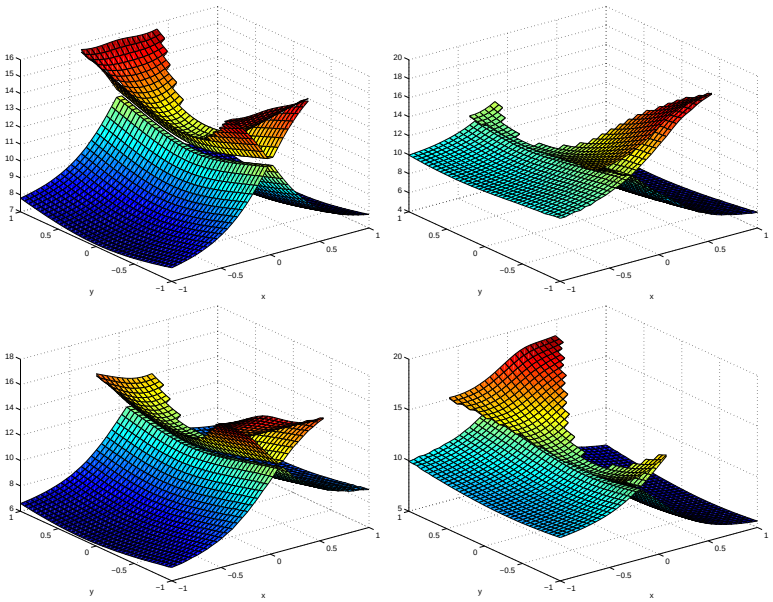


Figure 4.4: (Waveguide Model) Traveltimes in the physical space.

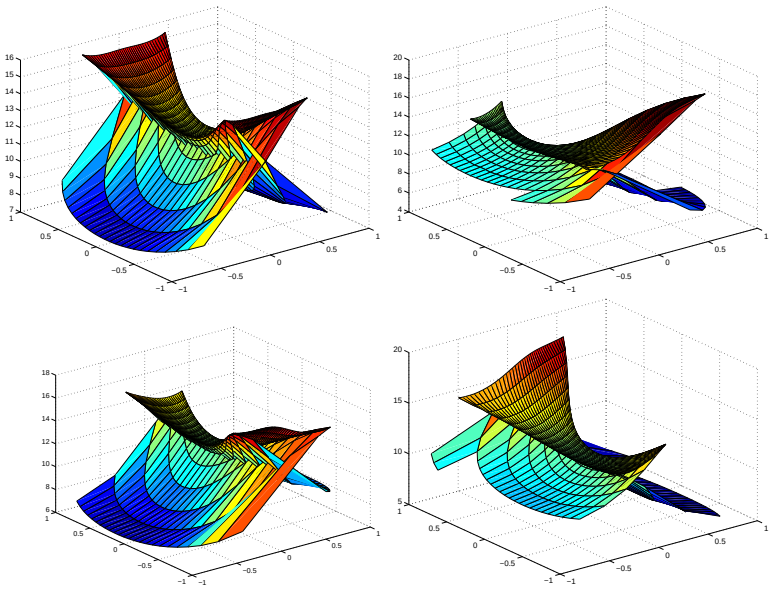


Figure 4.5: (Waveguide Model) Traveltimes in the physical space using ray tracing method.

Because of this non-uniform resolution in the solutions, it is difficult to compare the solutions using the current level set formulation with that from the ray tracing method directly. However, in this case, $c_y = c_z = 0$ and this implies

$$u^3 = \frac{c_x}{c\cos\psi} \text{ and } u^4 = 0\,. \tag{4.36}$$

Considering rays shooting out from the origin with $\psi = 0$ and using the paraxial ray tracing system (4.11), we get $\psi(z) \equiv 0$. This means that all the rays with $\psi(z = 0) = 0$ from the origin will stay on the cross section $y = 0$ (one can compare our solution with that in previous chapters). We compare the solution of this cross section with the one from the ray tracing method, as shown in the first sub-figure of Figure 4.6. The solution from the ray tracing method is plotted in the solid line, while the circles represent the solution from the current formulation. They match with each other very well.

There are three sheets of traveltime surfaces at $z = 1.2$ for those given point sources. When two sheets connect to each other in the phase space, they connect along the caustic curves, which are shown in the figures; this can also be seen more clearly in Figure 4.6. In all the sub-figures, we can see that the caustics develop and the traveltime becomes triple valued around caustics. To look at the solutions more closely, we concentrate on the solution for the point source located at $(x_s, y_s) = (0, 0)$. More cross sections of the multivalued traveltime are plotted on Figure 4.7. The locations of the slides are $y = (j - 21)/20$ for $j = 6, 11, 16, 26, 31, 36$. One can imagine that the leftmost slice is shifted to the left to the right and scaled according to the distance to the origin, since the velocity is a function of the x variable only.

Amplitudes of the arrival rays are also calculated. Figures 4.8 to 4.10 show the amplitude solutions corresponding to Figures 4.4, 4.6 to 4.7. In the calculation

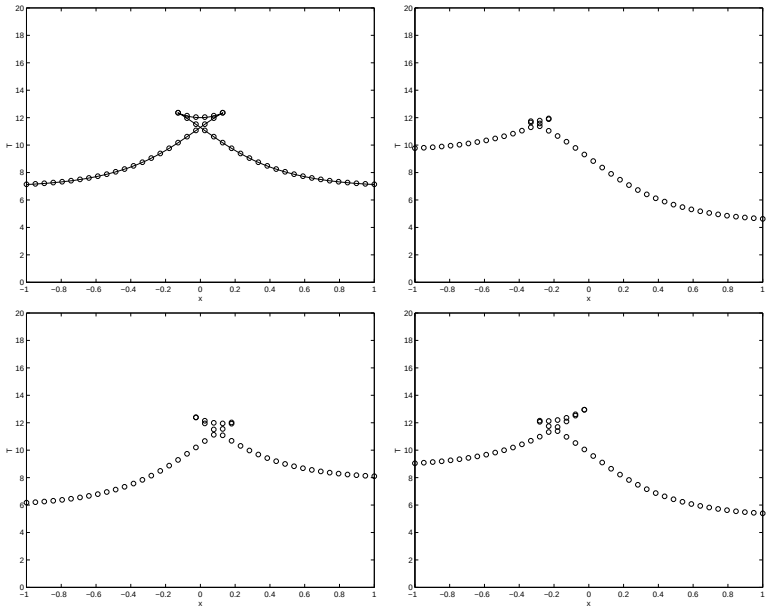


Figure 4.6: (Waveguide Model) Traveltimes in the physical space on the cross section $y = 0$. Solution using ray tracing method is plotted using solid line.

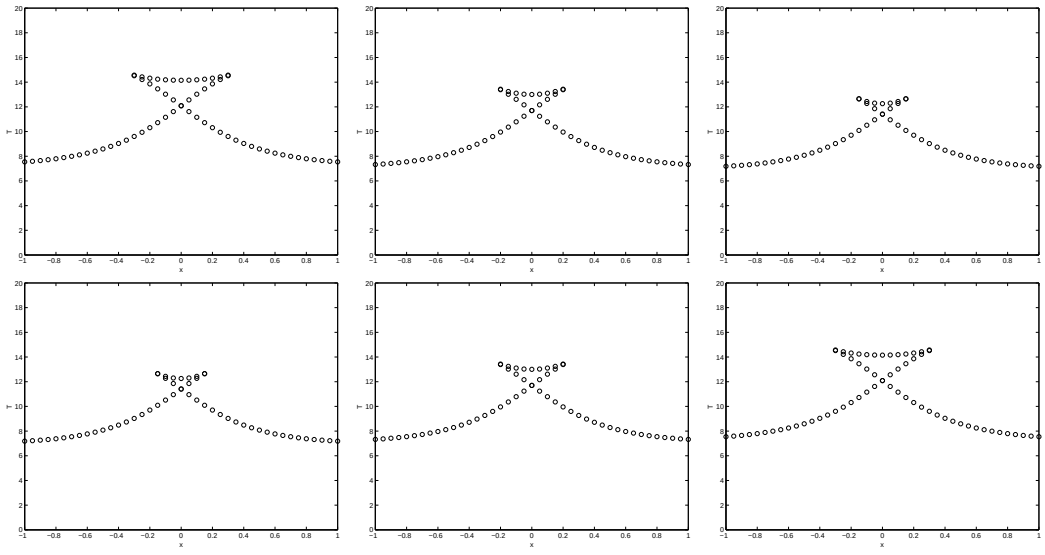


Figure 4.7: (Waveguide Model) Traveltimes in the physical space on different cross sections.

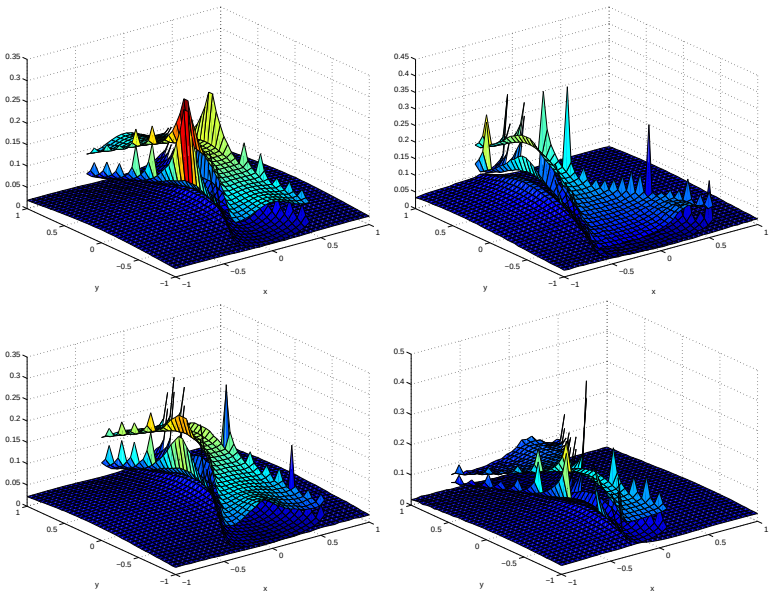


Figure 4.8: (Waveguide Model) Amplitudes in the physical space.

of the amplitude, one needs to calculate the Jacobian Δ_2. This quantity can be zero which corresponds to the location of *caustics*. Near caustics, the usual asymptotic expansion of the wave field is not valid anymore. This reflects in the fact that the amplitude blows up, as seen clearly in these figures.

4.5.2 Vinje's Gaussian Model

This velocity model comes from [117] where

$$c(x, y, z) = 3 - 1.75 \exp\left(-\frac{x^2 + y^2 + (z - 0.75)^2}{0.5^2}\right) . \tag{4.37}$$

Figure 4.11 shows the multivalued traveltimes at $z = 1.5$ with sources located at (x_s, y_s) for $s = 1, \cdots, 4$ respectively using $M = 15$, $1 \leq i, j \leq 51$ and $1 \leq k, l \leq 401$. As the source varies, the traveltime varies for a ray to reach a specific location. There are three sheets of traveltime surfaces at $z = 1.5$ for those given

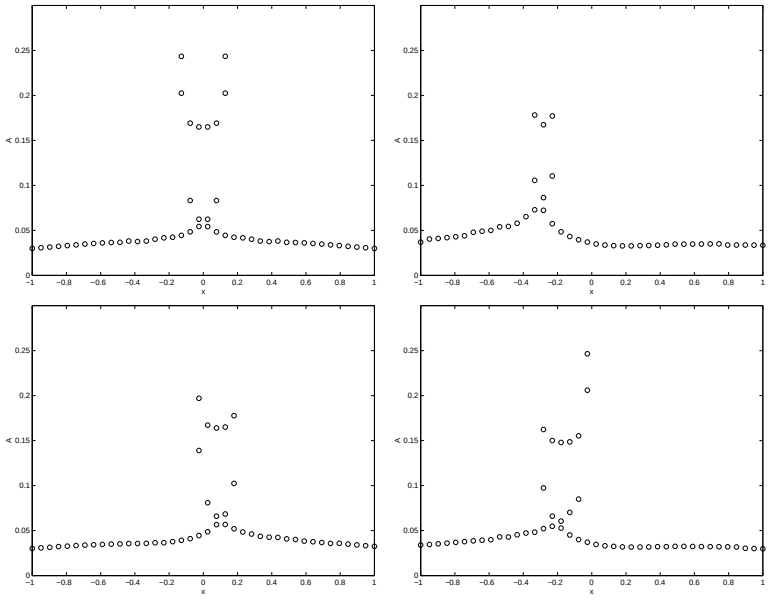


Figure 4.9: (Waveguide Model) Amplitudes in the physical space on the cross section $y = 0$.

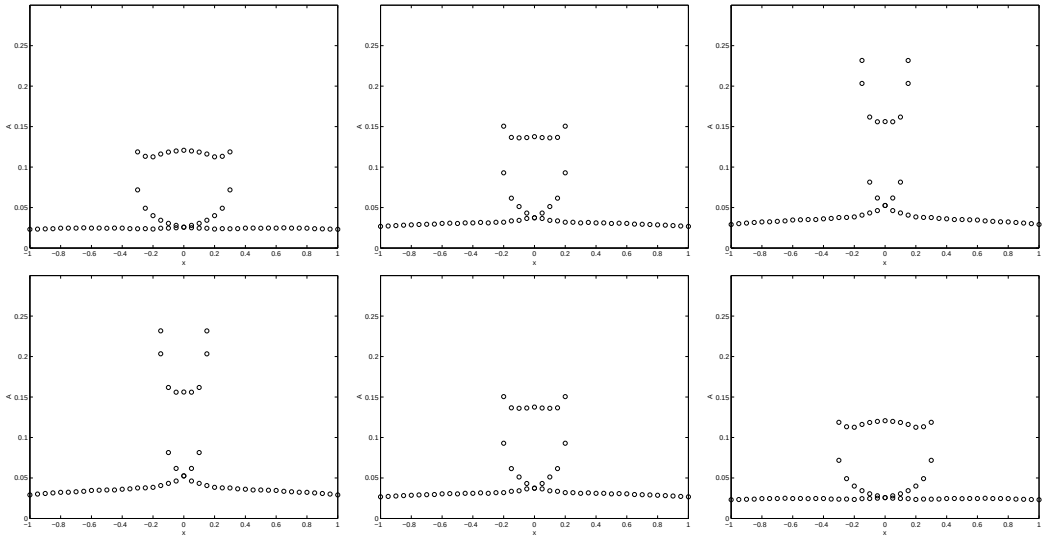


Figure 4.10: (Waveguide Model) Amplitudes in the physical space on different cross sections.

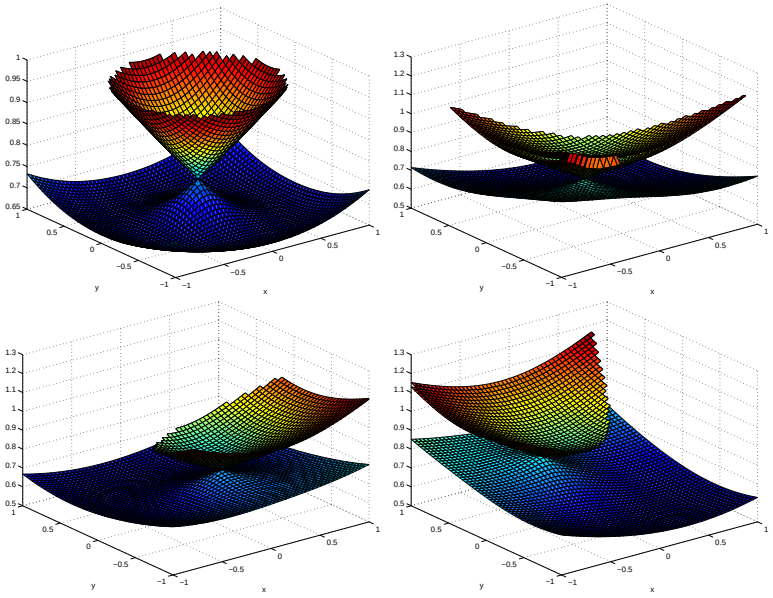


Figure 4.11: (Vinje's Gaussian Model) Traveltimes in the physical space.

sources. When two sheets connect to each other in the phase space, they connect along the caustic curves, which are shown in the figures; this can also be seen more clearly in Figure 4.13.

In this case, although u^4 cannot be simplified to 0, rays from the origin with initial $\psi = 0$ are still staying on the cross section $y = 0$. From the last equation in equation (4.11), if $\psi(z = 0) = 0$ and $y = 0$, which implies $c_y = 0$, we can obtain $\psi(z) \equiv 0$. Therefore, we can still compare our solution with the one obtained from ray tracing method on the cross section $y = 0$, as shown in Figure 4.13. Again, the solid line represents the solution using the ray tracing method. The solutions by the method presented here are plotted using circles. Two solutions match with each other very well. In all the sub-figures, we can see that the caustics develop and the traveltime becomes triple valued around caustics. To look at the solutions more closely, we concentrate on the solution for the source located at $(x_s, y_s) = (0, 0)$. More cross sections of the multivalued traveltime are plotted on

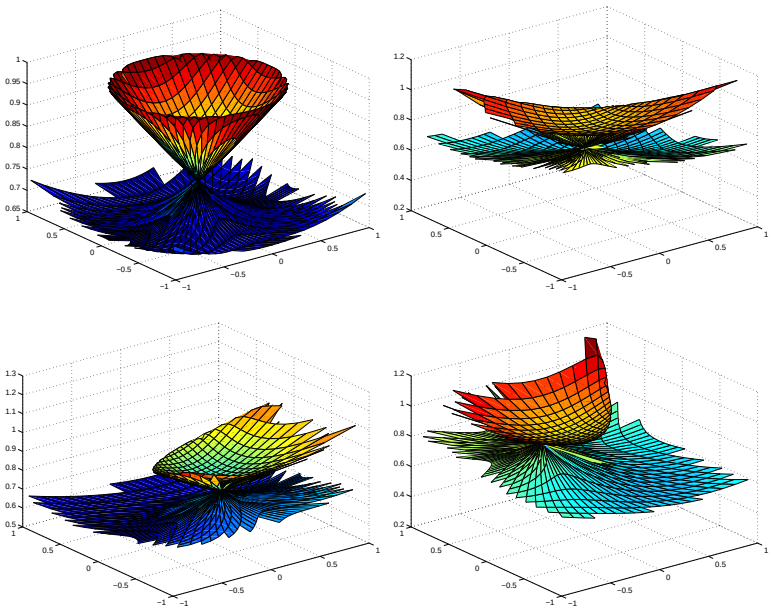


Figure 4.12: (Vinje's Gaussian Model) Traveltimes in the physical space using ray tracing method.

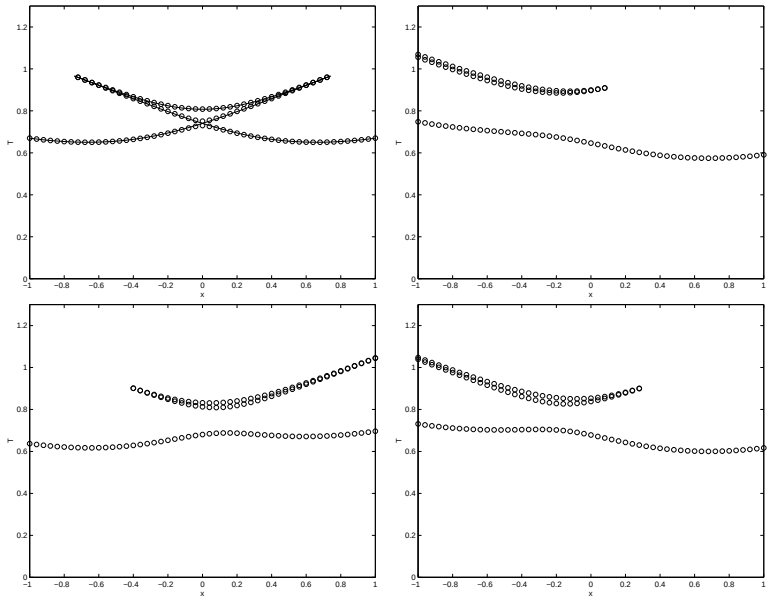


Figure 4.13: (Vinje's Gaussian Model) Traveltimes in the physical space. Solution using ray tracing method is plotted using solid line.

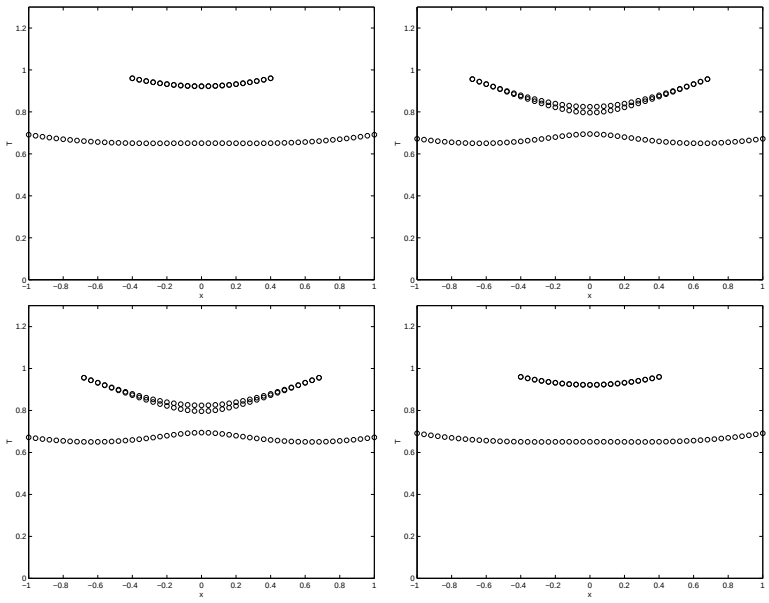


Figure 4.14: (Vinje's Gaussian Model) Traveltimes in the physical space on different cross sections.

Figure 4.14. The locations of the slides are $y = (j - 26)/25$ for $j = 11, 21, 31, 41$. One can see the obvious symmetry since the velocity has rotational invariance for z fixed.

We have also calculated amplitudes of the arrival rays. Figures 4.15 to 4.17 show the amplitude solutions corresponding to Figures 4.11, 4.13 to 4.14.

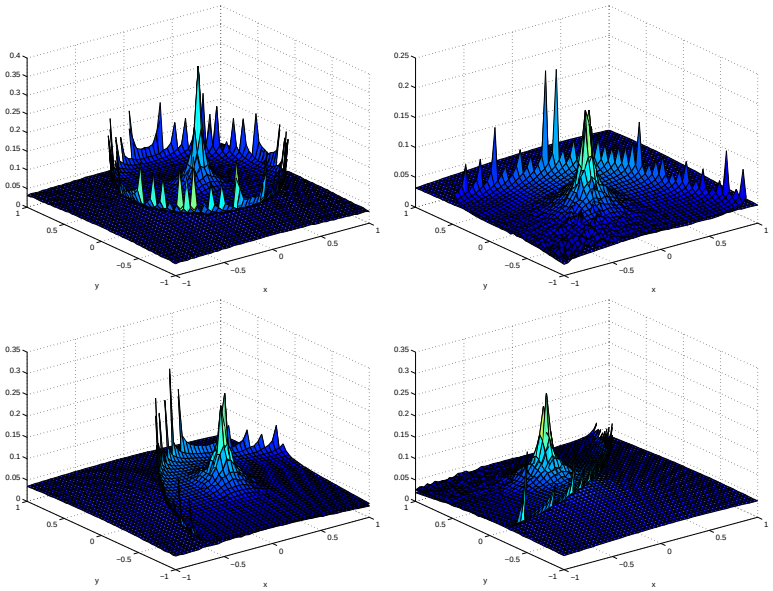


Figure 4.15: (Vinje's Gaussian Model) Amplitudes in the physical space.

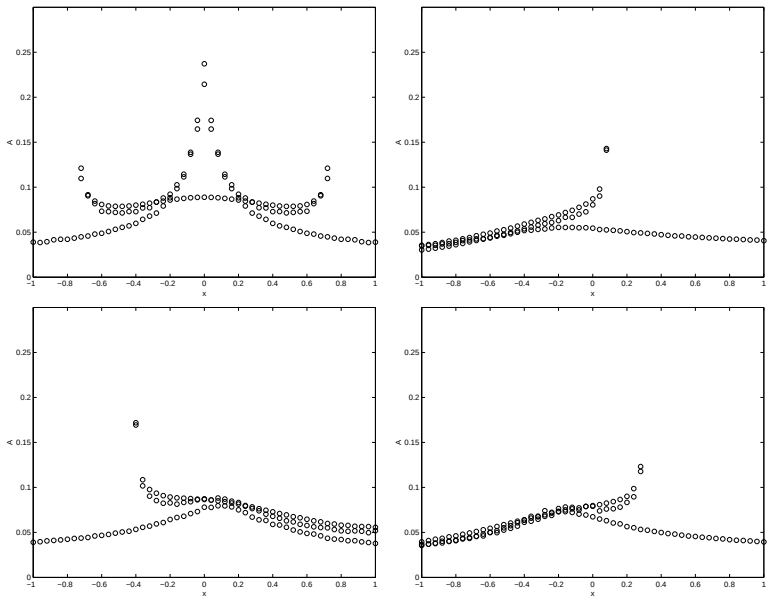


Figure 4.16: (Vinje's Gaussian Model) Amplitudes in the physical space on the cross section $y = 0$.

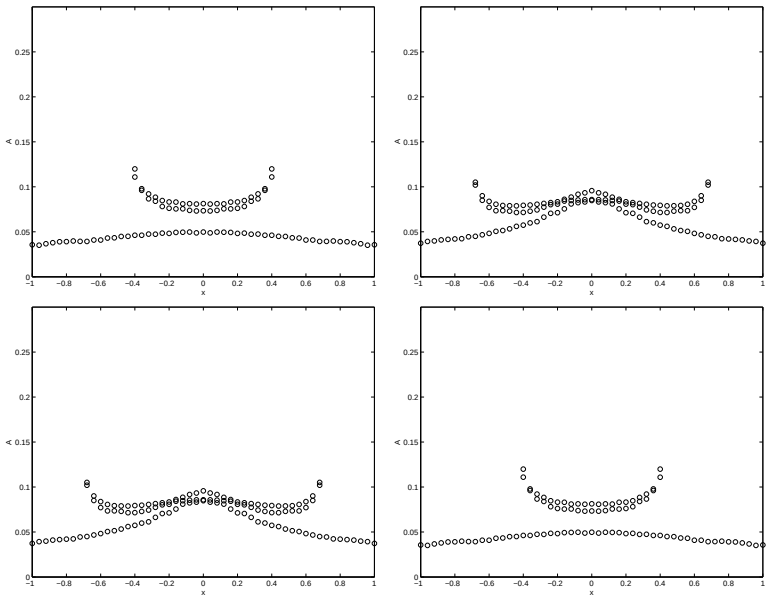


Figure 4.17: (Vinje's Gaussian Model) Amplitudes in the physical space on different cross sections.

CHAPTER 5

Transmission Traveltime Tomography Using First Arrivals

5.1 Introduction

The goal of transmission traveltime tomography is to estimate the wave-speed distribution from acoustic, seismic or electromagnetic (possibly multivalued) traveltime data. In seismics, this velocity analysis is often an important step in prospect evaluation in areas where lithology and structure undergo significant lateral change. In this chapter, we propose a new, robust and efficient tomography method based on only first arrival data. In the next chapter, we will discuss how we can incorporate multiple arrival data.

All traditional methods of traveltime tomography are directly based on the Fermat's least traveltime principle and they all bear a close link to the X-ray computerized tomography (CT) used in medical diagnosis. In medical CT the measured data are modeled by line integrals of wave amplitude attenuation for straight ray-paths passing through the body, and the Radon transform provides a foundation for medical CT. In seismics however, the ray-path curvature has to be taken into account in that lithology and structure usually have strong inhomogeneity, and the resulting ray-paths can strongly depend on the unknown wave speeds. To achieve such a purpose, ray-tracing based traveltime tomography

methods require very complicated data structure to trace curved rays through each pixel [11]; see [119] for 3-D examples. In addition, such ray-tracing based methods inevitably produce irregular ray coverage of the computational domain, and the resulting system of equations may not be well-conditioned [8, 9, 10]. In this paper we propose a PDE-based Eulerian approach to traveltime tomography so that we can avoid the cumbersome ray-tracing.

Recall that a necessary condition for Fermat's least traveltime principle to hold is characterized by the eikonal equation for traveltime [45], and the viscosity solution for the eikonal equation with a point-source condition is the least traveltime from the source to an arbitrary point connected by a shortest ray-path, as observed by [63, 70]. Because of its continuous dependence on the wave-speed distribution and source locations, the viscosity solution can be stably computed by various numerical schemes. In this chapter, we model traveltimes from a single source to multiple receivers by using the eikonal equation and propose a fast sweeping based adjoint state method for transmission traveltime tomography. This new approach not only overcomes some shortcomings inherited in the traditional ray-tracing based traveltime tomography but also enjoys quadratic convergence, thus it is very fast and robust.

In this chapter, we will only concentrate on first-arrival based traveltime tomography. We start from a mismatching functional between measured and simulated data and drive the functional to zero by a well designed limited memory BFGS (L-BFGS) optimization method. Although our approach shares some similarities with that in Sei-Symes [94, 95], that work was based on the paraxial formulation of the eikonal equation and only illustrated the feasibility of computing the traveltime gradient by using the adjoint state method. Instead of the paraxial formulation of the eikonal equation, we derive the gradient of the

mismatching functional directly from the steady eikonal equation by using the adjoint state method; furthermore, we apply the fast sweeping method [122, 111, 60] to solve the eikonal equation directly (the forward problem) and design a new fast sweeping method to solve the adjoint equation of the linearized eikonal equation (the adjoint problem) so that the required gradient can be computed highly efficiently; finally a limited memory BFGS optimization method drives the mismatching functional to zero with quadratic convergence.

5.2 Governing Equations

We start with the eikonal equation with a point source condition in an isotropic medium

$$|\nabla T| = \frac{1}{c} \tag{5.1}$$

with the point source condition

$$T(\mathbf{x}_s) = 0\,, \tag{5.2}$$

where $T(\mathbf{x})$ is the traveltime of wave from the source $\mathbf{x}_s$ to the point $\mathbf{x}$, and $c \in C^1(\Omega)$ is a positive velocity function.

For a given velocity model c, the viscosity solution of this equation can be computed efficiently by fast sweeping methods, and such solutions correspond to the least traveltime or the first-arrival traveltime according to [70].

In this work, we are interested in a related inverse problem, the so-called transmission traveltime tomography problem: given both the first-arrival traveltime measurements on the boundary $\partial\Omega_p$ and the location of the point source

$\mathbf{x}_s \in \Omega_p$, we want to invert for the velocity field $c(\mathbf{x})$ inside the domain Ω_p.

To achieve this, we propose inverting the velocity model by minimizing the following mismatching functional (energy),

$$E(c) = \frac{1}{2} \int_{\partial\Omega_p} |T - T^*|^2 \,, \tag{5.3}$$

where $T^*|_{\partial\Omega_p}$ is the measurement and $T|_{\partial\Omega_p}$ is computed by solving (5.1) with a point source condition (5.2). In other words, this energy measures the L^2-difference between the experimental measurement, T^*, and the solution from the eikonal equation, T, on the boundary of the computational domain.

To minimize this energy, we use the method of gradient descent. We first perturb the velocity field c by $\epsilon\tilde{c}$, which causes a corresponding change in T by $\epsilon\tilde{T}$. The change in the energy is then given by

$$\delta E = \epsilon \int_{\partial\Omega_p} \tilde{T}(T - T^*) + O(\epsilon^2) \,. \tag{5.4}$$

From the state equation (5.1), the perturbations in c and T are related by

$$T_x\tilde{T}_x + T_y\tilde{T}_y + T_z\tilde{T}_z = -\frac{\tilde{c}}{c^3} \,. \tag{5.5}$$

We need to determine the perturbation in c, $\tilde{c}$, so as to decrease the energy $E(c)$. The main difficulty is that the perturbation in E, δE, depends on $\tilde{c}$ implicitly through $\tilde{T}$ and the partial differential equation (5.5). To efficiently compute $\tilde{c}$ which minimizes E, we apply the adjoint state method.

Multiplying (5.5) by $\epsilon\lambda$, integrating it over Ω_p, applying integration by parts,

and adding the resulting expression to (5.4), we have

$$\begin{aligned}\frac{\delta E}{\epsilon} &= \int_{\partial\Omega_p} \tilde{T}(T-T^*) + \int_y\int_z \lambda T_x \tilde{T}|_{x_{\min}}^{x_{\max}} + \int_x\int_z \lambda T_y \tilde{T}|_{y_{\min}}^{y_{\max}} + \int_x\int_y \lambda T_z \tilde{T}|_{z_{\min}}^{z_{\max}} \\ &\quad - \int_{\Omega_p} \tilde{T}[(\lambda T_x)_x + (\lambda T_y)_y + (\lambda T_z)_z] + \int_{\Omega_p} \frac{\tilde{c}\lambda}{c^3} + O(\epsilon)\,. \end{aligned} \tag{5.6}$$

Next, we choose λ satisfying

$$[(-T_x)\lambda]_x + [(-T_y)\lambda]_y + [(-T_z)\lambda]_z = 0\,, \tag{5.7}$$

with the boundary condition,

$$(\mathbf{n}\cdot\nabla T)\lambda = T^* - T, \tag{5.8}$$

on the boundary $\partial\Omega_p$, where $\mathbf{n}$ is the unit outward normal of the boundary. By introducing this adjoint state equation, one can eliminate the dependence of $\tilde{T}$ when determining the gradient of E with respect to c.

Ignoring all higher than linear order terms in the energy perturbation, we have

$$\frac{\delta E}{\epsilon} = \int_{\Omega_p} \frac{\tilde{c}\lambda}{c^3}\,. \tag{5.9}$$

To minimize the energy using the method of gradient descent, one could choose the perturbation $\tilde{c} = -\lambda/c^3$. This implies

$$\delta E = -\epsilon \int_{\Omega_p} \tilde{c}^2 \leq 0 \tag{5.10}$$

and the equality holds when $||\tilde{c}||_{H^0(\Omega_p)} = 0$. However, it is not straight-forward how one can guarantee the following two properties,

1. $\tilde{c}^k|_{\partial\Omega_p} = 0$;

2. $c^{k+1} = c^k + \epsilon\tilde{c}^k$ smooth.

The first condition assumes that we can measure c on the boundary $\partial\Omega_p$, denoted by $c^*|_{\partial\Omega_p}$, which is a reasonable assumption. This means that the variations of the velocity function on the boundary should be zero.

The second condition is a regularity condition on c^k. This regularity seems to be too restrictive in practice. In general, one only needs $c^k \in C^1$ to guarantee well-posedness of the state equation (5.1). However, assuming that one uses $\tilde{c}^k = -\lambda/c^3$ directly, it is not clear whether this function would give us the desired regularity. Even if this perturbation is in C^1, the numerical solution may have jumps or spikes. These irregularities will force one to pick a very small step-size, ϵ^k, in the minimization process. Therefore, to have faster convergence, we impose the above regularity in each iteration.

One way to satisfy both properties is to use the descent direction

$$\tilde{c} = -(I - \nu\Delta)^{-1}\left(\frac{\lambda}{c^3}\right), \tag{5.11}$$

where I is the identity operator, Δ is the Laplacian operator and $\nu \geq 0$ controls the amount of regularity that one wants. The homogeneous boundary condition is imposed in inverting the operator $(I - \nu\Delta)$. With this particular $\tilde{c}$, we have

$$\delta E = -\epsilon \int_{\Omega_p} (\tilde{c}^2 + \nu|\nabla\tilde{c}|^2) \leq 0\,. \tag{5.12}$$

We notice that this process amounts to seeking updates in some weighted Sobolev space in the case $\nu > 0$. Then the above equality holds when $\|\tilde{c}\|_{H^1_\nu(\Omega_p)} = 0$.

In the above calculation, we use the first-arrivals at different receivers associ-

ated with only a single point source. If we perform multiple such experiments, namely, we have many such data sets, then those can be easily incorporated into the formulation. For example, we can assume that there are N point sources located at $\mathbf{x}_s^i$, for $i = 1, \cdots, N$, and we also have N sets of first-arrival traveltime measurements T_i^* associated with these N sources. Then we can simply define a new energy

$$E^N(c) = \frac{1}{2} \sum_{i=1}^{N} \int_{\partial\Omega_p} |T_i - T_i^*|^2 \,, \tag{5.13}$$

where T_i is the solution from the eikonal equation with the corresponding point source condition $T(\mathbf{x}_s^i) = 0$. Utilizing the same approach as above, we have the following perturbation in the energy

$$\frac{\delta E^N}{\epsilon} = \int_{\Omega_p} \frac{\tilde{c}}{c^3} \sum_{i=1}^{N} \lambda_i \,, \tag{5.14}$$

where λ_i is the adjoint variable of T_i satisfying

$$\{[-(T_i)_x]\lambda_i\}_x + \{[-(T_i)_y]\lambda_i\}_y + \{[-(T_i)_z]\lambda_i\}_z = 0 \,, \tag{5.15}$$

with the boundary condition,

$$(\mathbf{n} \cdot \nabla T_i)\lambda_i = T_i^* - T_i \,, \tag{5.16}$$

for $i = 1, \cdots, N$.

Consequently, we can choose the following gradient direction to minimize the energy $E^N(c)$,

$$\tilde{c} = -(I - \nu\Delta)^{-1} \left(\frac{1}{c^3} \sum_{i=1}^{N} \lambda_i \right) . \tag{5.17}$$

We remark that the above updating procedure is similar to the so-called simul-

taneous iterative reconstruction technique frequently used in medical imaging; it is also possible to adopt the algebraic reconstruction type technique as used in [34] to update the velocity.

5.3 Algorithm and Numerical Implementations

5.3.1 Tomography Algorithm

Here we give an algorithm for this tomography problem.

Tomography Algorithm:

1. Initialize c^k for $k = 0$ by solving

$$(I - \nu\Delta)c^0 = 0\,, \tag{5.18}$$

   with the boundary condition $c^0|_{\partial\Omega} = c_{\text{exact}}|_{\partial\Omega}$.

2. Compute $T(x, z)$ by solving (5.1) with the point source condition (5.2) using $c = c^k$.

3. Compute $\lambda(x, z)$ by solving (5.7) with the boundary condition (5.8).

4. Determine $\tilde{c}^k$ using (5.11).

5. Determine ϵ^k using, for example, the Armijo-Goldstein rule or simply $\epsilon^k = \epsilon$.

6. Update

$$c^{k+1} = c^k + \epsilon^k \tilde{c}^k\,.$$

7. Go back to Step 2 until $||\tilde{c}^k(x, z)||_2 \leq \delta$ or $k \geq k_{\max}$, where δ and $k_{\max}$ are given convergence parameters.

To start the iteration, we need to initialize c^0. In the algorithm, we assume that we can measure the velocity at receivers, giving $c^0|_{\partial\Omega} = c_{\text{exact}}|_{\partial\Omega}$. This condition can of course be replaced by other assumptions. In practice, due to the nonlinearity in the problem, different initial guesses will generally converge to different energy minimizers. This non-uniqueness can be overcome by some a priori knowledge of the model. For example, the above assumption can be relaxed by replacing the Dirichlet conditions on both the left and right boundaries with the Neumann boundary conditions $\partial c^0/\partial x|_{x=x_{\min}} = \partial c^0/\partial x|_{x=x_{\max}} = 0$.

We can speed up the convergence by replacing the gradient descent method with BFGS-type iterations. To solve the elliptic equation in Step 4, we use the FFT. In Step 2 and Step 3, both the equations (5.1) and (5.7) can be solved by the fast sweeping method [122, 111, 60].

5.3.2 Fast Sweeping Method for Equation (5.1)

The fast sweeping method was originated in Boue and Dupis [12], its first PDE formulation was in implicit and non-parametric shape reconstruction from unorganized points using a variational level set method [124]; Zhao [122] proved the $O(N)$ convergence of the method for the eikonal equation based on the Godunov Hamiltonian on Cartesian meshes; later on, the fast sweeping method was extended to treat Hamilton-Jacobi equations with convex Hamiltonians based on the Godunov Hamiltonian [111] and handle Hamilton-Jacobi equations with non-convex Hamiltonians based on the Lax-Friedrichs Hamiltonian [60]; see [111, 60] and references therein for the fast sweeping method on Cartesian meshes and [90] for the method on triangulated meshes. Certainly, one may also use other methods such as the fast marching method [97].

To be self-contained, we give a short summary of the fast sweeping method

for eikonal equations. To avoid cluttered notations we present the algorithm for the 2-D case only; see [122] for more details.

First we discretize the rectangular domain $\Omega \subset \mathcal{R}^2$ into a uniform mesh with mesh points $\mathbf{x}_{i,j}$ and mesh sizes $\Delta x = \Delta z = h$, and we denote the numerical solution at $\mathbf{x}_{i,j}$ by $T_{i,j}$. Applying the Godunov numerical Hamiltonian to the eikonal equation, for $i = 2, \cdots, I-1$, $j = 2, \cdots, J-1$, we have

$$[(T_{i,j} - T_{xmin})^+]^2 + [(T_{i,j} - T_{zmin})^+]^2 = \frac{h^2}{c_{i,j}^2}\,, \tag{5.19}$$

where $T_{xmin} = \min(T_{i-1,j}, T_{i+1,j})$, $T_{zmin} = \min(T_{i,j-1}, T_{i,j+1})$ and $(x)^+$ denotes the positive part of x. At the boundary of the computational domain one sided difference is used.

Fast Sweeping Algorithm:

1. Initialize the point source condition $T(\mathbf{x}_s) = 0$ by assigning the exact value if $\mathbf{x}_s$ is a mesh point, or assigning to grid points near $\mathbf{x}_s$ exact values which are computed by using the constant velocity at the point source. These values are fixed in later iterations. Assign larger positive values at all other grid points, and these values will be updated later.

2. Update the solution by Gauss-Seidel iterations with alternating sweeping. At each grid point $\mathbf{x}_{i,j}$ whose value was not fixed during the initialization, compute the candidate solution, denoted by $\bar{T}$ of (5.19) from the current values of its neighbors $T_{i\pm1,j}$, $T_{i,j\pm1}$ and then update $T_{i,j}$ to be the smaller one between $\bar{T}$ and its current value; i.e., $T_{i,j}^{new} = \min(T_{i,j}^{old}, \bar{T})$. We sweep the whole domain with four alternate ordering repeatedly: $i = 1:I, j = 1:J$; $i = 1:I, j = J:1$; $i = I:1, j = 1:J$; $i = I:1, j = J:1$. Here i and j are the running

indices along x and y directions.

3. Test the convergence: given convergence criterion $\epsilon > 0$, check whether $\|T^{n+1} - T^n\|_{L^1} \leq \epsilon$.

We remark that the sweeping strategy can be used for more general Hamilton-Jacobi equations as long as an efficient local solver is available at each grid point, so that an iterative procedure is well defined at each local grid point.

5.3.3 Fast Sweeping Method for Equation (5.7)

Next we design a fast sweeping method for equation (5.7). Once again to simplify the notation, we give a 2-D formulation only; the extension to a 3-D formulation is straightforward.

The adjoint state equation (5.7) can be written in the following form

$$(a\lambda)_x + (b\lambda)_z = 0\,, \tag{5.20}$$

where a and b are given functions of (x, z).

Considering a computational cell centered at (x_i, z_j) and discretizing the equation in conservation form, we have

$$\frac{1}{\Delta x}\ a_{i+1/2,j}\lambda_{i+1/2,j} - a_{i-1/2,j}\lambda_{i-1/2,j}\ + \frac{1}{\Delta z}\ b_{i,j+1/2}\lambda_{i,j+1/2} - b_{i,j-1/2}\lambda_{i,j-1/2}\ = 0\,. \tag{5.21}$$

The values of λ on the interfaces, $\lambda_{i\pm1/2,j}$ and $\lambda_{i,j\pm1/2}$, are determined according to the propagation of characteristics. In the case when $a_{i+1/2,j} > 0$, the characteristic for determining λ goes from the left hand side of the interface to the right hand side, and this suggests that we use the value $\lambda_{i,j}$ to define $\lambda_{i+1/2,j}$; otherwise, we have $\lambda_{i+1/2,j} = \lambda_{i+1,j}$. The terms $\lambda_{i,j\pm1/2}$ can be defined in a similar way.

Introducing the following notations

$$a^{\pm}_{i+1/2,j} = \frac{a_{i+1/2,j} \pm |a_{i+1/2,j}|}{2} \quad , \quad a^{\pm}_{i-1/2,j} = \frac{a_{i-1/2,j} \pm |a_{i-1/2,j}|}{2} \,,$$
$$b^{\pm}_{i,j+1/2} = \frac{b_{i,j+1/2} \pm |b_{i,j+1/2}|}{2} \quad \text{and} \quad b^{\pm}_{i,j-1/2} = \frac{b_{i,j-1/2} \pm |b_{i,j-1/2}|}{2} \,,$$

we have

$$\frac{1}{\Delta x}\Big((a^{+}_{i+1/2,j}\lambda_{i,j} + a^{-}_{i+1/2,j}\lambda_{i+1,j}) - (a^{+}_{i-1/2,j}\lambda_{i-1,j} + a^{-}_{i-1/2,j}\lambda_{i,j})\Big) +$$
$$\frac{1}{\Delta z}\Big((b^{+}_{i,j+1/2}\lambda_{i,j} - b^{-}_{i,j+1/2}\lambda_{i,j+1}) - (b^{+}_{i,j+1/2}\lambda_{i,j-1} - b^{-}_{i,j+1/2}\lambda_{i,j})\Big) = 0 \,, \quad (5.22)$$

which can be rewritten as

$$\left(\frac{a^{+}_{i+1/2,j} - a^{-}_{i-1/2,j}}{\Delta x} + \frac{b^{+}_{i,j+1/2} - b^{-}_{i,j-1/2}}{\Delta z}\right)\lambda_{i,j} = \frac{a^{+}_{i-1/2,j}\lambda_{i-1,j} - a^{-}_{i+1/2,j}\lambda_{i+1,j}}{\Delta x} + \frac{b^{+}_{i,j-1/2}\lambda_{i,j-1} - b^{-}_{i,j+1/2}\lambda_{i,j+1}}{\Delta z} \,. \quad (5.23)$$

This gives an expression to build up a fast sweeping-type iterative method.

To apply this iterative scheme to equation (5.7), we need to specify the function values of a and b not at the cell centers (x_i, z_j), but on the cell interfaces $(x_{i\pm1/2}, z_j)$ and $(x_i, z_{j\pm1/2})$. This can be done easily using central differences. For example, we have $a_{i+1/2,j} = -(T_{i+1,j} - T_{i,j})/\Delta x$ and $a_{i-1/2,j} = -(T_{i,j} - T_{i-1,j})/\Delta x$. In addition, we have to incorporate the boundary condition (5.8) into the above linear system for λ as well. Then we can show that the coefficient matrix of the resulting linear system for λ is irreducibly diagonally dominant, therefore the alternating symmetrical Gauss-Seidel iteration converges.

Fast Sweeping Algorithm for equations (5.7) and (5.8):

1. On the boundary, compute $(\mathbf{n}\cdot\nabla T)$ from the solution of the eikonal solver using

one side difference. Next, compute the boundary condition for λ according to (8). These values will be fixed in the following computations.

2. Update $\lambda_{i,j}$ at the interior points according to (22). As in the fast sweeping method for (1), we sweep the whole domain with four alternate orderings.

3. For some given convergence criterion $\epsilon > 0$, repeat 2 until $||\lambda^{n+1} - \lambda^n||_{L^1} \leq \epsilon$.

We point out that the above fast sweeping method is different from the fast marching method used in [43], in that our method is iterative and the one in [43] is constructive based on upwinding properties.

5.3.4 L-BFGS Method

In the tomography algorithm above, we update the approximation to the velocity by the typical gradient descent method, where

$$c^{k+1} = c^k - \epsilon^k \tilde{c}^k \,. \tag{5.24}$$

Although it is simple to implement, the method is not efficient because it takes a large number of iterations to converge to the steady state solution.

To speed up the convergence, we can apply the quasi-Newton method defined by

$$c^{k+1} = c^k + \epsilon^k s^k \,, \tag{5.25}$$

where $s^k = -A_k^{-1} E'(c^k)$ and A_k is a positive definite operator satisfying the secant condition

$$A_{k+1}(c^{k+1} - c^k) = E'(c^{k+1}) - E'(c^k) \,. \tag{5.26}$$

In this iteration, the operator A_{k+1} is updated by modifying the previous operator

A_k.

One possible way to modify this operator is given by the Broydon-Fletcher-Goldfarb-Shanno (BFGS) procedure,

$$A_k v = A_{k-1} v + \alpha < p, v > p + \beta < q, v > q \,, \tag{5.27}$$

where

$$\begin{aligned} p = y/||y|| \quad , \quad q = A_{k-1}s/||A_{k-1}s|| \,, \\ \alpha = ||y||^2/ < y, s > \quad , \quad \beta = -||A_{k-1}s||^2/ < s, A_{k-1}s > \end{aligned} \tag{5.28}$$

with $s = c^k - c^{k-1}$, $y = E'(c^k) - E'(c^{k-1})$ and $A_0 = I$.

However, in practice, the condition number of A_k can be increased significantly throughout the iteration, which makes the computation inaccurate. To alleviate this, one can modify the iteration using the limited memory BFGS (L-BFGS) given by

$$A_k v = v + \sum_{j=k-L+1}^{k} \left(\alpha_j < p_j, v > p_j + \beta_j < q_j, v > q_j \right) . \tag{5.29}$$

In this paper, we adapt the L-BFGS-B code from [16]. This code requires user to provide only subroutines to compute both the energy to be minimized and the gradient of this energy. The step size ϵ^k is automatically determined.

5.4 Two-Dimensional Numerical Examples

In the following examples, we use 129×129 grid points in the x-z space. Using the above formulation, we need measurements, denoted by ϕ^* and T^*, on the boundary $\partial\Omega_p$. If the point source is located inside Ω, the characteristics of

the eikonal equation always flow out from the domain. Therefore, in synthetic experiment the boundary measurements can be obtained by solving the equation (5.1) directly using the fast sweeping method together with the exact velocity c.

For each velocity model below, we have implemented the following two cases - one source and ten sources. For the one source case, we use the boundary measurements from the only point source located at $(x, z) = (0, 0.1)$. In the cases with ten point sources, we use nine more sets of boundary measurements, and these correspond to source locations at $(x, z) = (\pm 0.25, 0.1)$, $(\pm 0.5, 0.1)$, $(0, 1.9)$, $(\pm 0.25, 1.9)$ and $(\pm 0.5, 1.9)$, respectively. However, to save some space we only present the results corresponding to the case of ten sources.

To start the algorithm, we initialize the velocity c^0 by solving the above elliptic equation (5.18) with $\nu = 1$.

5.4.1 Constant Model

The exact velocity model is given by $c \equiv 1$. We use the BFGS method to invert for the velocity. The results are shown in Figure 5.1. As we can see, the recovered velocity is almost exact, the relative error is almost negligible, and we observe the typical quadratic convergence of the algorithm due to the L-BFGS method.

5.4.2 Waveguide Model

The exact velocity model is given by

$$c(x, z) = 3 - 2.5 \exp\left(-\frac{x^2}{2}\right) . \tag{5.30}$$

We apply the BFGS method to invert for the velocity. Figure 5.2 shows the relative error at convergence and the convergence history. In Figure 5.3, we show

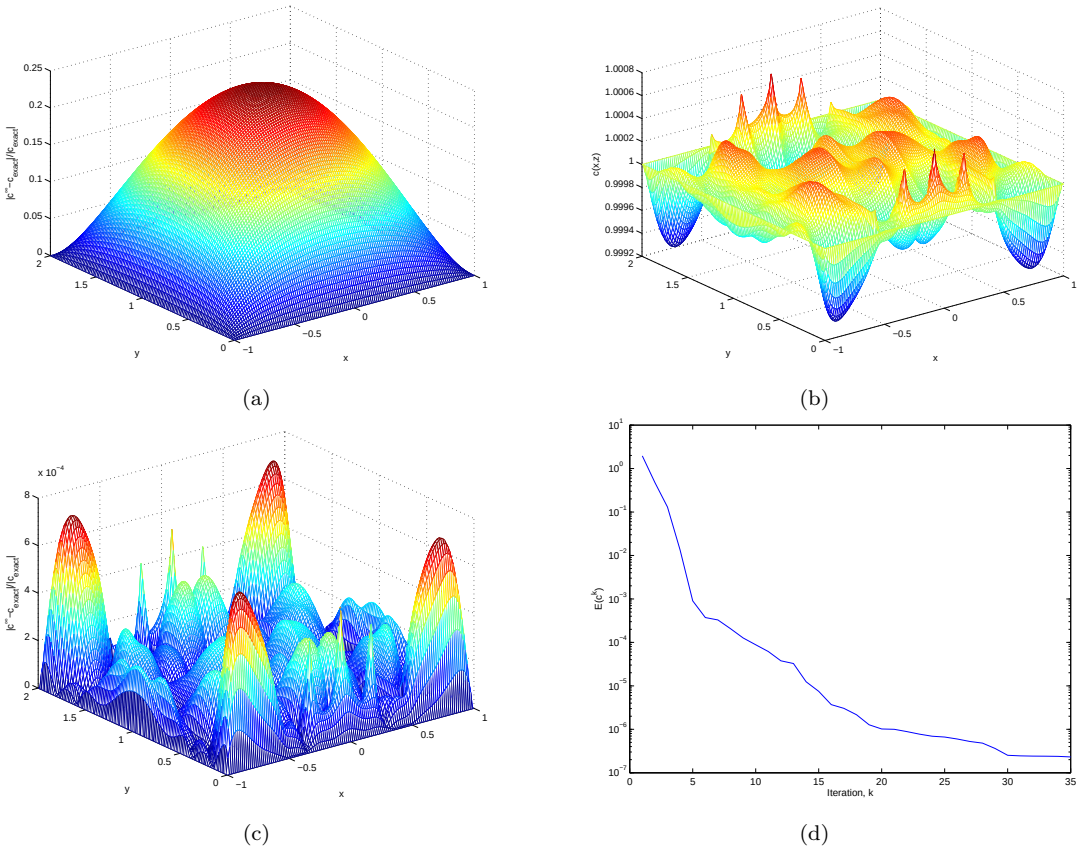
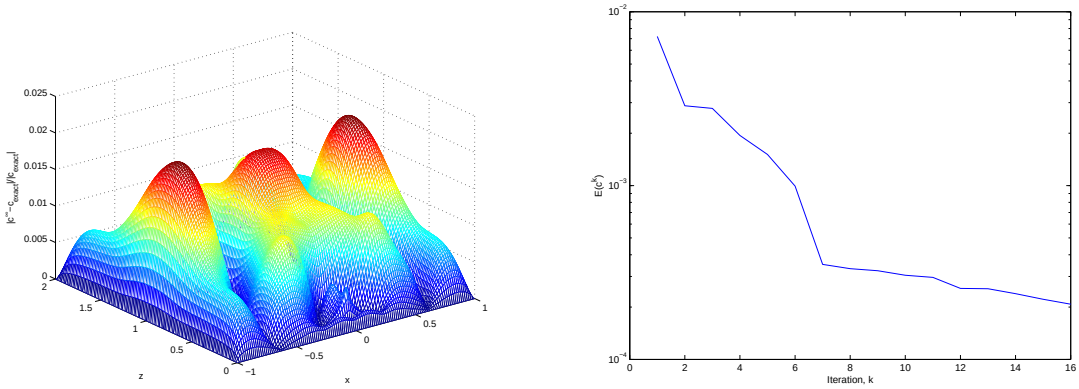


Figure 5.1: (Constant Model. Ten Sources.) BFGS. (a): the initial guess; (b): final approximated c; (c): the relative error in the solution; (d): the convergence history of energy.



(a) (b)

Figure 5.2: (Waveguide Model. Ten Sources.) (a): the relative error in the solution and (b): the convergence history of energy.

slices of the cross-sections of the solution along $z = 1$ and $x = 0$. As we can see, we are able to recover the velocity as well.

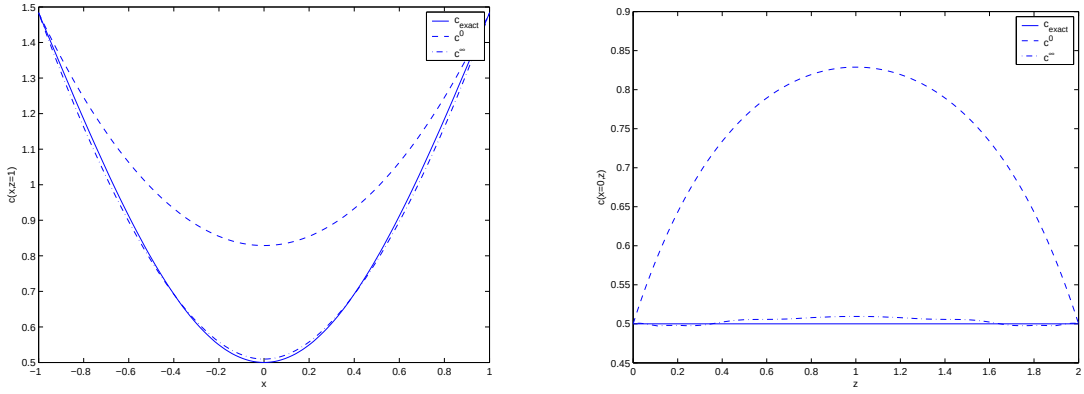
5.4.3 Gaussian Model

The exact velocity model is given by

$$c(x,z) = 3 - \frac{1}{2}\exp\left(-\frac{x^2+(z-0.5)^2}{0.5^2}\right) - \exp\left(-\frac{x^2+(z-1.25)^2}{0.5^2}\right). \quad (5.31)$$

In Figure 5.4, we show the convergent velocity and the convergence history of the algorithm; once again, we observe quadratic convergence. In Figure 5.5, we show slices of cross-sections of the final converged velocity; as we can see, they fit well with the exact velocity.

To further test the algorithm, we repeat the experiment but perturb the synthetic data T^* with some noise. Using the same velocity model, we first compute the traveltime on the boundary of the domain. These measurements are added 5% Gaussian noise with zero mean. Figures 5.6 and 5.7 show that we have ro-



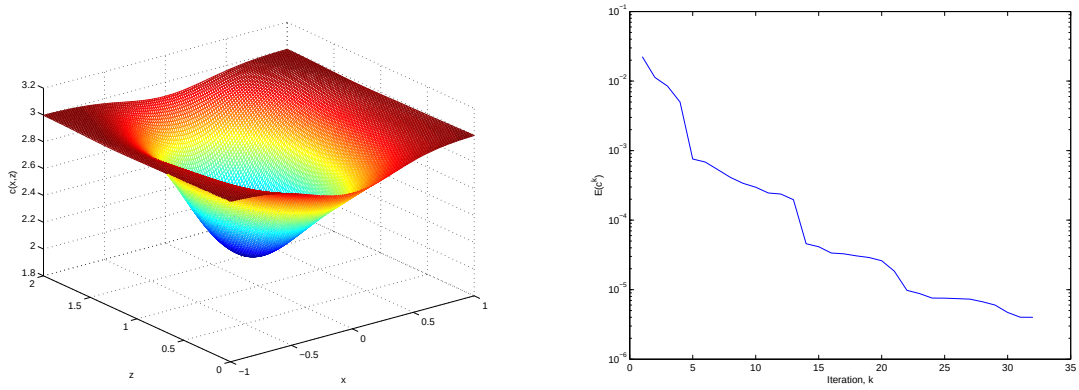
(a) (b)

Figure 5.3: (Waveguide Model. Ten Sources.) Cross-sections of the solutions. (a): $z = 1$ and (b): $x = 0$.

bust convergence as well. As shown in Figure 5.6(b), we are not able to drive the energy to zero. This is expected because the boundary measurements are highly oscillatory, and in general we cannot find a smooth velocity c which produces exactly the same traveltimes as those noisy data.

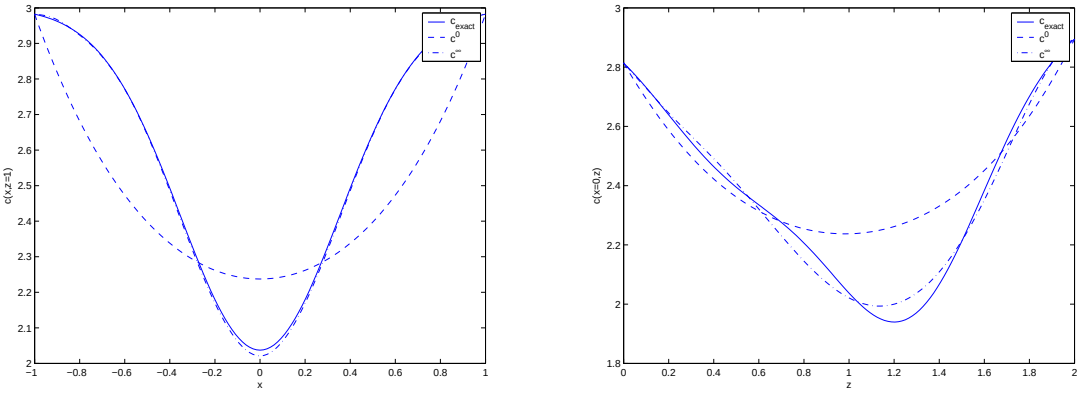
5.5 Three-Dimensional Numerical Examples

In the following examples, we use $65 \times 65 \times 65$ grid points in the three-dimensional space. Using the above formulation, we need measurements, denoted by T^*, on the boundary $\partial\Omega_p$. For each velocity model shown below, we have implemented the following case, 49 sources on the levels $z = 0.1$ and $z = 1.9$, and we have 98 sets of measurements in total. To start the algorithm, we initialize the velocity c^0 by solving the elliptic equation (5.18) with $\nu = 1$.



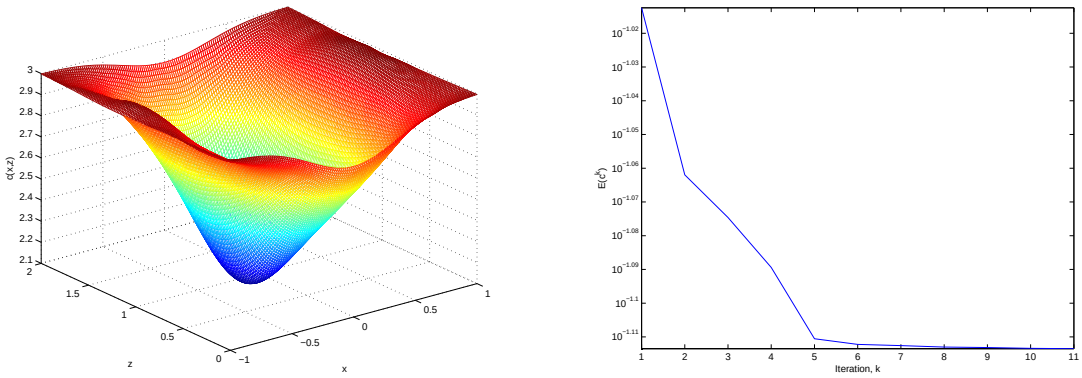
(a) (b)

Figure 5.4: (Gaussian Model. Ten Sources.) BFGS. (a): the final approximated c and (b): the convergence history of energy.



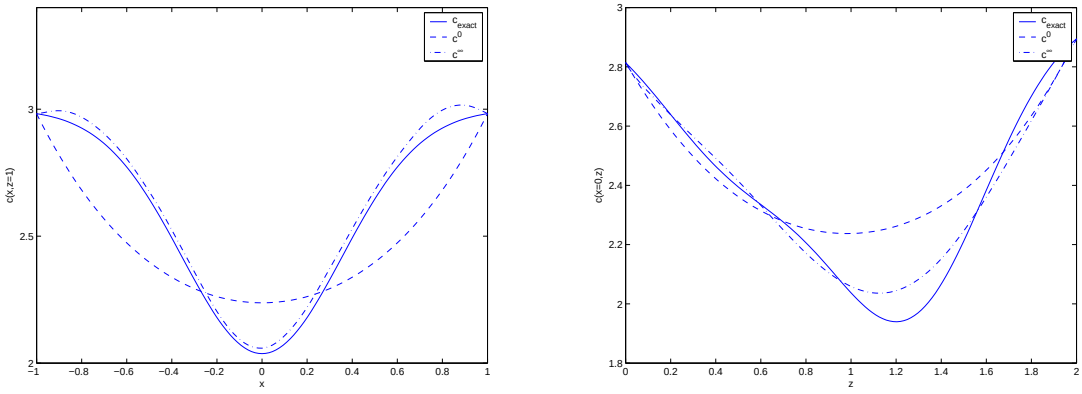
(a) (b)

Figure 5.5: (Gaussian Model. Ten Sources.) BFGS. Cross-sections of the solutions. (a): $z = 1$ and (b): $x = 0$.



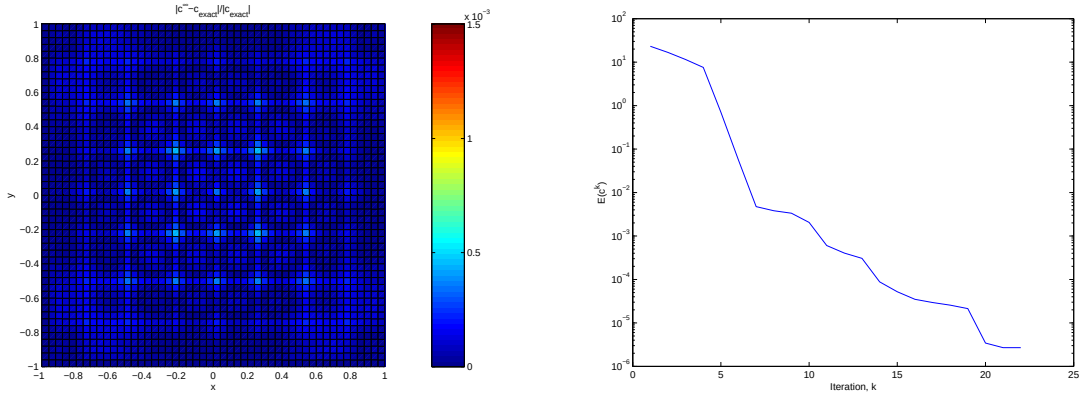
(a) (b)

Figure 5.6: (Gaussian Model with added noise. Ten Sources.) BFGS. (a): the final approximated c; (b): the convergence history of energy.



(a) (b)

Figure 5.7: (Gaussian Model with added noise. Ten Sources.) BFGS. Cross-sections of the solutions: (a): $z = 1$ and (b): $x = 0$.



(a) (b)

Figure 5.8: (Constant Model. 98 Sources.) 3-D case. (a): the relative error in the solution on the cross-section $z = 1$ and (b): the convergence history of energy.

5.5.1 Constant Model

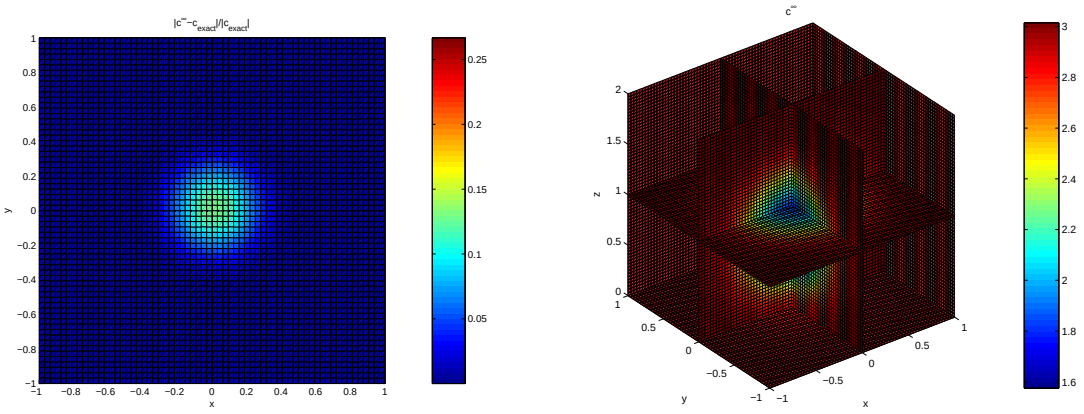
The exact velocity model is given by $c \equiv 1$. We use the BFGS method to invert for the velocity. The results are shown in Figure 5.8; we observe the quadratic convergence once again.

5.5.2 Gaussian Model

The exact velocity model is given by

$$c(x, y, z) = 3 - \frac{1}{2} \exp\left(-\frac{x^2 + y^2 + (z - 0.5)^2}{0.5^2}\right) - \exp\left(-\frac{x^2 + y^2 + (z - 1.25)^2}{0.5^2}\right). \tag{5.32}$$

We use the gradient descent method to invert for the velocity. The results are shown in Figure 5.9.



(a) (b)

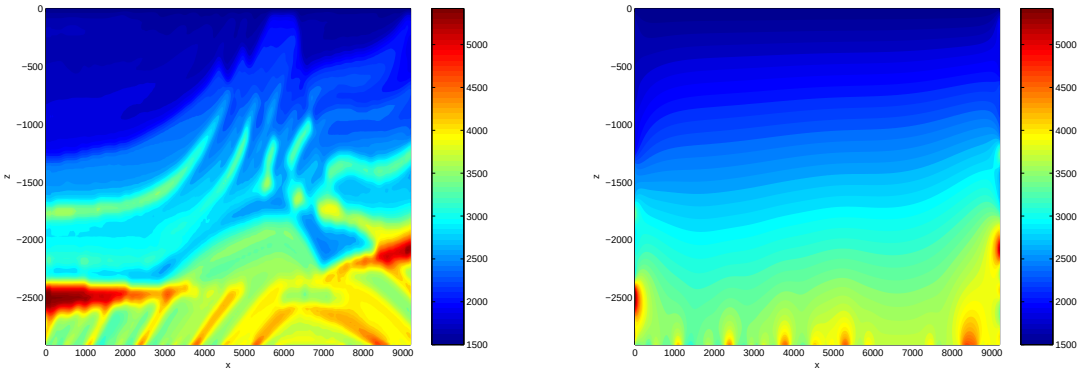
Figure 5.9: (Gaussian Model. 98 Sources.) 3-D case. (a): the relative error in the solution on the cross-section $z = 1$ and (b): the final approximated c.

5.6 Synthetic Marmousi Model

We use twenty sources and their (x, z)-coordinates are (200, 2800), (1000: 1000: 9000, 2800), (200, 100) and (1000: 1000: 9000, 100), respectively, where we have used by now the standard Matlab colon notation.

The true Marmousi velocity model is illustrated in Figure 5.10(a). As we can see, this velocity model has high contrast with variations of different scales. On the one hand, since the fast sweeping method used here is unconditionally stable, the forward eikonal solver will not have difficulty in computing traveltime to first order accuracy. On the other hand, viscosity-solution based first-arrival traveltimes will not be able to give us too much information about variations of small scales occurring in the velocity model; to retain the information related to small scales, we have to use multiple arrivals, which in turn calls for multiple-arrival based traveltime tomography. In this regard, for computing multiple arrivals of the Marmousi model in the Eulerian framework, see chapters 2-4 for more.

To start the algorithm, we initialize the velocity c^0 by solving the Laplace



(a) (b)

Figure 5.10: (Marmousi model) (a): the true velocity distribution and (b): the initial profile c^0.

equation $-\Delta c^0 = 0$ with $c^0|_{\partial\Omega} = c_{\text{exact}}|_{\partial\Omega}$. The solution is plotted in Figure 5.10(b).

We use the BFGS method to invert for the velocity. Figure 5.11 presents the inversion results for different cases in terms of the sampling size Δx and the parameter ν.

Comparing Figure 5.11(a) with the true model Figure 5.10(a), we have successfully imaged the macro scale variations of the velocity model, but not able to compute those finer scale variations of the velocity model which does exist in the true velocity model. However, transmission tomography usually has very limited resolution, and we believe that this result is near optimal using the current approach.

To confirm this, we refine the velocity model by doubling the number of grid points in each direction while keeping the regularization parameter ν fixed; the corresponding solution is shown in Figure 5.11(d). We also check the following

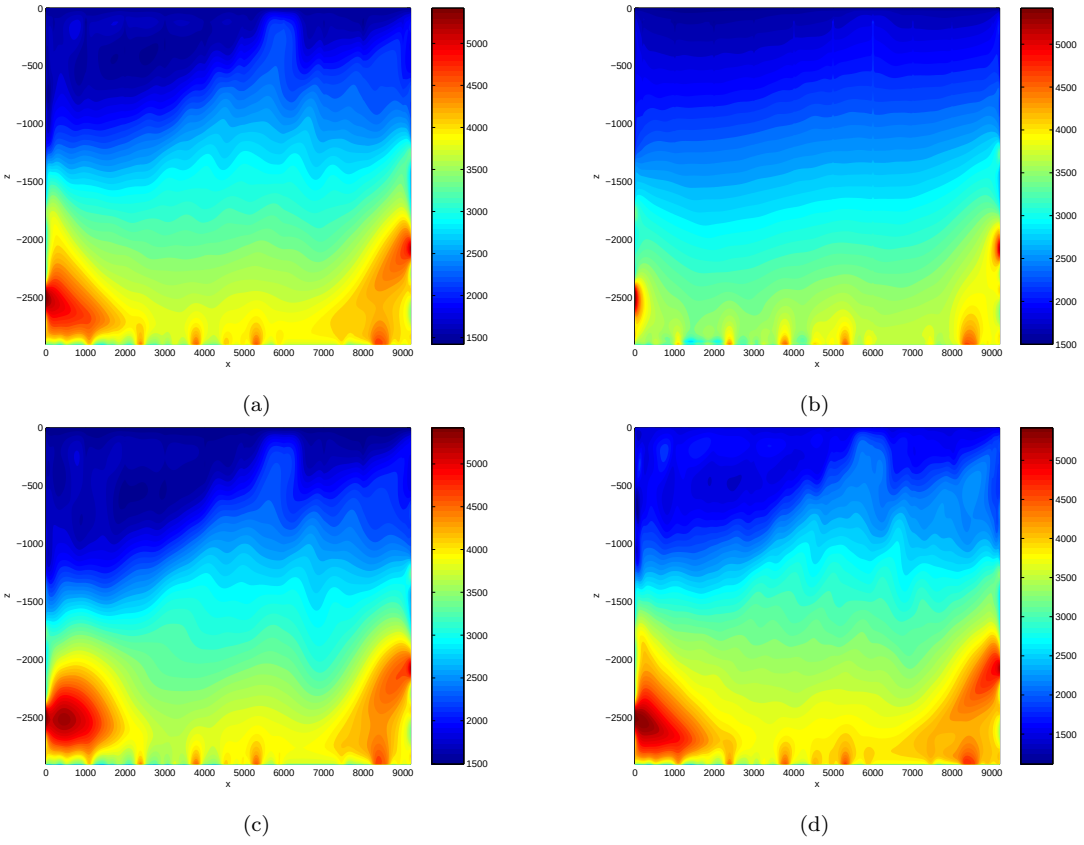
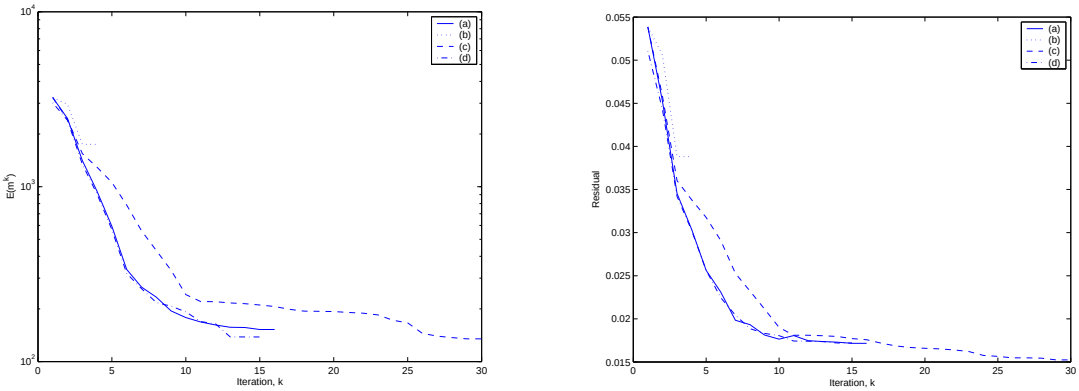


Figure 5.11: (Marmousi model) Converged solutions. (a): $\nu = 10^4$ and $\Delta x = 24$; (b): $\nu = 10^2$ and $\Delta x = 24$; (c): $\nu = 10^6$ and $\Delta x = 24$; (d): $\nu = 10^4$ and $\Delta x = 12$.



(I) (II)

Figure 5.12: (Marmousi model) The change in (I): the energy and (II): the residual. Legend: (a): $\nu = 10^4$ and $\Delta x = 24$; (b): $\nu = 10^2$ and $\Delta x = 24$; (c): $\nu = 10^6$ and $\Delta x = 24$; (d): $\nu = 10^4$ and $\Delta x = 12$.

residual in the solution defined by

$$R = \frac{\Sigma_{i=1}^{N} \int_{\partial\Omega} |T_i - T_i^*|/T_i^* ds}{N \int_{\partial\Omega} ds} \tag{5.33}$$

where N is the number of sources defined above. This quantity essentially is the average relative error in the first-arrival time per source per receiver.

If the above residual is not changing too much, we will accept the inversion result since there is not much model misfit left to drive improvement. Indeed, as shown in Figure 5.12, even if we refine the velocity model, the residuals are almost the same after 15 BFGS steps. In fact, the solutions from the coarse and fine resolution are similar, as shown in Figures 5.11(a) and 5.11(d) .

Using a relative small $\nu = 10^2$, the BFGS iteration has difficulty in converging to a smooth solution. This is clearly seen in Figure 5.11(b). The BFGS iteration stops at the fourth iteration with $E(m^4) \simeq 1700$, where $m = \log c$; see Figure 5.12. This difficulty comes from the sharp spikes in the solution near the source locations, where the traveltime field is not differentiable [88]. These sudden

Δx	Eikonal equation	Adjoint equation
24	20 (6.68×10^{-10})	17 (1.62×10^{-9})
12	28 (9.60×10^{-11})	25 ($3.38\times^{-8}$)

Table 5.1: Iteration count for the fast sweeping methods. The numbers in the brackets are the errors in the corresponding iteration, $||T^{n+1}-T^n||$ or $||\lambda^{n+1}-\lambda^n||$.

changes will degrade accuracy in the computed gradient and make it hard for BFGS to search for a descent direction.

Increasing the magnitude of ν (from 10^4 to 10^6), we have a little bit better convergent result. As seen in the energy plot, the energy which uses the larger ν (the dashed line) reaches a lower state than that using $\nu = 10^4$ (the solid line). Theoretically we penalize the gradient of $\tilde{c}$ so that it is small in a weighted Sobolev space as illustrated in equation (5.12).

Concerning the speed, the computational time for the cases with $\nu = 10^4$ using $\Delta x = 24$ and 12 are 53 minutes and 387 minutes, respectively. We also list in Table 1 the number of iterations required to solve both the eikonal equation and the adjoint equation for each given velocity field. These numbers are obtained for the case $\nu = 10^4$ with only one point source located at $(5000, -2800)$ in the first BFGS iteration. The first row shows the number of iterations with $\Delta x = 24$, while the second row corresponds to the case with $\Delta x = 12$.

CHAPTER 6

Transmission Traveltime Tomography Using Multiple Arrivals

6.1 Introduction

Traveltime tomography [11, 32, 27, 14, 74, 119, 75] is an important class of inverse problems. Theoretically, it amounts to determining uniquely a Riemannian metric by knowing the length of geodesics joining points of the boundary of a two-dimensional or three-dimensional compact domain; those geodesics solve a family of Hamilton-Jacobi equations. Computationally, it amounts to designing fast, high resolution methods to invert given data for the unknown velocities or other material parameters [94, 95, 112, 113, 8, 9, 10, 51, 54].

Geophysical traveltime tomography is closely related to X-ray computerized tomography used in medical diagnosis [33]. The so-called transmission traveltime tomography uses traveltime data between boreholes to invert velocity as a discretized field [14, 119, 31, 76]. Seismic tomography is usually formulated as a minimization problem that produces a velocity model which minimizes the difference between traveltimes generated by tracing rays through the model and traveltimes measured from the data [11, 14, 119, 8, 9, 10, 51, 54]. The ray tracing in the above methods is based on Fermat's Principle and the resulting ray path is computed from an explicit discretization of a nonlinear ray path integral; as

such, the methods inherit some shortcomings from typical ray tracing methods, such as shadow zones due to non-uniform coverage of the computational domain.

On the other hand, the work in [94, 95] has shown that it is possible to formulate the transmission tomography problem in a purely Eulerian framework based on finite difference eikonal solvers; [94, 95] have also derived the linearized eikonal equation and the traveltime gradient based on the adjoint state method borrowed from the optimal control theory.

However, all the above cited methods, together with the one we discuss in the previous chapter, are based on first arrival traveltimes only. As far as we know, no attempt has been made to formulate the transmission tomography problem by using all arrivals, including multivalued traveltimes. The works presented in [32, 27] have used multivalued traveltimes from ray tracing methods, but those works are on reflection traveltime tomography which is different from transmission tomography in that rays start at the surface, reflect off interfaces whose depths are to be determined, and return to the surface.

Because multivalued traveltimes and resulting multipathings are common in complex velocity structures, it is necessary to take into account all the arrivals systematically. To achieve this purpose, we propose to formulate transmission tomography by using the Liouville equation based PDE framework in phase space.

The Liouville equation is a linear hyperbolic equation derived from a Hamiltonian system, which in turn is obtained from a Hamilton-Jacobi equation by the method of characteristics; therefore, the Liouville equation shares the same characteristics as the original nonlinear HJ equation, but it is linear at the price of doubling the number of independent variables; see [37, 4, 57, 24]. In particular, in [86, 87, 69], we have proposed paraxial formulations of the Liouville equation for geometrical optics; the resulting Eulerian framework for computing geometrical

optics related quantities has shown to be computationally efficient and accurate, since to some extent this formulation has overcome the shortcoming of doubling the number of independent variables in the Liouville framework.

Encouraged by the success of the formulations using the paraxial Liouville equations, we apply such techniques to seismic transmission traveltime tomography problems so that multivalued traveltimes and resulting multipathings can be utilized systematically. To that end, we minimize a new energy functional which consists of mismatching terms for multivalued traveltimes and source locations. To minimize the functional, we derive its gradient using the adjoint state technique. Starting from some initial guess, we minimize the nonlinear energy functional by a Newton-type method. The required gradient is computed by solving one forward and one adjoint problem of the paraxial Liouville equations. Then the velocity model is updated iteratively by solving a Helmholtz equation with the computed gradient as the right-hand side. Therefore, the functional can be efficiently minimized by using the simple method of gradient descent, which is a simplified version of the Newton method.

We are also going to carry out the comparison between first arrival based and multiple arrivals based transmission tomography to justify the effectiveness of the approach.

6.2 Tomography Based on Paraxial Liouville Equations

In a forward problem as discussed in Chapter 2 to 4, we assume that the velocity c in the physical domain Ω_p is known; this velocity can then be used to compute the corresponding multivalued arrival-times. The inverse problem is to determine the velocity inside Ω_p using the multivalued arrival-times measured on the physical

boundary.

Denote $\tilde{\Omega} = \Omega \times (0, z_f)$. According to the forward problem given in the above section, we have the following state equations,

$$\begin{aligned} \phi_z + u\phi_x + v\phi_\theta &= 0, \\ T_z + uT_x + vT_\theta &= \frac{1}{c\cos\theta}, \end{aligned} \tag{6.1}$$

where $\mathbf{u} = (u, v) = (\tan\theta, m_z \tan\theta - m_x)$ and $m = m(c) = \log c$. The initial and boundary conditions of this system are given by

$$\begin{aligned} \phi(z_0, \cdot, \cdot) &= x, \\ T(z_0, \cdot, \cdot) &= 0, \\ \phi(z, \cdot, \cdot)|_{\partial\Omega} &= \begin{cases} \phi^* & \text{if } (\mathbf{u}\cdot\mathbf{n}) < 0, \\ \text{no b.c. needed} & \text{if } (\mathbf{u}\cdot\mathbf{n}) \geq 0; \end{cases} \\ T(z, \cdot, \cdot)|_{\partial\Omega} &= \begin{cases} T^* & \text{if } (\mathbf{u}\cdot\mathbf{n}) < 0, \\ \text{no b.c. needed} & \text{if } (\mathbf{u}\cdot\mathbf{n}), \geq 0, \end{cases} \end{aligned} \tag{6.2}$$

where $\mathbf{n}$ is the outward normal vector of $\partial\Omega$, and * denotes the conditions on the inflow boundary of $\partial\Omega$, where $\mathbf{u}\cdot\mathbf{n} < 0$.

In the following, without abusing of notation, we also use * to denote the measured values on the outflow boundary of $\partial\Omega$, where $\mathbf{u}\cdot\mathbf{n} \geq 0$, and the measured values on the level $z = z_f$.

Assume that one can measure the data $\phi(z, \cdot, \cdot)|_{\partial\Omega}$ on the outflow boundary, $\phi(z_f, \cdot, \cdot)$ on the level $z = z_f$, $T(z, \cdot, \cdot)|_{\partial\Omega}$ on the outflow boundary, $T(z_f, \cdot, \cdot)$ on the level $z = z_f$ and $m|_{\partial\Omega_p}$. Such measurement can be picked from seismic data by suitably pairing as in [32].

Based on the above assumptions, the traveltime tomography problem is to determine m, and therefore c, such that the predicted data from the solutions for the system (6.1) and (6.2) are as close to these measurements as possible. To achieve this, we propose to minimize the following energy

$$\begin{aligned} E(m) &= \frac{1}{2}\int_{\Omega}(\phi-\phi^*)^2|_{z=z_f} + \frac{1}{2}\int_z\int_{\partial\Omega}(\mathbf{u}\cdot\mathbf{n})(\phi-\phi^*)^2 \\ &\quad +\frac{\beta}{2}\int_{\Omega}(T-T^*)^2|_{z=z_f} + \frac{\beta}{2}\int_z\int_{\partial\Omega}(\mathbf{u}\cdot\mathbf{n})(T-T^*)^2 \end{aligned} \tag{6.3}$$

The proposed energy is always positive. Although it is possible that $(\mathbf{u}\cdot\mathbf{n}) < 0$ on $\partial\Omega$, in that case $\phi - \phi^* = T - T^* = 0$. This implies that these negative values of $(\mathbf{u}\cdot\mathbf{n})$ will have no contribution to the overall energy. In other words, the proposed energy measures the difference between the computed solution and the measurements on the **outflow** boundary. On the **inflow** boundary, the numerical solution automatically matches the measurements according to the conditions in (6.2).

The parameter β balances the mismatching of the traveltimes and the arrival angles. The energy functional may also consist of a regularization term related to m due to the ill-posedness of an inverse problem. More discussion on the regularity of the solution m, or c, will be discussed later.

To compute the minimizer of the energy (6.3), one can use the method of gradient descent. To do that we need to compute the gradient of this functional by linearization. Here we give a formal derivation of the gradient. Therefore, we perturb m by $\epsilon\tilde{m}$; the corresponding changes in ϕ and T, denoted by $\epsilon\tilde{\phi}$ and $\epsilon\tilde{T}$, respectively, satisfy

$$\tilde{\phi}_z + u\tilde{\phi}_x + v\tilde{\phi}_\theta = [\tilde{m}_x - \tilde{m}_z\tan\theta]\phi_\theta, \tag{6.4}$$

$$\tilde{T}_z + u\tilde{T}_x + v\tilde{T}_\theta \;=\; [\tilde{m}_x - \tilde{m}_z \tan\theta]T_\theta - \frac{\tilde{m}}{c\cos\theta}\,. \tag{6.5}$$

Define $\tilde{\mathbf{u}} = (0, \tilde{v}) = (0, \tilde{m}_z \tan\theta - \tilde{m}_x)$. The corresponding change in the energy is given by

$$\begin{aligned}
\delta E &= E(m + \epsilon\tilde{m}) - E(m) \\
&= \epsilon\left[\int_\Omega \tilde{\phi}(\phi - \phi^*)|_{z=z_f} + \int_z\int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\tilde{\phi}(\phi - \phi^*)\right. \\
&\quad + \frac{1}{2}\int_z\int_{\partial\Omega} (\tilde{\mathbf{u}}\cdot\mathbf{n})(\phi - \phi^*)^2 + \beta\int_\Omega \tilde{T}(T - T^*)|_{z=z_f} \\
&\quad \left. + \beta\int_z\int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\tilde{T}(T - T^*) + \frac{\beta}{2}\int_z\int_{\partial\Omega} (\tilde{\mathbf{u}}\cdot\mathbf{n})(T - T^*)^2\right] \\
&\quad + O(\epsilon^2)\,.
\end{aligned} \tag{6.6}$$

However, it is not clear how to choose the perturbation $\tilde{m}$ at this moment so that the energy is decreased. The reason is that δE depends not only on $\tilde{m}$, but also on the perturbations in $\tilde{\phi}$ and $\tilde{T}$, and the latter two variations depend on $\tilde{m}$ implicitly according to (6.4) and (6.5).

To make $\tilde{m}$ explicit in the above energy perturbation, we use integration by parts and introduce adjoint equations. First, introducing a Lagrange multiplier λ_1 for equation (6.4), multiplying the equation by λ_1 and integrating over $\tilde{\Omega}$, we have

$$\begin{aligned}
0 &= \int_{\tilde{\Omega}} (\lambda_1, u\lambda_1, v\lambda_1)\cdot\nabla\tilde{\phi} + \int_{\tilde{\Omega}} \lambda_1\tilde{v}\phi_\theta \\
&= \int_\Omega \lambda_1\tilde{\phi}|_{z=z_f} + \int_z\int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\lambda_1\tilde{\phi} \\
&\quad - \int_{\tilde{\Omega}} \nabla\cdot(\lambda_1, u\lambda_1, v\lambda_1)\tilde{\phi} + \int_{\tilde{\Omega}} \lambda_1\tilde{v}\phi_\theta\,.
\end{aligned} \tag{6.7}$$

Similarly, we introduce another Lagrange multiplier λ_2 for equation (6.5).

This gives

$$\begin{aligned} 0 &= \int_\Omega \lambda_2 \tilde{T}|_{z=z_f} + \int_z \int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\lambda_2 \tilde{T} - \int_{\tilde{\Omega}} \nabla\cdot(\lambda_2, u\lambda_2, v\lambda_2)\tilde{T} \\ &\quad + \int_{\tilde{\Omega}} \lambda_2 \left\{ \tilde{v} T_\theta + \frac{\tilde{m}}{c\cos\theta} \right\} . \end{aligned} \tag{6.8}$$

To eliminate $\tilde{\phi}$ and $\tilde{T}$ in δE, we multiply equations (6.7) and (6.8) by ϵ and $\beta\epsilon$, respectively, and then add them to equation (6.6). This gives

$$\begin{aligned} \delta E &= \epsilon \Bigg[\int_\Omega \tilde{\phi}(\phi-\phi^*)|_{z=z_f} + \int_z \int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\tilde{\phi}(\phi-\phi^*) \\ &\quad + \frac{1}{2}\int_z \int_{\partial\Omega} (\tilde{\mathbf{u}}\cdot\mathbf{n})(\phi-\phi^*)^2 + \beta \int_\Omega \tilde{T}(T-T^*)|_{z=z_f} \\ &\quad + \beta \int_z \int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\tilde{T}(T-T^*) + \frac{\beta}{2}\int_z \int_{\partial\Omega} (\tilde{\mathbf{u}}\cdot\mathbf{n})(T-T^*)^2 \\ &\quad + \int_\Omega \lambda_1 \tilde{\phi}|_{z=z_f} + \int_z \int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\lambda_1\tilde{\phi} \\ &\quad - \int_{\tilde{\Omega}} \nabla\cdot(\lambda_1, u\lambda_1, v\lambda_1)\tilde{\phi} + \int_{\tilde{\Omega}} \lambda_1 \tilde{v}\phi_\theta \\ &\quad + \beta \int_\Omega \lambda_2 \tilde{T}|_{z=z_f} + \beta \int_z \int_{\partial\Omega} (\mathbf{u}\cdot\mathbf{n})\lambda_2\tilde{T} \\ &\quad - \beta \int_{\tilde{\Omega}} \nabla\cdot(\lambda_2, u\lambda_2, v\lambda_2)\tilde{T} + \beta \int_{\tilde{\Omega}} \lambda_2 \left\{ \tilde{v} T_\theta + \frac{\tilde{m}}{c\cos\theta} \right\} \Bigg] \\ &\quad + O(\epsilon^2) . \end{aligned} \tag{6.9}$$

Next, we choose λ_1 and λ_2 satisfying

$$\begin{aligned} (\lambda_1)_z + (u\lambda_1)_x + (v\lambda_1)_\theta &= 0, \\ (\lambda_2)_z + (u\lambda_2)_x + (v\lambda_2)_\theta &= 0 , \end{aligned} \tag{6.10}$$

with the *initial* conditions on $z = z_f$,

$$\lambda_1(z=z_f) = \phi^* - \phi \ \text{ and } \ \lambda_2(z=z_f) = T^* - T, \tag{6.11}$$

and the boundary conditions,

$$\begin{aligned}
\lambda_1|_{\partial\Omega} &= \begin{cases} \phi^* - \phi & \text{if } (\mathbf{u}\cdot\mathbf{n}) > 0, \\ \text{no b.c. needed} & \text{if } (\mathbf{u}\cdot\mathbf{n}) \le 0; \end{cases} \\
\lambda_2|_{\partial\Omega} &= \begin{cases} T^* - T & \text{if } (\mathbf{u}\cdot\mathbf{n}) > 0, \\ \text{no b.c. needed} & \text{if } (\mathbf{u}\cdot\mathbf{n}) \le 0. \end{cases}
\end{aligned} \tag{6.12}$$

Ignoring all higher than linear order terms in the above equation for δE, we have the energy perturbation

$$\begin{aligned}
\delta E &= \epsilon \int_{\tilde{\Omega}} \left[\tilde{v}(\lambda_1\phi_\theta + \beta\lambda_2 T_\theta) + \frac{\beta\lambda_2\tilde{m}}{c\cos\theta} \right] \\
&\quad + \frac{\epsilon}{2}\int_{\Omega_p} \tilde{v}(\phi-\phi^*)^2|_{\theta=\theta_{\min}}^{\theta=\theta_{\max}} + \frac{\epsilon\beta}{2}\int_{\Omega_p} \tilde{v}(T-T^*)^2|_{\theta=\theta_{\min}}^{\theta=\theta_{\max}} \\
&= -\epsilon \int_{\Omega_p} \left\{ \tilde{m}_x \left[\int_\theta \lambda_1\phi_\theta + \beta\lambda_2 T_\theta \right] - \tilde{m}_z \left[\int_\theta \tan\theta\,(\lambda_1\phi_\theta + \beta\lambda_2 T_\theta) \right] \right. \\
&\quad \left. - \frac{\tilde{m}\beta}{c}\left[\int_\theta \frac{\lambda_2}{\cos\theta} \right] - \frac{1}{2}\int_{\Omega_p} \tilde{v}\left[(\phi-\phi^*)^2 + \beta(T-T^*)^2 \right]|_{\theta=\theta_{\min}}^{\theta=\theta_{\max}} \right\} \\
&= \epsilon \int_{\Omega_p} \tilde{m} g\,,
\end{aligned} \tag{6.13}$$

where

$$\begin{aligned}
g(x,z) &= [f_1(x,z)]_x - [f_2(x,z)]_z + \frac{\beta}{c} f_3(x,z) + f_4(x,z), \\
f_1(x,z) &= \int_\theta \lambda_1\phi_\theta + \beta\lambda_2 T_\theta, \\
f_2(x,z) &= \int_\theta \tan\theta\,(\lambda_1\phi_\theta + \beta\lambda_2 T_\theta)\,, \\
f_3(x,z) &= \int_\theta \frac{\lambda_2}{\cos\theta}, \\
f_4(x,z) &= \frac{1}{2}\left\{ \frac{\partial}{\partial x}\left[(\phi-\phi^*)^2 + \beta(T-T^*)^2 \right] \right.
\end{aligned}$$

$$- \tan\theta \frac{\partial}{\partial z} \left[(\phi - \phi^*)^2 + \beta (T - T^*)^2 \right] \Bigg\} \Bigg|_{\theta=\theta_{\min}}^{\theta=\theta_{\max}} . \qquad (6.14)$$

To minimize the energy using the method of gradient descent, one could choose the perturbation $\tilde{m} = -g$. This implies

$$\delta E = -\epsilon \int_{\Omega_p} \tilde{m}^2 \leq 0 \qquad (6.15)$$

and the equality holds when $||\tilde{m}|| = 0$. However, it is not straight-forward how one can guarantee the following two properties:

1. $\tilde{m}|_{\partial\Omega_p} = 0$;

2. $m^{k+1} = m^k + \epsilon \tilde{m}^k$ smooth.

The first condition assumes that we can measure m on the boundary $\partial\Omega_p$, denoted by $m^*|_{\partial\Omega_p}$, which is a reasonable assumption. This means that the variations of the velocity function near the boundary should be zero.

The second condition is a regularity condition in m^k. The regularity requirement seems to be too restrictive in practice. In general, one only needs $m^k \in C^1$, rather than C^∞ as required here, to guarantee well-posedness of state equations (6.1). However, assuming that one uses $\tilde{m}^k = -g$ directly, it is not clear whether this function g would give us the desired regularity. Even if $g \in C^1$, the numerical solution of g could have jumps or spikes. These irregularities will force one to pick a very small step-size, ϵ^k, in the minimization process, and/or a very restrictive step-size, dz, in solving the advection equations (6.1) and the conservation laws (6.10). Therefore, to have faster convergence to the minimizer, we impose the above regularity in each iteration.

One way to satisfy the above two properties is to use the descent direction

$$\tilde{m} = -(I - \nu\Delta)^{-1} g \,, \tag{6.16}$$

where I is the identity operator, Δ is the Laplacian operator and $\nu \geq 0$ controls the amount of regularity that one wants. The homogeneous boundary condition is imposed in inverting the operator $(I - \nu\Delta)$. With this $\tilde{m}$, we have

$$\delta E = -\epsilon \int_{\Omega_p} (|\tilde{m}|^2 + \nu |\nabla \tilde{m}|^2) \leq 0 \,, \tag{6.17}$$

and the equality is achieved when $\tilde{m} \equiv 0$.

Here, we give a brief justification on the above regularization. Assume that $g \in L^2(\Omega_p)$. Consider the operator equation

$$-\tilde{m} = g \tag{6.18}$$

in $H_0^1(\Omega_p)$. Apparently this equation is ill-posed. To use the Tikhonov regularization method [110], we constrain $\tilde{m}$ to be bounded in $H_0^1(\Omega_p)$; then by the Poincare's inequality, we only need to bound $\nabla\tilde{m}$ in L^2-norm.

We have the following theorem.

Theorem 1 *Consider equation (6.18) in*

$$\tilde{\mathcal{M}}_{ad} = \{\tilde{m} : \|\nabla\tilde{m}\|_{L^2} \leq B, \tilde{m} \in H_0^1(\Omega_p)\},$$

where B is a positive number. The following functional

$$J(\tilde{m}) = \| - \tilde{m} - g\|_{L^2}^2 + \nu \|\nabla\tilde{m}\|_{L^2}^2 \tag{6.19}$$

has a unique minimizer in $\tilde{\mathcal{M}}_{ad}$ *satisfying*

$$-(I - \nu\Delta)\tilde{m} = g, \tag{6.20}$$

where $\nu \geq 0$ *is a so-called regularization parameter.*

Proof: It is easy to verify that $\tilde{\mathcal{M}}_{ad}$ is a closed, convex bounded set in $H_0^1(\Omega_p)$. Tikhonov regularization reduces solving equation (6.18) in $\tilde{\mathcal{M}}_{ad}$ to minimizing the functional $J(\tilde{m})$. Then the variational principle applied to $J(\tilde{m})$ yields the equation (6.20). Because $J(\tilde{m})$ is a quadratic functional, according to the standard theory $J(\tilde{m})$ has a unique minimizer given by the condition (6.20). □

Determining an optimal regularization parameter is one of the crucial points in applications of regularization methods. One simple way is to fix this parameter in all iterations. In most examples below, we determine this regularization parameter by trial and error. However, as m^k is getting closer to a minimizer, it is not necessary to have large regularization. Therefore, one may try to decrease the magnitude of ν as k increases. In this paper, we have also tried in Section 6.5.2 to vary ν^k by $\nu^k = (\nu_{\max} - \nu_{\min})\nu_0^k + \nu_{\min}$ with k is the number of iteration.

Remark:

1. We may define some functional spaces and inner products, introduce some operator notations, derive the corresponding adjoint operators and gradient operators accordingly; however, here we prefer the above formal derivation which we think is much more transparent.

2. We have to justify the Frechet differentiability of the nonlinear functional, which in general is not an easy task.

3. We may also carry out the sensitivity analysis of the underlying forward nonlinear operator. The above issues will be addressed in a forthcoming paper.

6.3 An Algorithm and Some Implementation Details

According to the above formulation, we have the following algorithm.

Algorithm:

1. Initialize $m = m^k$ for $k = 0$.
2. Obtain $\phi(x, \theta, z)$ and $T(x, \theta, z)$ by solving (6.1), with the velocity $m = m^k$ using the initial and boundary conditions (6.2).
3. Obtain $\lambda_1(x, \theta, z)$ and $\lambda_2(x, \theta, z)$ by solving (6.10) with the velocity $m = m^k$ using the initial condition (6.11) and the boundary condition (6.12).
4. Compute g using (6.14).
5. Obtain $\tilde{m}^k(x, z)$ by solving (6.16).
6. Update $m^{k+1} = m^k + \epsilon^k \tilde{m}^k$.
7. Go back to Step 2 until $||\tilde{m}^k|| \leq \delta$ or $k \geq k_{\max}$, where δ is a small positive number and $k_{\max}$ is a large positive integer.

In the above algorithm, one might improve the computational efficiency, for example, by using the Armijo-Goldstein rule in picking ϵ^k or by replacing the method of gradient descent by the BFGS method. However, to simplify the presentation we use the method of gradient descent to demonstrate the effectiveness

of the new formulation; thus we simply choose $\epsilon^k = \epsilon$ to be a sufficiently small constant.

The initial guess for m, denoted by m^0, has to satisfy the conditions

1. $m^0|_{\partial\Omega_p} = m^*|_{\partial\Omega_p}$;
2. m^0 smooth.

One way to meet these conditions is to solve

$$(I - \nu\Delta)c^0 = 0\,, \tag{6.21}$$

with the boundary condition $c^0|_{\partial\Omega_p} = c^*|_{\partial\Omega_p}$, and set $m^0 = \log c^0$. Or, m^0 can directly be obtained by solving $(I - \nu\Delta)m^0 = 0$ with the boundary condition $m^0|_{\partial\Omega_p} = \log c^*|_{\partial\Omega_p}$. Yet another possibility is to initialize it by solving a transmission tomography problem based on the first arrival-time only, for example, as in [94, 95] or in the previous chapter. By using this approach, the initial guess m^0 should be closer to the *exact* minimizer of (6.3) than that from solving the above elliptic equation.

In Step 2 and Step 3 in the above algorithm, we need to solve two advection equations and two conservation laws. They are solved by the third order weighted essentially non-oscillatory (WENO) scheme [55] in space and the second order total variation diminishing (TVD) Runge-Kutta (RK) method in time [99]. In Step 5, we solve the Helmholtz equation (6.16) by FFT.

6.4 Practical Details

In this section, we briefly explain how the above method can be used in real applications. The first concern is the problem of association. When a receiver

measures the arrival-time, it usually does not have any further information on the arrival-angle, θ. In using the above algorithm directly, the first difficulty is that we need to match the measurements with our numerical solution in the phase space and this requires the information in the θ-direction. This problem can be easily solved. According to the arrival-time information, we can easily approximate the arrival-angle using

$$\tan\theta^* = \frac{T_x}{\sqrt{c^{-2} - T_x^2}} \quad \text{or} \quad \tan\theta^* = \frac{T_z}{\sqrt{c^{-2} - T_z^2}} \tag{6.22}$$

according to whether the receiver is on the level $z = 0$ or $x = x_{\min,\max}$. This approach is similar to the ray parameter method developed as in [32].

Therefore, to associate different arrival rays in the phase space in practise, we first numerically compute T_x, or T_z, along each branch of the arrival-time measured at a receiver on $z = z_f$, or $x = x_{\min,\max}$ respectively. Assuming that we know the velocity close to the surface, the arrival-angle can be approximated using the above expression. After this arrival-angle is determined for each arrival-rays received at a receiver, we associate the level set function and the arrival-time function by $\phi(x, \theta^*, z_f) = x_s$ and $T(x, \theta^*, z_f) = T^*$, respectively, with x_s is the corresponding source location on the underground level $z = 0$.

Another difficulty concerns with the boundary conditions (6.2), and therefore (6.12). In practise, it might not be obvious how one can obtain all boundary values of ϕ^* and T^*. For example, it might be expansive to have receivers everywhere on the sides $x = x_{\min}$ and $x = x_{\max}$. Even if we had enough resources to do so, the above formulation also requires measurements on the surfaces $\{(z, x, \theta) : 0 \le z \le z_f, x_{\min} \le x \le x_{\max} \text{ and } \theta = \theta_{\min}\}$ and $\{(z, x, \theta) : 0 \le z \le z_f, x_{\min} \le x \le x_{\max} \text{ and } \theta = \theta_{\max}\}$. These information might not be easily accessible in general. We may need to modify the above algorithm in real applica-

tion to cope with the situation when only a limited number of both receivers and sources are available. In particular, it is more natural to have several receivers on $z = z_f$ and some sources on $z = 0$. This is what we are going to assume for the rest of this section.

To solve this problem, we propose the following strategy. We first denote the measurements on the ground level in the phase space by the set

$$\Gamma(z_f) = \{(x, \theta) : \text{ both } \phi(x, \theta, z_f) \text{ and } T(x, \theta, z_f) \text{ are known}\}. \tag{6.23}$$

Next, we define $\Gamma(z)$ as the collection of all characteristic curves in the phase space obtained by tracing back all arrival rays from the set $\Gamma(z_f)$ according to the Liouville equation. Because of the paraxial assumption, any characteristics from $\Gamma(z)$ have to satisfy $-\pi/2 < -\theta_{\max} < \theta(z) < \theta_{\max} < \pi/2$ for all $z \in [0, z_f]$. This means that any characteristics from $\Gamma(z)$ should never touch the surfaces $\{(z, x, \theta) : 0 \leq z \leq z_f, x_{\min} \leq x \leq x_{\max} \text{ and } \theta = \pm\pi/2\}$.

Because the Liouville equation is a linear hyperbolic equation and therefore any characteristics will never intersect, we can use an enlarged numerical domain for our computations so that any imposed boundary conditions (6.2) and (6.12) for ϕ, T, λ_1 and λ_2 will not interfere any values located in $\Gamma(z)$ at all in any later calculations. In the implementation, one could for example simply set numerically $(\mathbf{n} \cdot \nabla\phi)|_{\partial\Omega} = 0$ for the level set equation, if a boundary condition on $\partial\Omega$ is necessary.

Next, we need to modify our energy. The reason is that we have finite measurements only at $\Gamma(z_f)$ in the phase space. We can match all measurements with our numerical solutions only at these locations. Now, because all characteristics

from $\Gamma(z)$ will not touch $\partial\Omega$, our original energy reduces to

$$E_{\text{new}}(m) \;=\; \frac{1}{2}\int_{\Omega}(\phi-\phi^*)^2\delta(\Gamma(z))\Big|_{z=z_f} + \frac{\beta}{2}\int_{\Omega}(T-T^*)^2\delta(\Gamma(z))\Big|_{z=z_f} \tag{6.24}$$

where $\delta(.)$ is the Dirac's delta function. Any of these terms

$$\int_z\int_{\partial\Omega}(\mathbf{u}\cdot\mathbf{n})(\phi-\phi^*)^2 \text{ and } \int_z\int_{\partial\Omega}(\mathbf{u}\cdot\mathbf{n})(T-T^*)^2 \tag{6.25}$$

will have no contributions to this new energy.

Even with these modifications, we do not need to change much of the above formulations however. For the forward problem, we can still keep both paraxial Liouville equations (6.1) and their corresponding initial conditions. Furthermore, because we have enlarged our computational domain, the boundary conditions for these equations will not affect the energy we are going to minimize. If necessary, as mentioned above, we could simply numerically impose a Neumann boundary condition.

The adjoint equations are still the same as in (6.10). Their corresponding *initial* conditions (6.11), however, are modified to incorporate with the measurements

$$\lambda_1(z=z_f) = (\phi^*-\phi)\delta(\Gamma(z_f)) \text{ and } \lambda_2(z=z_f) = (T^*-T)\delta(\Gamma(z_f))\,. \tag{6.26}$$

Their boundary conditions are treated in the same way as for the state equations. Numerically, we could approximate this delta-function by the following smeared version as in the standard level set method

$$\delta_\epsilon(x) = \frac{1}{4\epsilon}\left[1-\text{sign}(|x|-\epsilon)\right]\left[1+\cos\left(\frac{\pi x}{\epsilon}\right)\right]\,, \tag{6.27}$$

for some small $\epsilon > 0$. Therefore, we are smearing the set of the experimental measurements $\Gamma(z_f)$. The characteristics $\Gamma(z)$ now becomes tubes with radius ϵ in the phase space.

The gradient of this new energy, $g_{\text{new}}(x, z)$, is also similar to the expression as before. Following the same approach as in Section 6.4, we have

$$g_{\text{new}}(x, z) = [f_1(x, z)]_x - [f_2(x, z)]_z + \frac{\beta}{c} f_3(x, z)\,, \tag{6.28}$$

with the functions f_1, f_2 and f_3 same as in (6.14).

Practical Algorithm:

1. Initialize $m = m^k$ for $k = 0$.
2. Obtain $\phi(x, \theta, z)$ and $T(x, \theta, z)$ by solving (6.1), with the velocity $m = m^k$ using the initial condition as in (6.2) and a Neumann boundary conditions.
3. Obtain $\lambda_1(x, \theta, z)$ and $\lambda_2(x, \theta, z)$ by solving (6.10) with the velocity $m = m^k$ using the initial condition (6.26) and a Neumann boundary conditions.
4. Compute $g_{\text{new}}(x, z)$ using (6.28).
5. Obtain $\tilde{m}^k(x, z)$ by solving (6.16).
6. Update $m^{k+1} = m^k + \epsilon^k \tilde{m}^k$.
7. Go back to Step 2 until $||\tilde{m}^k|| \leq \delta$ or $k \geq k_{\max}$, where δ is a small positive number and $k_{\max}$ is a large positive integer.

There are a few ways one can further speed up this algorithm. For example, it is possible to apply a Semi-Lagrangian method as in Chapter 4 to

solve all the state equations and the adjoint equations. As a simple case, to determine $\phi(x^*, \theta^*)$, one could solve the following the system of equations for $(x(z=0), \theta(z=0))$

$$\frac{d}{dz}\begin{pmatrix} x(z) \\ \theta(z) \end{pmatrix} = \begin{pmatrix} u(x,\theta) \\ v(x,\theta) \end{pmatrix} \tag{6.29}$$

with terminal conditions $x(z=z_f) = x^*$ and $\theta(z=z_f) = \theta^*$. Then, $\phi(x^*, \theta^*) = x(z=0)$. This numerical method is similar to the classical ray tracing method, except that we are now tracing the ray backward instead. With this Semi-Lagrangian method, it is also possible to speed up the computations by solving ϕ, T, λ_1 and λ_2 only within the computational tube $\Gamma(z)$. This is because the initial conditions for λ's are non-zero only in a neighborhood of the set $\Gamma(z_f)$. According to (6.10), this initial condition implies that both λ_1 and λ_2 will both be zero in $\tilde{\Omega} \setminus \Gamma(z)$. This concludes with (6.28) that we can skip computing ϕ and T outside the tube of characteristics $\Gamma(z)$. In the current paper, however, we have not implemented this idea. Instead, we follow the same Eulerian approach as before by solving all these equations using WENO3-RK2.

6.5 Examples

In the following examples, both the parameters β and ν are set to be equal to 1. This means that the regularization for the velocity comes from the operator $(I - \nu\Delta)$. The physical domain Ω_p is defined by $(x_{\min}, x_{\max}) = (-1, 1)$ and $(z_0, z_f) = (0, 2)$. For the constant model, we use $(\theta_{\min}, \theta_{\max}) = (-9\pi/20, 9\pi/20)$. For the waveguide and Gaussian example, we use $(\theta_{\min}, \theta_{\max}) = (-\pi/3, \pi/3)$. The initial guess of the model m^0 is determined by solving equation (6.21) with FFT.

The **inflow** boundary conditions for ϕ and T, represented by ϕ^* and T^*, are

obtained by solving equations (6.1) using a semi-Lagrangian method, as in [69]; here the characteristic system is solved using the MATLAB function ode45.

The **outflow** boundary conditions ϕ^* and T^* are obtained by solving the system (6.1) using the exact velocity c with the third order WENO scheme in space and the second order TVD-RK stepping in time.

To demonstrate the practical algorithm, we also repeat these examples but with limited information recorded on the boundary. We consider the situation where only 1 source is located at $(x, z) = (0, 0)$ and 5 receivers are located at $z = 2$ and $x_s = 0.2, 0.1, 0.0, 0.1$ and 0.2. The set $\Gamma(z_f)$ is obtained by solving (6.1) using a semi-Lagrangian method on all grid points on the level $z = z_f$. Then,

$$\Gamma(z_f) = \left\{ (x, \theta) : \sum_{x_s, x_r} \delta_\epsilon(\phi(x, \theta) - x_s)\delta_\epsilon(x - x_r) \neq 0 \right\},$$

where $\delta_\epsilon(\cdot)$ is defined by (6.27).

6.5.1 Constant Model

The exact velocity model is given by $c \equiv 1$. In this case, the boundary measurements of m^* are simply given by

$$m^*|_{\partial\Omega_p} = 0\,. \tag{6.30}$$

The number of grid points used in this test case is $129 \times 129 \times 129$ in the x-θ-z space.

The first row of Figure 6.1 shows the initial guess c^0 and the approximated solution c^{500}; the second row shows the error in c^{500} and the change in the energy during the iteration.

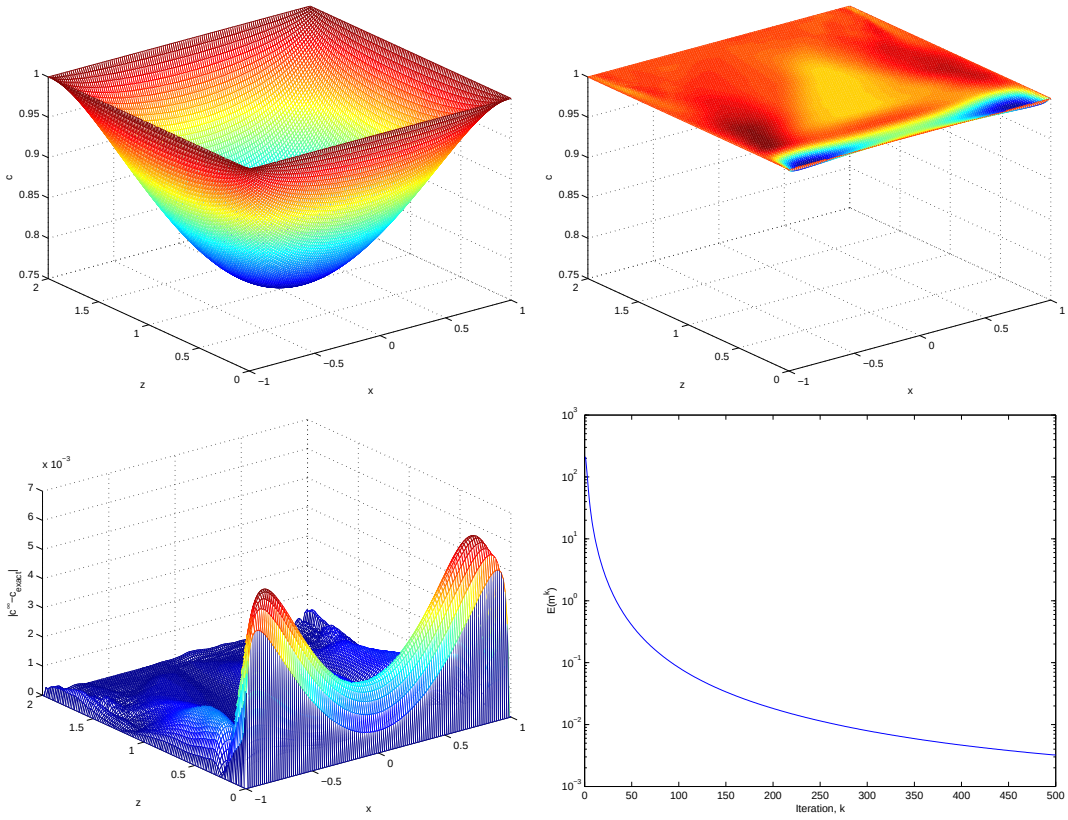


Figure 6.1: (Constant Model) The initial guess and final approximated c, the error in the solution and the convergence history of energy in semi-log scale.

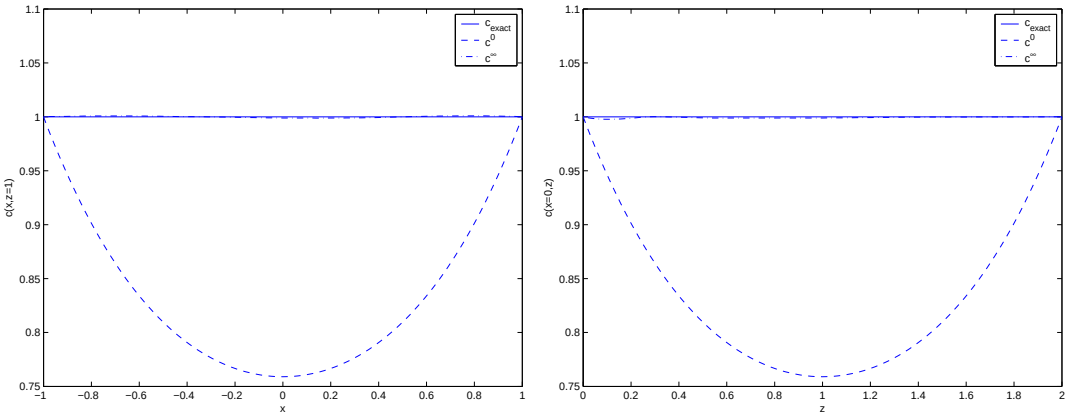


Figure 6.2: (Constant Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

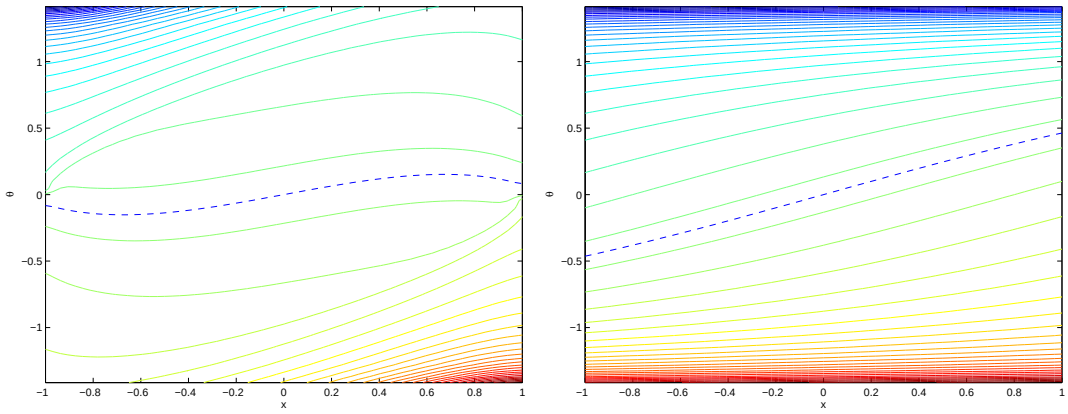


Figure 6.3: (Constant Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively.

Figure 6.2 shows the cross-sections of the initial guess, the convergent velocity and the exact velocity; the results validate the effectiveness and the convergence of the algorithm.

Figure 6.3 shows the contour plots of the level set function $\phi(z = 2, x, \theta)$ using $c = c^0$ and $c = 1$. The dashed line represents the zero level set. An essential part of the transmission tomography problem is to drive the level sets with $c = c^0$ to match with the level sets with the exact velocity model c.

To compare with the results from the first-arrival based traveltime tomography discussed in the previous chapter, we start from the same initial guess as illustrated in Figure 6.1 and carry out the first-arrival based tomography using the same physical discretization with the mesh size of 129×129 in the x-z space. Figure 6.4 shows the final converged velocity c after 10000 iterations and the convergence history, where the fast sweeping method is used to carry out the forward simulation [122, 60]; Figure 6.5 shows the cross sections of the converged velocity along $z = 1$ and $x = 0$; the above computation is carried out with a single source only, namely, the traveltime information being generated from a single point source.

As we can see, the Liouville formulation proposed here does give us some advantages over the traditional first-arrival based traveltime tomography since it takes into account all the sources in a natural way even though there are no multi-arrivals for the constant velocity model.

6.5.2 Waveguide Model

The exact velocity model is given by

$$c(x, z) = 3.0 - 2.5 \exp\left(-\frac{x^2}{2}\right) . \tag{6.31}$$

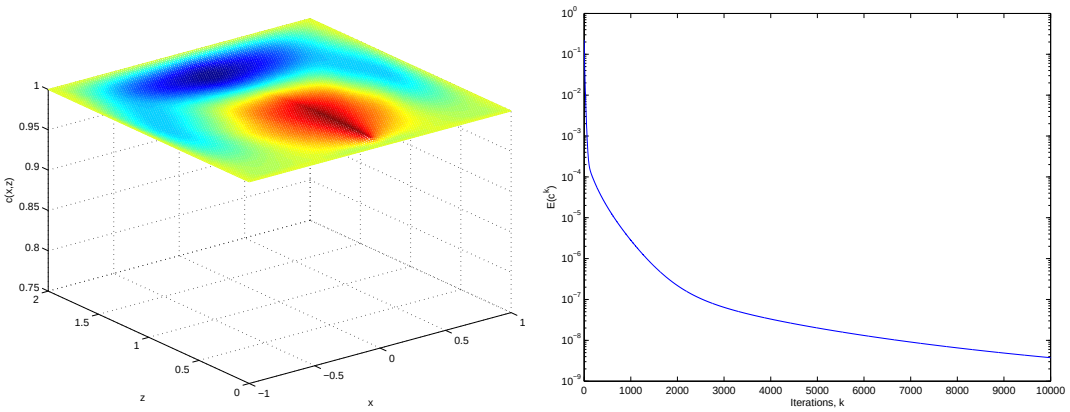


Figure 6.4: (Constant Model) First-arrival based traveltime tomography with a single source and multiple receivers: the final approximated c and the convergence history of energy in semi-log scale.

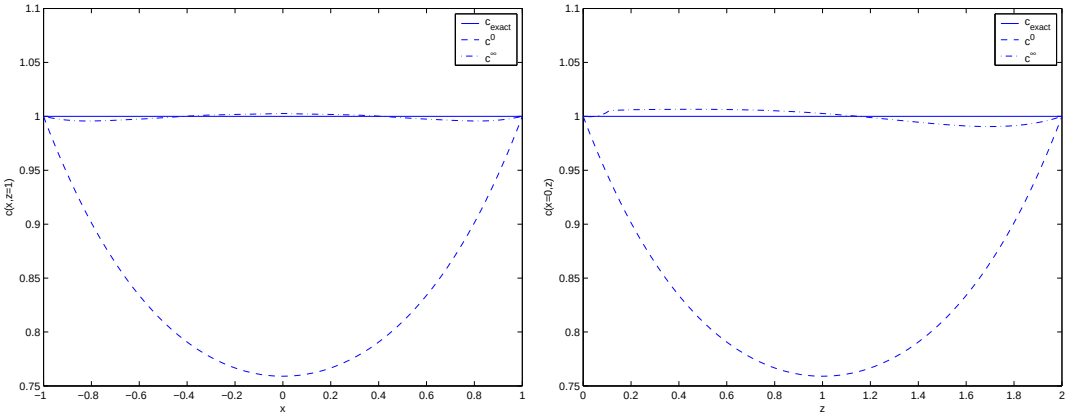


Figure 6.5: (Constant Model) First-arrival based traveltime tomography with a single source and multiple receivers: Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

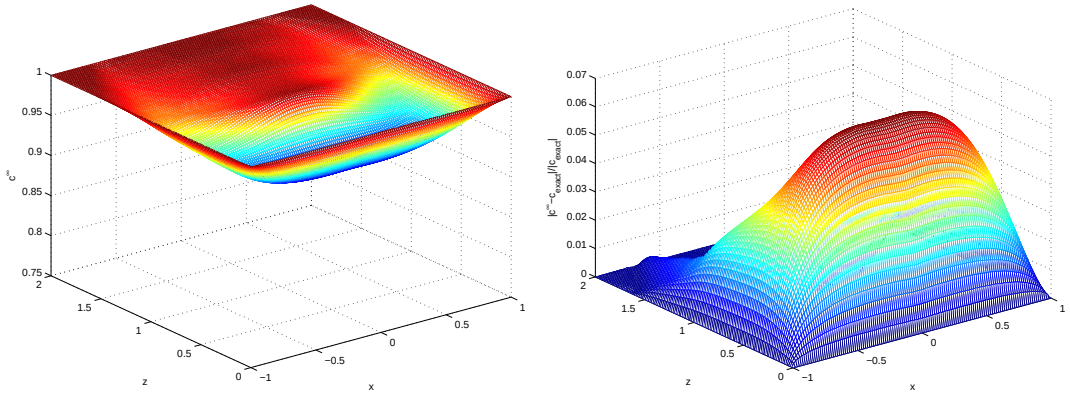


Figure 6.6: (Constant Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution.

The number of grid points used is $129 \times 129 \times 129$ in the x-θ-z space.

The first row of Figure 6.7 shows the initial guess c^0 and the approximated solution c^{100}; the second row of the figure shows the error in c^{100} and the change in the energy during the iteration.

Figure 6.9 shows the contour plots of the level set function $\phi(z = 2, x, \theta)$ using $c = c^0$ and the exact velocity model. The dashed line represents the zero level set. Multivalued arrival angles in θ are clearly seen, and this implies that the corresponding traveltimes are also multivalued. For example, for the exact velocity field c, if we look at rays from source $(x_s, z_s) = (0, 0)$, then we have 3 arrivals at most locations on the level $z = 2$. Even though multivalued solutions are obtained from the initial guess, the shape of the level sets are quite different from that produced by the exact velocity model.

There are some artifacts related to the aperture limitation introduced by the paraxial formulation as we can see from Figure 6.7; these could be filtered out by some filters applied to the data near the surface.

As discussed in Section 6.3, we have regularized the minimization by search-

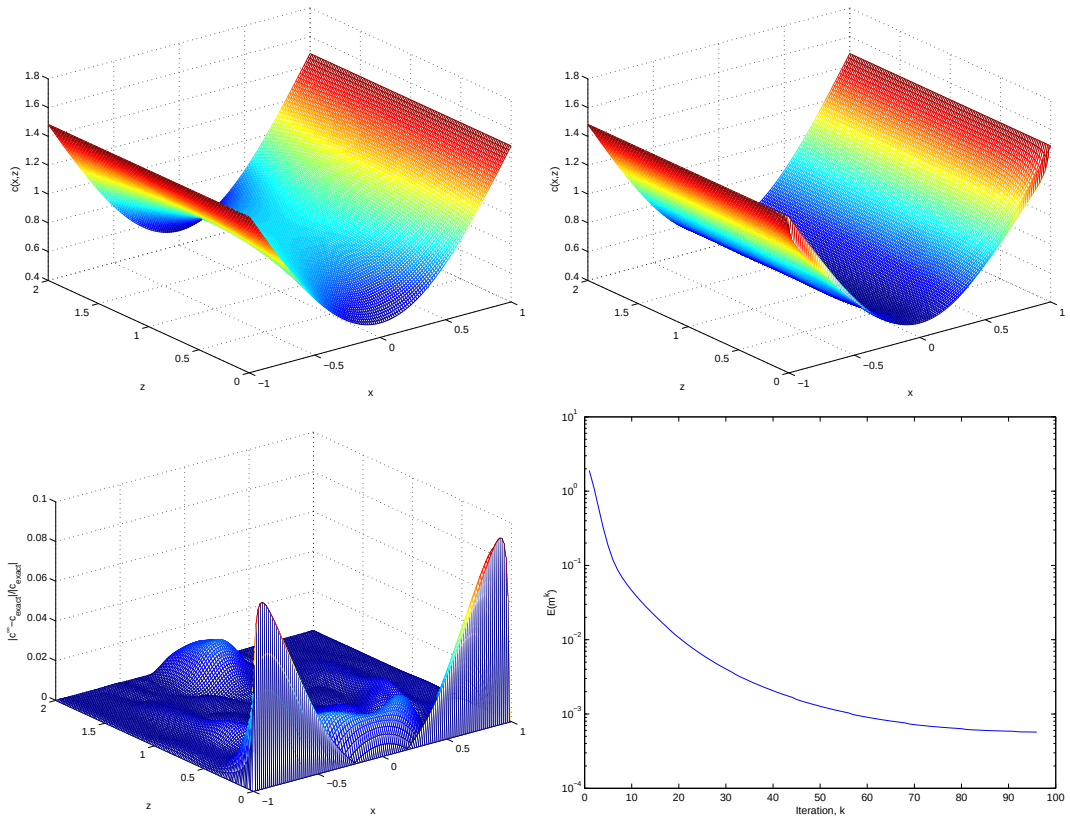


Figure 6.7: (Waveguide Model) The initial guess and final approximated c, the relative error in the solution and the convergence history of energy in semi-log scale.

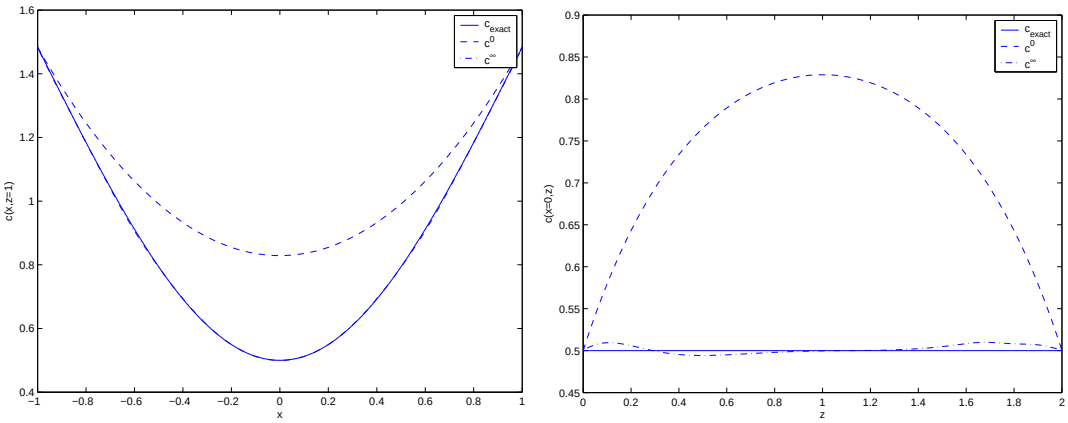


Figure 6.8: (Waveguide Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

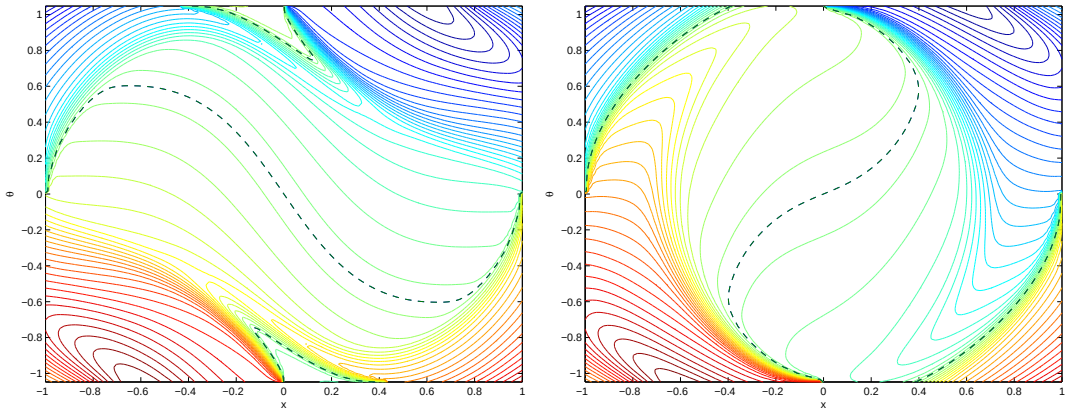


Figure 6.9: (Waveguide Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively.

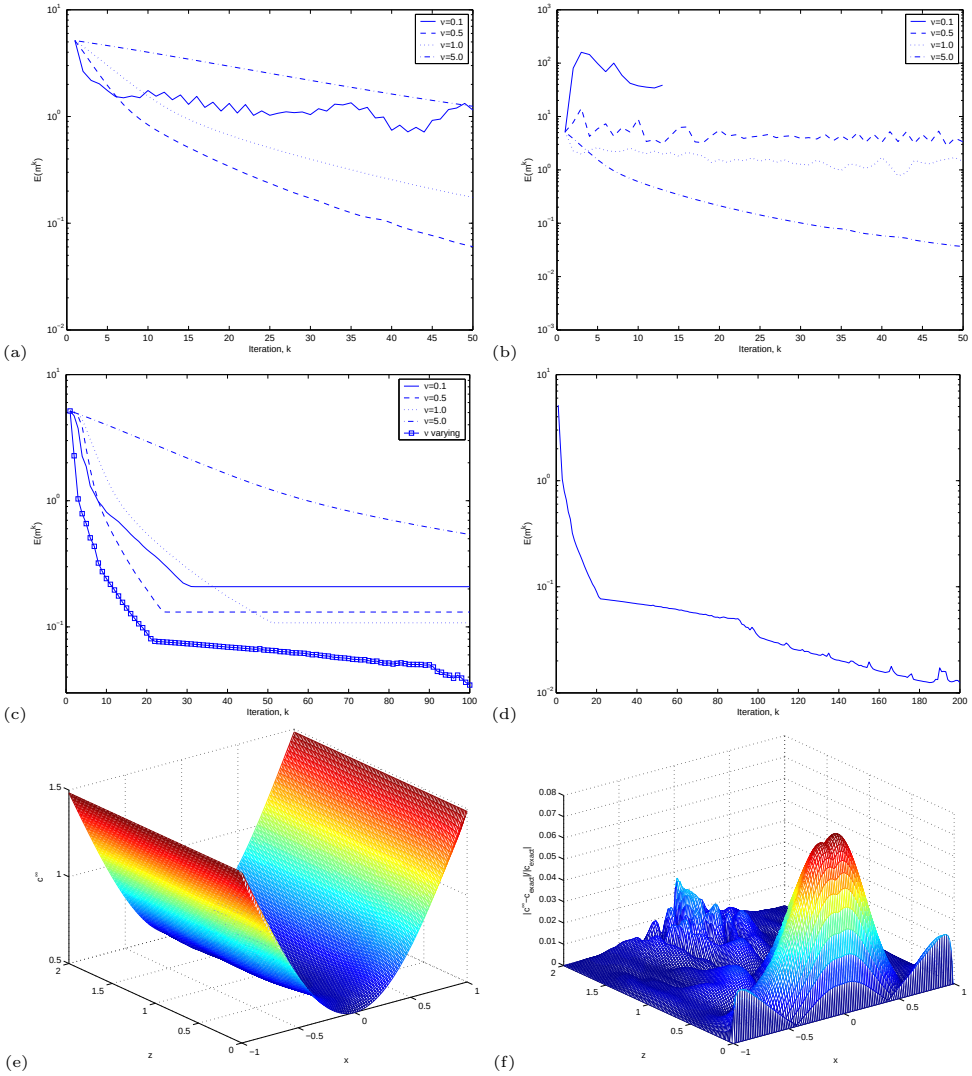


Figure 6.10: (Waveguide Model) Energy histories with different regularization ν using the method of gradient descent with (a) $\epsilon = 0.01$ and (b) $\epsilon = 0.10$, and (c) varying ν using the strategy proposed in Section 6.2 and (d) the relative error in the solution using this strategy.

ing the gradient $\tilde{m}$ in a Sobolev space $H_0^1(\Omega_p)$ through equation (6.16). In all examples above, we have fixed $\nu = 1.0$ in our computations. We have also studied the effect of the solution by changing this variable ν. Figure 6.10 (a) and (b) show the convergent histories of different ν's using two different but fixed ϵ's, respectively. On (a), we have plotted the energies for the case with a relatively smaller ϵ. We first notice that the smaller the regularization ν, the faster the energy converges to it's minimum. This can be seen clearly from the first few iterations. Among those four ν's, $\nu = 0.1$ gives the fastest decreasing rate in the energy. However, as seen from those later iterations in the case $\nu = 0.1$ in the plot (a) and the cases $\nu = 0.1, 0.5$ and 1.0 from (b), we also notice that the smaller the regularization, the smaller the step size ϵ is required in the iteration. If we pick a relatively large ϵ, the energy will not converge to its minimum, but will over-shoot around it. Therefore, to have a stable convergence, we conclude numerically that a smaller regularization parameter will require more iterations to reach its steady state.

Apart from the rate of convergence, we also note that the minimizer itself depends on the magnitude of the amount of regularization we imposed. To speed up the computations and remove the stability dependence from ϵ as discussed above, we have applied the Armijo-Goldstein rule in deciding a $\epsilon^k > \epsilon_{\min}$. Corresponding changes in the energies are plotted in Figure 6.10 (c). Considering these energies with fixed ν, we find that the larger the magnitude of the regularization we imposed, the lower the energy can achieve. However, as we increase the value of ν, it takes more iterations to reach the minimizer.

As discussed earlier near the end of Section 6.2, when the solution m^k is close to a minimizer, it is possible to reduce the amount of the regularization in $\tilde{m}$. We have implemented this idea by decreasing ν^k exponentially using $\nu_{\min} = 0.1$,

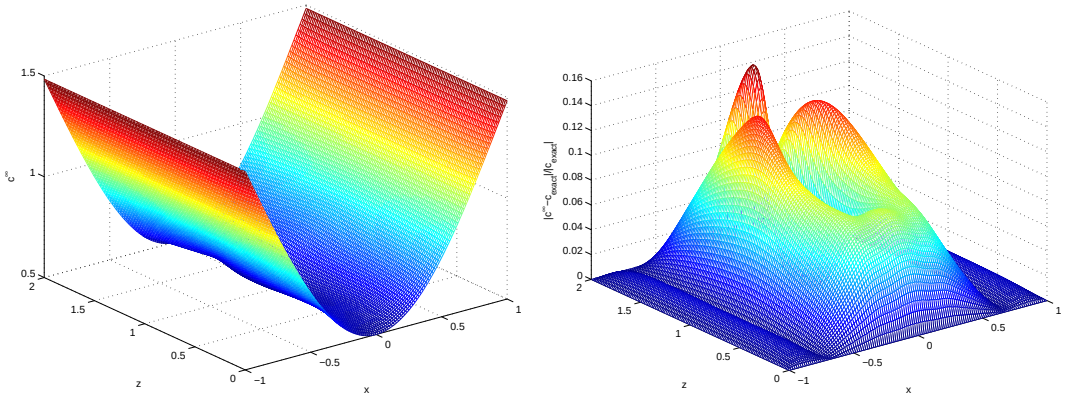


Figure 6.11: (Waveguide Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution.

$\nu_{\max} = 1$ and $\nu_0 = 0.975$. The change in the energy is shown in Figure 6.10 (c) and (d). As we can see from these two subplots, we can further lower the energy of the minimizer using a gradually reducing ν. We also plot the corresponding minimizer and the relative error in this solution in Figure 6.10 (e) and (f), respectively.

As discuss before, it may be difficult to obtain perfect measurements as required in Section 6.4. In practise, we may only have finite receivers on the ground level. Figure 6.11 shows the case where we have only one source at $(x, z) = (0, 0)$ and 5 receivers at $x = -0.2 : 0.1 : 0.2$ and $z = 2$.

6.5.3 Gaussian Model

The exact velocity model is given by

$$c(x,z) = 3 - \frac{1}{2}\exp\left(-\frac{x^2 + (z-0.5)^2}{0.5^2}\right) - \exp\left(-\frac{x^2 + (z-1.25)^2}{0.5^2}\right). \quad (6.32)$$

The number of grid points used is $257 \times 129 \times 257$ in the x-θ-z space.

The first row of Figure 6.12 shows the initial guess c^0 and the convergent

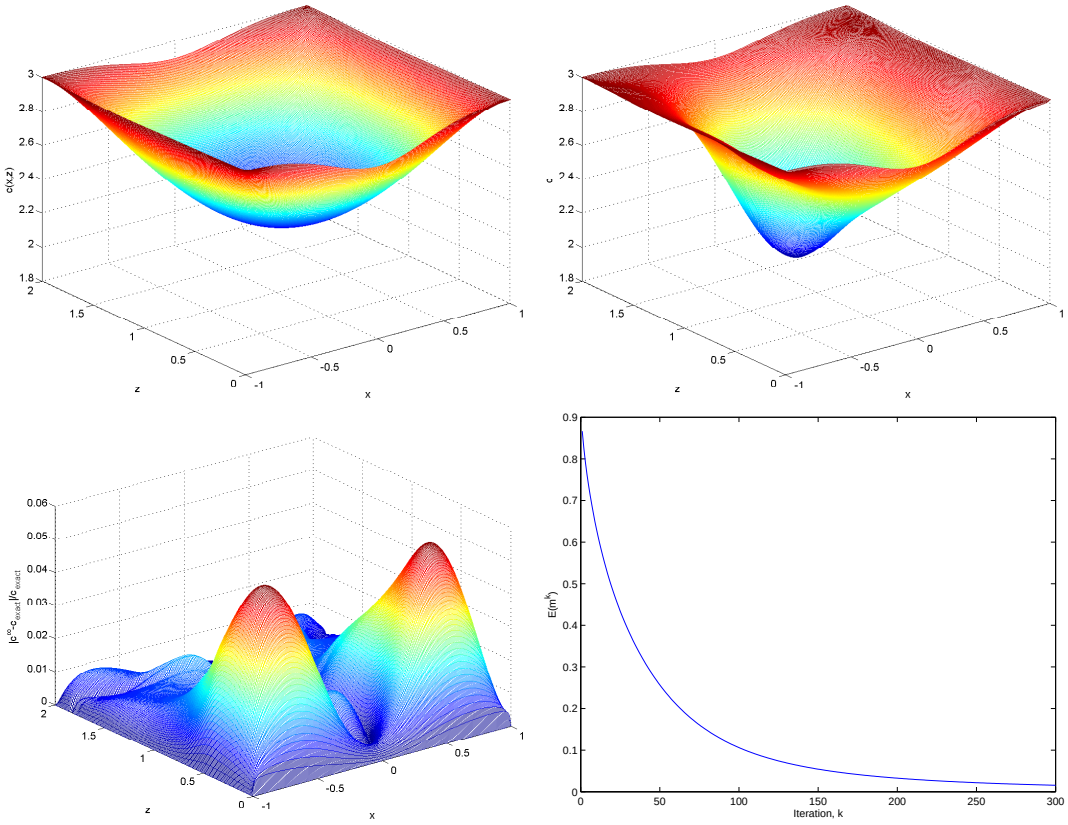


Figure 6.12: (Gaussian Model) The initial guess and final approximated c, the relative error in the solution and the convergence history of energy in semi-log scale.

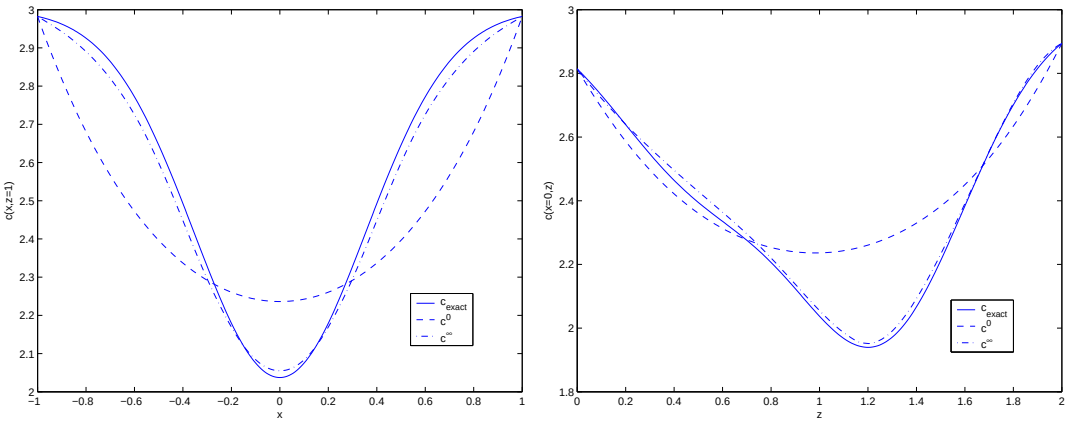


Figure 6.13: (Gaussian Model) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

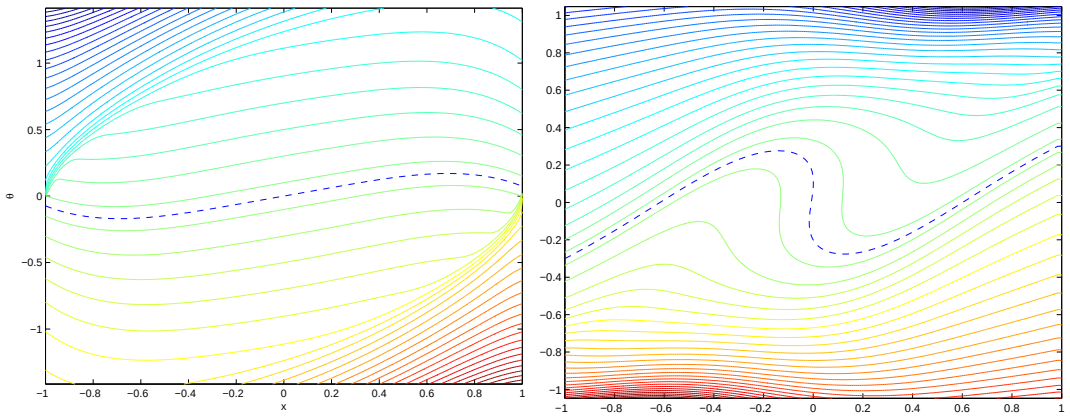


Figure 6.14: (Gaussian Model) The contour plot of $\phi(z = 2, x, \theta)$ using c^0 and the exact c, respectively.

solution c^{300}; the second row of the figure shows the error in c^{300} and the change in the energy during the iteration. Figure 6.13 shows the cross-section of the initial guess, the convergent velocity and the exact velocity; the results validate the effectiveness and the convergence of the algorithm. Figure 6.14 shows the contour plots of the level set function $\phi(z = 2, x, \theta)$ using $c = c^0$ and the exact velocity model. The dashed line represents the zero level set.

To compare with the results from the first-arrival based traveltime tomography as discussed in Chapter 5, we start from the same initial guess as illustrated in Figure 6.1 and carry out the tomography using the same physical discretization with the mesh size of 257×257 in the x-z space. Figure 6.15 shows the final converged velocity c after 10000 iterations and the convergence history, where again the fast sweeping method is used to carry out the forward simulation; Figure 6.16 shows the cross sections of the converged velocity along $z = 1$ and $x = 0$; the above computation is carried out with a single source only, namely, the traveltime information being generated from a single point source. As we can see, the Liouville formulation proposed here yields much better results than the traditional first-arrival based traveltime tomography does since it takes into account all the sources and multiple arrivals in a systematic way.

To test the robustness of the new formulation, we have added the Gaussian noise up to 5% to the measured data and carried out the tomography process. The results are shown in Figures 6.17, 6.18 and 6.19; comparing with the results without noise, the new formulation is robust and accurate.

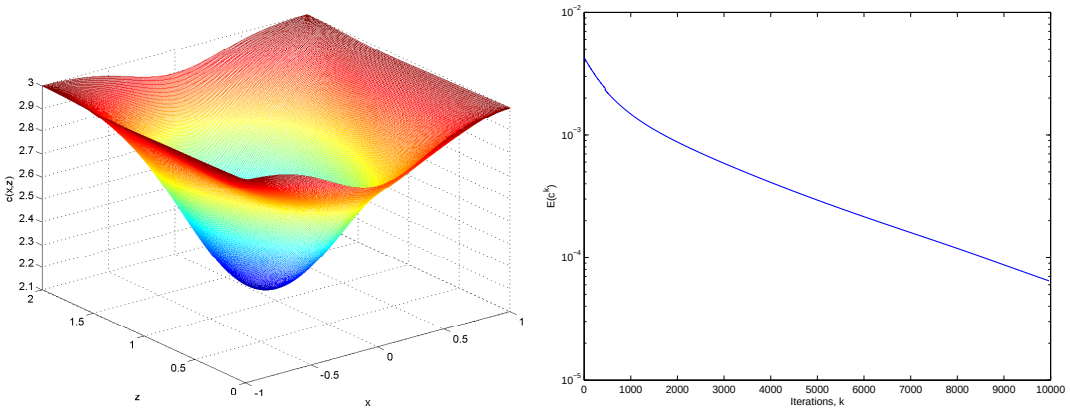


Figure 6.15: (Gaussian Model) First-arrival based traveltime tomography with a single source and multiple receivers: the final approximated c and the convergence history of energy in semi-log scale.

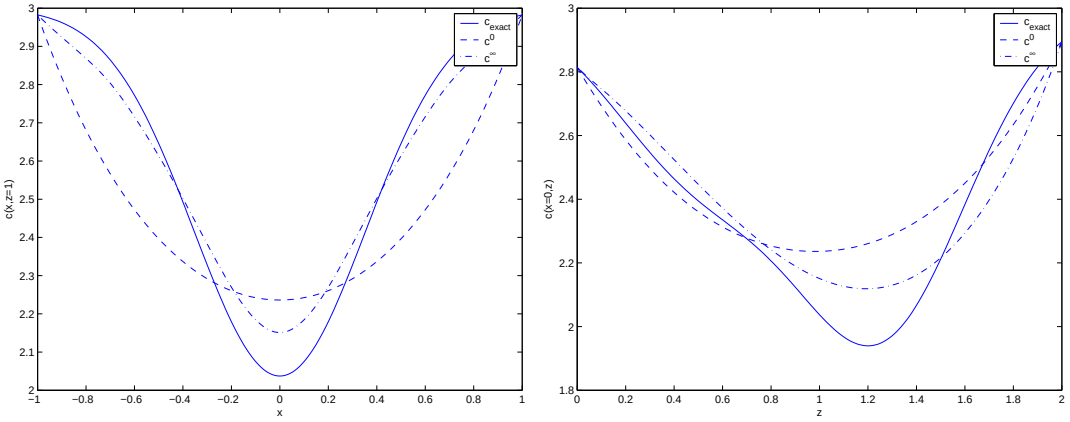


Figure 6.16: (Gaussian Model) First-arrival based traveltime tomography with a single source and multiple receivers: cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

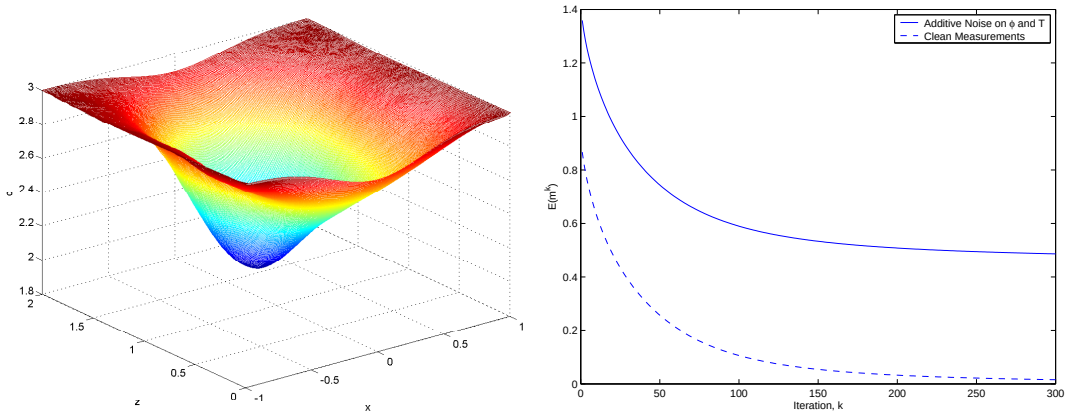


Figure 6.17: (Gaussian Model with Additive Gaussian Noise) The final approximated c and the convergence history of energy in semi-log scale.

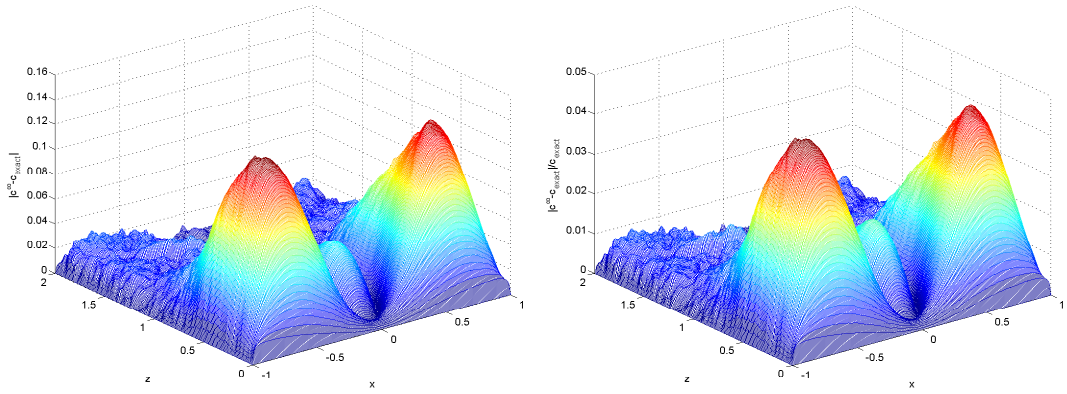


Figure 6.18: (Gaussian Model with Additive Gaussian Noise) Errors and relative errors in c^{∞}.

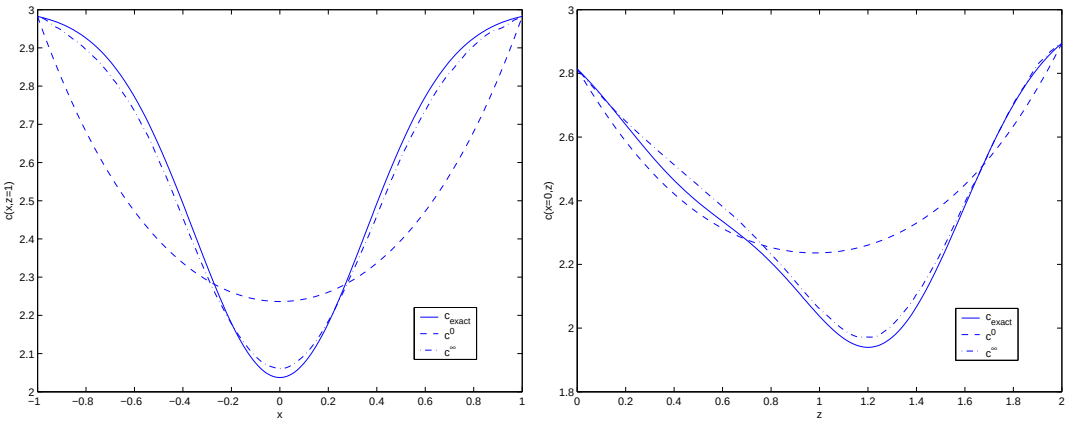


Figure 6.19: (Gaussian Model with Additive Gaussian Noise) Cross-sections of the velocity c_{exact}, c^0 and c^∞ along $x = 0$ and $z = 1$.

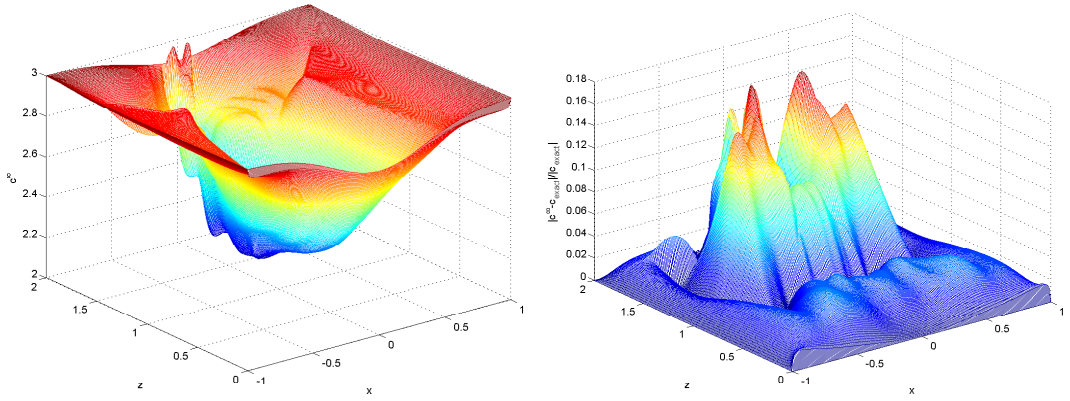


Figure 6.20: (Gaussian Model) Practical Algorithm. Inverted velocity and the corresponding relative error in the solution.

CHAPTER 7

Global Minimization of the Active Contour Model with TV-Inpainting and Two-Phase Denoising

7.1 Introduction

Image segmentation, image restoration and image inpainting are a few basic yet important areas in image processing and computer vision. Traditionally, these closely related fields were developed independently. However, the use of the level set method and variational methods in recent years started to bring all these fields together. One example is the TV-inpainting model [22]. We can perform inpainting in a desired domain while applying the ROF model [92] to remove noise from the rest of the domain using only one energy functional. Another example is the Mumford-Shah model [77] which was originally designed for image segmentation but can also be used as an image denoising tool. Extension of the Mumford-Shah model to image inpainting was also carried out in [39].

There are two interesting recent developments about the connection between different fields in image processing. We will discuss later in this paper how they link different fields in an interesting way. The first development concerns the impulse-noise removal method and the variational method for image regularization. In two recent papers by Chan, Nikolova et al. [18, 19], a two-phase method

was proposed to remove impulse-type noise. For a true image $u^*(x)$ and an observed image $f(x)$ defined in a domain Ω, impulse-type noise is defined by

$$f(x) = \begin{cases} r(x) & \text{with probability } r_0 \\ u^*(x) & \text{with probability } (1 - r_0). \end{cases} \tag{7.1}$$

As an example, for the so-called salt-and-pepper noise, $r(x)$ is simply the maximum or the minimum of the image intensity (0 and 255 for grey-scaled images). The main idea in those papers is to separate the denoising process into a noise detection phase and a noise removal phase. In the first stage, a median-type filter is applied to the observed image to detect the possible locations of the impulse noise. Then in the second phase, instead of replacing the intensity at all locations by the median intensity around a certain neighborhood, a L^1-regularization method is applied only to those locations reported in the first phase while keeping the other pixels unchanged. The resulting method was shown to be able to remove the salt-and-pepper noise efficiently even at a very high noise level (for example $r_0 = 0.75$). The main reason for its success is that this method retains those pixels that are unlikely to be polluted and maintains sharp edges in the whole image.

Another interesting development is a new model that uses variational method for image segmentation [20, 13]. The idea of the model is to minimize an energy functional consisting of a weighted TV-norm with a L^1-fidelity term. For the segmentation of a binary image, the papers showed an equivalence between the new energy functional and that of the active contour model. This relation can be used to overcome the problem of the active contour model in which the energy function is not convex. Very often the snake will be trapped in a **local** minimizer thus giving unsatisfactory segmentation results. The link between these two

energies, as demonstrated in the above papers, provides a convenient way to determine the **global** minimizer of the active contour energy.

In this chapter, we will combine the two recent advances in image processing mentioned above. This provides an efficient and unified way to perform image denoising (for both impulse-type and Gaussian-type noises), image segmentation, and image inpainting at the same time.

7.2 Two Recent Developments

7.2.1 A Two-Phase Method to Remove Impulse-Type Noise

Unlike the usual way to denoise impulse noise by applying the median-type filter to the image and replacing the image intensity everywhere, the idea in [18, 19] is to separate the denoising processing into a noise-detection phase and a noise-removal phase. Mathematically, the first phase can be formulated as determining a noise candidate set

$$\mathcal{N} = \{x \in \Omega : f(x) \neq f_{\text{MF}}(x)\} \tag{7.2}$$

in which Ω is the image domain, $f(x)$ is the observed image intensity at the pixel x, and $f_{\text{MF}}(x)$ is the intensity at x after applying a median-type filter, such as the classical median filter or the adaptive median filter. In the second phase, the following functional is minimized

$$F|_{\mathcal{N}}(u) = \int_{\mathcal{N}} \left\{ |u(x) - f(x)| + \frac{\beta}{2}[S_1(u) + S_2(u)] \right\} \tag{7.3}$$

where

$$\begin{aligned} S_1(u) &= \int_{\mathcal{V}(x)\cap(\Omega\setminus\mathcal{N})} 2\,\phi[u(x)-f(y)]dy \\ S_2(u) &= \int_{\mathcal{V}(x)\cap\mathcal{N}} \phi[u(x)-u(y)]dy\,, \end{aligned} \tag{7.4}$$

$\mathcal{V}(x)$ is the neighborhood centered at x and ϕ is an edge-preserving potential function. As seen in [18, 19], one possible choice for ϕ is

$$\phi(t) = \sqrt{t^2+\epsilon^2} \tag{7.5}$$

with a small constant ϵ. The first term in the curve-bracket is a L^1-fidelity term. The terms in the square-bracket can be interpreted as an approximation of the total variation (TV) of u.

In the simple case when the noise can be separated accurately in the first step, the fidelity term is not important. A simplification of this whole algorithm is therefore the same as an image inpainting algorithm. For example, if ROF [92] or L^2-fidelity is used instead, we arrive at the TV-inpainting [22]. That is, given an observed image f, one minimizes the following energy

$$E_1(u) = \int_\Omega |\nabla u| + \frac{1}{2}\int_\Omega \lambda(x)|u-f|^2 \tag{7.6}$$

where

$$\lambda(x) = \begin{cases} 0 & \text{if } f(x) = f_{\text{MF}}(x) \\ \lambda_\infty \simeq \infty & \text{otherwise.} \end{cases} \tag{7.7}$$

The idea of using a piecewise constant $\lambda(x)$ in TV-inpainting is not new [22]. However, it is interesting to see here the relationship between the impulse-type noise removal and image inpainting by using a piecewise constant $\lambda(x)$ determined

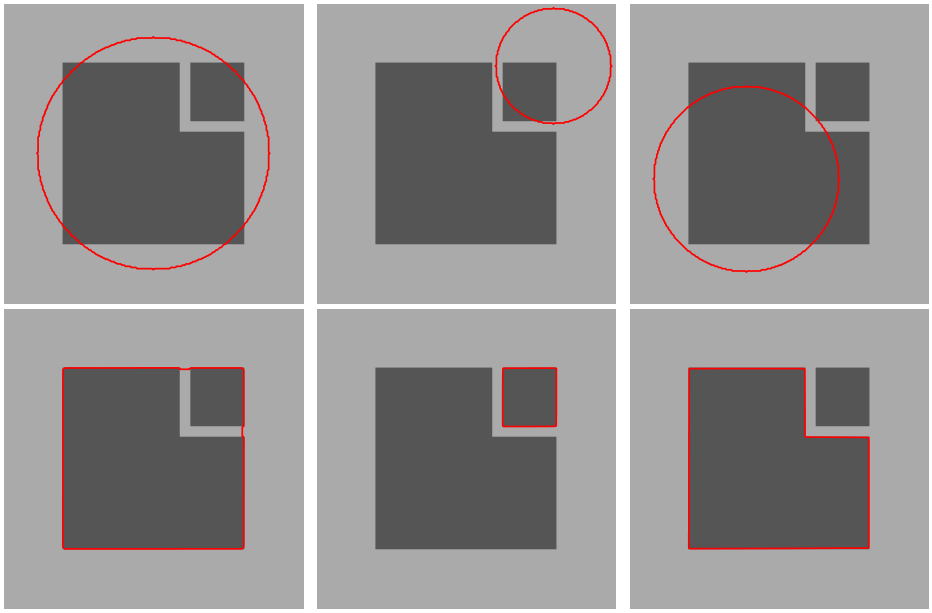


Figure 7.1: Segmentation results using the active contour model. We show different initial configurations of the snake on the first row. The corresponding segmented results using these initial conditions are shown on the second row.

by a median-type filter.

7.2.2 Global Minimizer of the Active Contour Model

In the classical active contour model, the initial guess of the segmented image plays a very important role. We show in Figure 7.1 some minimizers of the active contour model. As we can see, different initial conditions in the evolution will give different segmented region. More importantly, none of these results corresponds to the *true* segmented results, i.e. curves which separate all regions with different intensities in the whole image. One reason for these unsatisfactory results is that the minimization problem of the active contour is not convex, and therefore it is very likely that the energy minimization could be trapped into a **local** minimizer, as shown in the left most case.

Recently, a few algorithms were proposed [20, 13] to determine the **global** minimizers of some image segmentation models. In particular, an algorithm to determine the global minimizer of the active contour model based on the ROF model was given in [13]. The idea is to modify the ROF energy

$$E_{\text{ROF}}(u,\lambda) = \int_\Omega |\nabla u| + \frac{\lambda}{2}\int_\Omega |u-f|^2 \tag{7.8}$$

by first replacing the TV-norm by a weighted TV-norm and then, more importantly, changing the measure in the fidelity term from L^2-norm to L^1-norm. This gives

$$E_2(u,\lambda) = \int_\Omega \tilde{g}(f)|\nabla u| + \lambda\int_\Omega |u-f| \tag{7.9}$$

in which

$$\tilde{g}(f) = \frac{1}{1+\beta|\nabla f|^2}\,. \tag{7.10}$$

As pointed out in [13], if u is the characteristic function of a set Ω_C with boundary given by the curve C (i.e. $u = \mathbf{1}_{\Omega_C}$), the minimizer of the above energy E_2 is the same as the minimizer of the active contour energy

$$E_{\text{AC}}(C) = \int_C \tilde{g}(f)ds \tag{7.11}$$

with f approximated (in the sense of L^1) by a binary function of a region Ω_C.

Numerically, the minimization problem (7.9) is convex. This means that the method of gradient descent will converge to a unique minimizer, i.e. the global minimum of the energy function, independent of the initial condition. The significance of this equivalence is that by minimizing (7.9), one can determine the global minimizer of the active contour model (7.11) without the danger of trapping into any local minimum and the uncertainty in picking an initial configuration of the

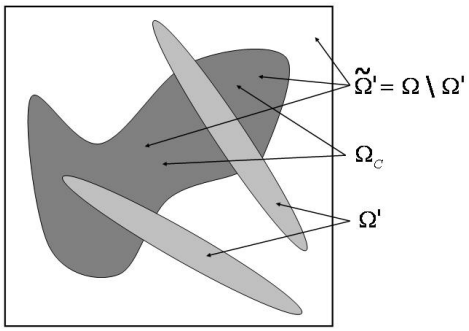


Figure 7.2: Problem setting. Definition of the set Ω' (domain for inpainting), $\tilde{\Omega}'$ (compliment of Ω') and Ω_C (domain bounded by the curve C).

snake.

7.3 The New Energy

7.3.1 The Energy

Here, we propose a new model to combine the two recent developments in image processing. Given an observed image f, we minimize the energy

$$E(u) = \int_\Omega g(f)|\nabla u| + \int_\Omega \lambda(x)|u - f|\,. \tag{7.12}$$

This energy is similar to (7.9), except that $\lambda(x)$ is now changed to a function in space and the weight in the weighted TV-norm is also modified. The function $\lambda(x)$ has the following properties.

$$\lambda(x) = \begin{cases} 0 & \text{TV-inpainting} \\ \lambda_0 & \text{Denoising} \\ \lambda_\infty \simeq \infty & \text{Unchanged.} \end{cases} \tag{7.13}$$

In the subdomain for image inpainting, $f(x)$ (and therefore $\tilde{g}(f)$) might not be known. We therefore set $g(f) = 1$, or $\beta = 0$. For the rest of the domain, we keep $g(f) = \tilde{g}(f)$. In other words, we have

$$g(f) = \begin{cases} 1 & \text{if } \lambda(x) = 0 \\ \tilde{g}(f) \equiv (1 + \beta|\nabla f|^2)^{-1} & \text{otherwise.} \end{cases} \tag{7.14}$$

Mathematically, minimizing the above energy (7.12) is the same as

$$\min_u \int_\Omega g(f)|\nabla u| \tag{7.15}$$

such that

$$\int_\Omega \omega(x)|u - f| = \text{ constant} \tag{7.16}$$

for some weighted function $\omega(x)$. Therefore, the way to determine $\lambda(x)$ is equivalent to the way to spread the error in approximating the observed image f.

Here we give some suggestions in picking such $\lambda(x)$ and also provide a variation in using the above minimization algorithm.

For the salt-and-pepper noise, the following $\lambda(x)$ works efficiently. We define $d(x)$ to be the difference in the intensities between the original image $f(x)$ and the modified image after applying the median-type filter $f_{\text{MF}}(x)$, i.e.

$$d(x) = |f(x) - f_{\text{MF}}(x)|\,. \tag{7.17}$$

Then one can set

$$\lambda_1(x) = \begin{cases} \lambda_\infty & \text{if } d(x) = 0 \text{ and } x \notin \Omega' \\ 0 & \text{otherwise} \end{cases} \tag{7.18}$$

where $\Omega' \subset \Omega$ is a given subdomain for doing image inpainting and is characterized by an user predefined mask function. This means that if x is in the inpainting domain Ω' or if the noise-detector detects that the image at x is polluted (therefore $f(x)$ will be different from the intensity after applying the median-type filter $f_{\text{MF}}(x)$), then the intensity at x will be modified by a TV-type regularization. Otherwise, the intensity at that location will remain unchanged.

If the impulse noise is random-valued instead, one can use a similar $\lambda(x)$

$$\lambda_2(x) = \begin{cases} \lambda_0 & \text{if } d(x) = 0 \text{ and } x \notin \Omega' \\ 0 & \text{otherwise} \end{cases} \tag{7.19}$$

with $\lambda_0 << \lambda_\infty$.

For the Gaussian-type noise, one can simply use

$$\lambda_3(x) = \begin{cases} \lambda_0 & \text{if } x \notin \Omega' \\ 0 & \text{otherwise.} \end{cases} \tag{7.20}$$

In the case when the type of noise is not known *a priori*, one can try to minimize (7.12) iteratively. More specifically, given the observed image $u_0 = f$, for $m = 1, \cdots, m_{\max}$, one minimizes

$$E(u_m) = \int_\Omega g(u_{m-1})|\nabla u_m| + \int_\Omega \lambda_4(x)|u_m - u_{m-1}| \tag{7.21}$$

iteratively with

$$\lambda_4(x) = \begin{cases} \lambda_0 & \text{if } d(x) \leq d^* \text{ and } x \notin \Omega' \\ 0 & \text{otherwise} \end{cases} \tag{7.22}$$

where d^* is a threshold in the intensity difference function $d(x) \equiv |u_{m-1} - (u_{m-1})_{\text{MF}}|$.

7.3.2 The Link Between Active Contour for Segmentation, Denoising and TV-Inpainting

The relations between the minimization of the energy functional (7.12), the active contour model and the TV-inpainting model are explained here.

Assuming $\Omega' = \{x \in \Omega : \lambda(x) = 0\}$ is the subdomain for inpainting (notice that $\mathcal{N} \subset \Omega'$) and $\tilde{\Omega}' = \Omega \setminus \Omega'$, we have

$$\begin{aligned} E(u) &= \int_{\tilde{\Omega}'} g(f)|\nabla u| + \int_{\tilde{\Omega}'} \lambda_0 |u - f| + \int_{\Omega'} |\nabla u| \\ &= E^1(u) + E^2(u) \end{aligned} \tag{7.23}$$

where

$$\begin{aligned} E^1(v) &= \int_{\tilde{\Omega}'} g(f)|\nabla v| + \int_{\tilde{\Omega}'} \lambda_0 |v - f| \\ E^2(w) &= \int_{\Omega'} |\nabla w| \end{aligned} \tag{7.24}$$

with $v : \tilde{\Omega}' \to [u_{\min}, u_{\max}]$ and $w : \Omega' \to [u_{\min}, u_{\max}]$. So minimizing $E(u)$ is the same as

$$\min_v E^1(v) + \min_w E^2(w)\,, \tag{7.25}$$

and the minimizer of $E(u)$ is given by $u = \mathbf{1}_{\tilde{\Omega}'}(x) \cdot v + \mathbf{1}_{\Omega'}(x) \cdot w$ in which $\mathbf{1}_{\tilde{\Omega}'}$ is the characteristic function of the set $\tilde{\Omega}'$.

First we consider the energy $E^1(v)$. If Ω_C is a set in $\tilde{\Omega}'$ whose boundary is denoted by C and if the minimizer of $E^1(v)$ is given by $v = \mathbf{1}_{\Omega_C}$, then we have

$$\begin{aligned} E^1(v) &= \int_{\tilde{\Omega}'} g(f)|\nabla \mathbf{1}_{\Omega_C}| + \int_{\tilde{\Omega}'} \lambda_0 |\mathbf{1}_{\Omega_C} - f| \\ &= \int_C g(f) ds + \int_{\tilde{\Omega}'} \lambda_0 |\mathbf{1}_{\Omega_C} - f|\,. \end{aligned} \tag{7.26}$$

Therefore, minimizing $E^1(v)$ in the subdomain $\tilde{\Omega}'$ in the case of a binary observed image is equivalent to minimizing the active contour energy in $\tilde{\Omega}'$, given by

$$\min_C \int_C g(f)ds \tag{7.27}$$

while

$$\begin{gathered}\text{approximating } f \text{ (in the } L^1 \text{ sense) in } \tilde{\Omega}' \\ \text{by a binary function of the set/region } \Omega_C.\end{gathered}$$

For the energy $E^2(w)$ defined in the complement, Ω', we have

$$E^2(w) = \int_{\Omega'} |\nabla w| \tag{7.28}$$

together with the boundary condition $w|_{\partial\Omega'} = v|_{\partial\Omega'}$ where v is the minimizer of $E^1(v)$. In the case when v is binary on $\partial\Omega'$, we have $w = \mathbf{1}_{\Omega_{C'}}$ again. This gives

$$E^2(w) = \int_{\Omega'} |\nabla \mathbf{1}_{\Omega_{C'}}| = \int_{C'} ds\,. \tag{7.29}$$

This implies that when the boundary $\partial\Omega'$ is binary valued, minimizing $E^2(w)$ in Ω' is equivalent to

$$\min_{C'} \int_{C'} ds \tag{7.30}$$

while the end points of C' are fixed on $\partial\Omega'$. Further analysis on the behavior of TV-inpainting can be found in [21].

7.4 Numerical Method

To minimize the above energy, one can use the method of gradient descent. The Euler-Lagrange equation of the energy functional (7.12) is given by

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{g(x)}{|\nabla u|} \nabla u \right) - \lambda(x) \frac{u - f}{|u - f|} \,. \tag{7.31}$$

Here we give the details of the algorithm in updating u^{n+1} by solving (7.31) using the Alternative Direction Explicit (ADE) technique. This numerical scheme is second order accurate in time for linear equations and fully explicit but yet unconditionally stable for any time-step Δt. Given the observed image f and an intermediate approximation u^n, we use the following procedures.

Algorithm:

1. Define $v^n = u^n$ and $w^n = u^n$.

2. Compute

$$\begin{aligned}
h_{i,j}^{x\pm} &= \{1 + \beta[(D_x^{\pm} f_{i,j})^2 + (D_y^0 f_{i,j})^2]\}^{-1} [(D_x^{\pm} u_{i,j}^n)^2 + (D_y^0 u_{i,j}^n)^2 + \delta^2]^{-1/2} \\
h_{i,j}^{y\pm} &= \{1 + \beta[(D_x^0 f_{i,j})^2 + (D_y^{\pm} f_{i,j})^2]\}^{-1} [(D_x^0 u_{i,j}^n)^2 + (D_y^{\pm} u_{i,j}^n)^2 + \delta^2]^{-1/2} \\
\alpha_{i,j} &= \frac{\Delta t}{2} \left(h_{i,j}^{x+} + h_{i,j}^{x-} + h_{i,j}^{y+} + h_{i,j}^{y-} \right) + \frac{\lambda_{i,j} \Delta t}{2\sqrt{(u_{i,j}^n - f_{i,j})^2 + \epsilon^2}} \,.
\end{aligned}$$

   where $D_x^{\pm}$, D_x^0, $D_y^{\pm}$ and D_y^0 are the standard forward, backward and central difference operators in the x- and y-directions respectively.

3. For $i = 1, 2, \cdots, n_x$ and $j = 1, 2, \cdots, n_y$, compute

$$v_{i,j}^{n+1} = \frac{1}{(1 + \alpha_{i,j})} [(1 - \alpha_{i,j}) v_{i,j}^n + \Delta t (h_{i,j}^{x+} v_{i+1,j}^n$$

$$+h^{x-}_{i,j}v^{n+1}_{i-1,j} + h^{y+}_{i,j}v^{n}_{i,j+1} + h^{y-}_{i,j}v^{n+1}_{i,j-1}) + \frac{\Delta t\lambda_{i,j}f_{i,j}}{\sqrt{(v^n_{i,j}-f)^2+\epsilon^2}}] \tag{7.32}$$

4. For $i = n_x, n_x-1, \cdots, 1$ and $j = n_y, n_y-1, \cdots, 1$, compute

$$\begin{aligned} w^{n+1}_{i,j} &= \frac{1}{(1+\alpha_{i,j})}[(1-\alpha_{i,j})w^n_{i,j} + \Delta t(h^{x+}_{i,j}w^{n+1}_{i+1,j} \\ &+h^{x-}_{i,j}w^{n}_{i-1,j} + h^{y+}_{i,j}w^{n+1}_{i,j+1} + h^{y-}_{i,j}w^{n}_{i,j-1}) \\ &+\frac{\Delta t\lambda_{i,j}f_{i,j}}{\sqrt{(w^n_{i,j}-f)^2+\epsilon^2}}] \end{aligned} \tag{7.33}$$

5. Compute

$$u^{n+1}_{i,j} = \frac{1}{2}\left(v^{n+1}_{i,j} + w^{n+1}_{i,j}\right) . \tag{7.34}$$

7.5 Examples

In the following examples, we use $u^0(x,y) = 0$ as the initial condition for the Euler-Lagrange equation. Unlike the classical active contour/snake model, different initial guesses used here will give the same **global** minimizer of the segmentation model in the case of binary images.

As discussed before, the ADE scheme has no stability condition imposed on Δt, and therefore we used $\Delta t = 100$ in all of the examples below.

7.5.1 Example 1

The true image used in this example has 256×256 pixels and is shown on the left in Figure 7.3. The user predefined mask is shown on the right. The dark region is the domain Ω' where we want to perform TV-inpainting. Figure 7.4 shows the

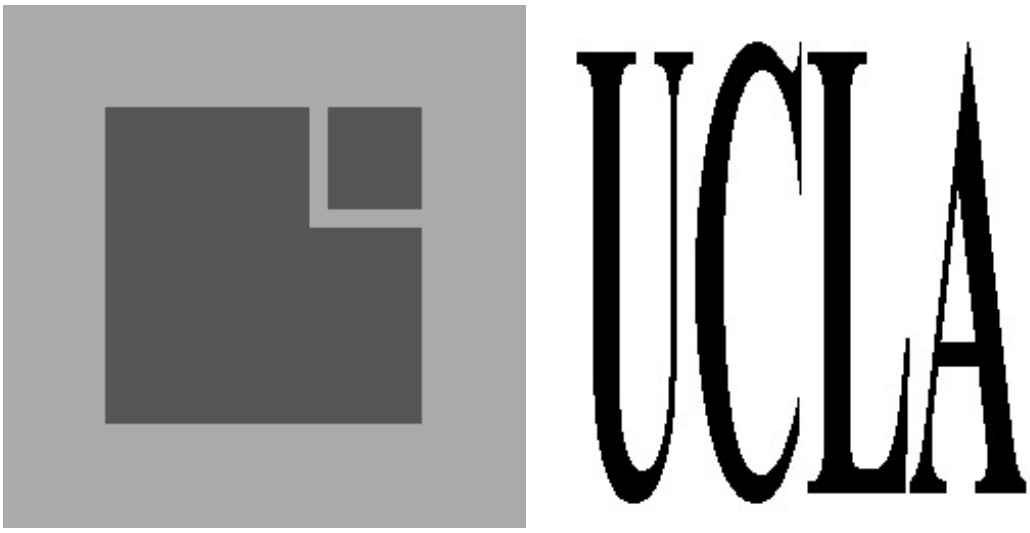


Figure 7.3: The original true image and the user defined mask.

noisy versions of the true image.

Figure 7.5 shows the denoised together with the segmented results of the image with 75% salt-and-pepper noise. The graphs show the intensity contour of the minimizer u. The salt-and-pepper noise is completely removed from the image. The red curves on the graph are the boundaries of the segmented regions. Unlike the minimization of the active contour model, we can now easily reach

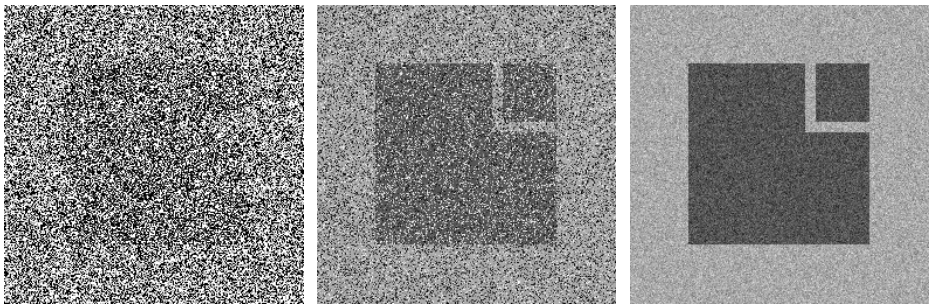


Figure 7.4: The original image with 75% salt-and-pepper noise, 50% random-valued impulse noise and additive Gaussian noise ($\sigma = 20$) respectively.

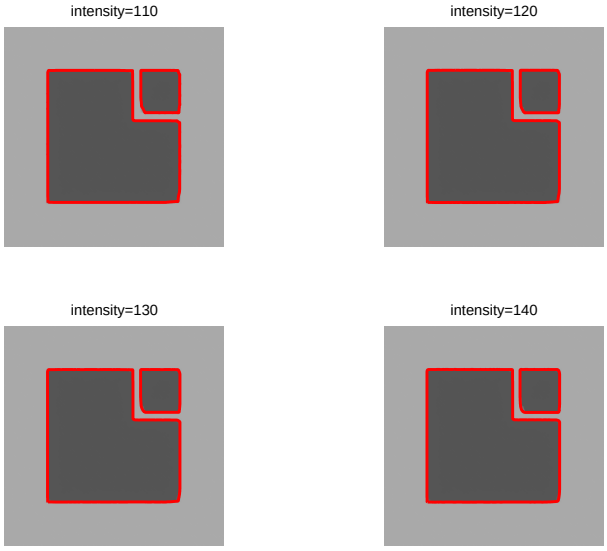


Figure 7.5: (Salt-and-Pepper) The minimizer for the energy (7.12) without an extra mask.

the global minimum of the energy for this binary image regardless of the initial condition of the Euler-Lagrange equation. In the case of denoising together with image inpainting, the segmented results are shown in Figure 7.6. Again, the salt-and-pepper noise is removed completely from the image and we are able to fill in the missing part of the image using only one energy function. Figure 7.7 and 7.8 show the denoised, inpainted and segmented results when the random-valued impulse noise is added to the original true image. Figure 7.9 and 7.10 show the case when the additive Gaussian noise is added to the true image.

7.5.2 Example 2

The true image of Elaine used in this example has 512×512 pixels and is shown on the left of Figure 7.11. On the right, we give the predefined mask. 75% salt-and-pepper noise and 50% random-valued impulse noise are added to the original image and these observed images are shown on the left and middle of Figure 7.12,

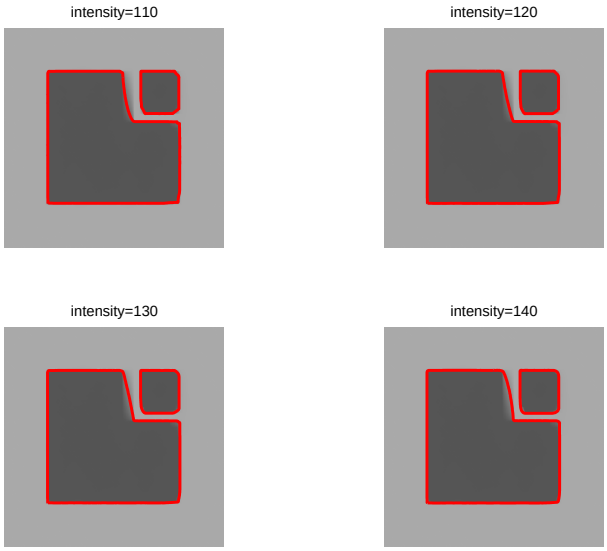


Figure 7.6: (Salt-and-Pepper) The minimizer for the energy (7.12) with an extra mask.

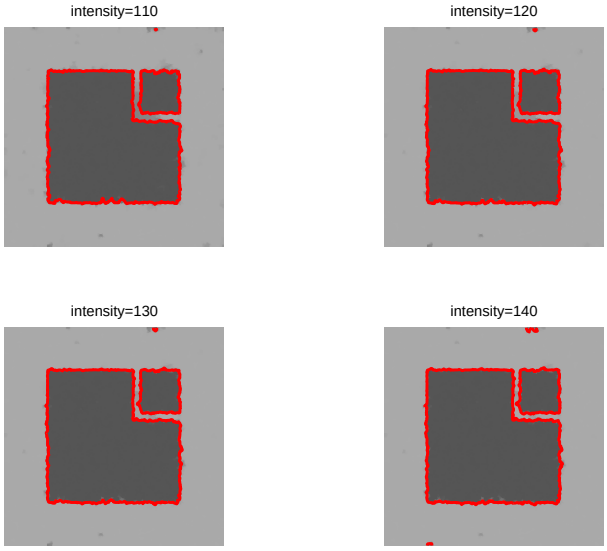


Figure 7.7: (Random-valued Impulse) The minimizer for the energy (7.12) without an extra mask.

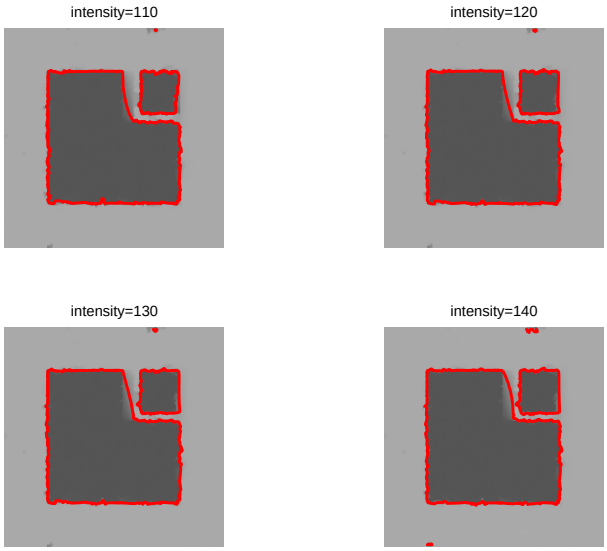


Figure 7.8: (Random-valued Impulse) The minimizer for the energy (7.12) with an extra mask.

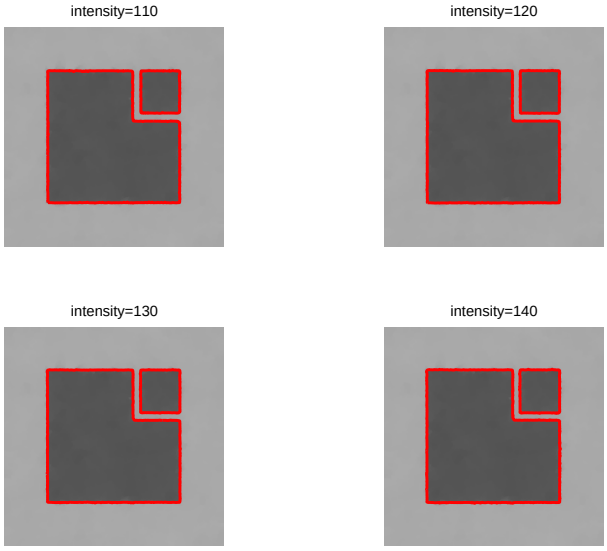


Figure 7.9: (Additive Gaussian) The minimizer for the energy (7.12) without an extra mask.

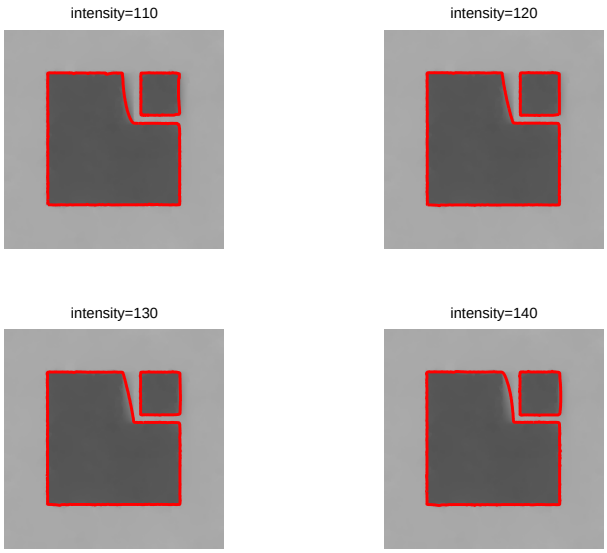


Figure 7.10: (Additive Gaussian) The minimizer for the energy (7.12) with an extra mask.

Figure 7.11: The original true image and the user defined mask.

Figure 7.12: The original image with 75% salt-and-pepper noise, 50% random–valued impulse noise and additive Gaussian noise ($\sigma = 20$) respectively.

Figure 7.13: (Salt-and-Pepper) The minimizer for the energy (7.12) without (left) and with (right) an extra mask.

Figure 7.14: (Random-valued impulse) The minimizer for the energy (7.12) without (left) and with (right) an extra mask.

Figure 7.15: (Additive Gaussian) The minimizer for the energy (7.12) without (left) and with (right) an extra mask.

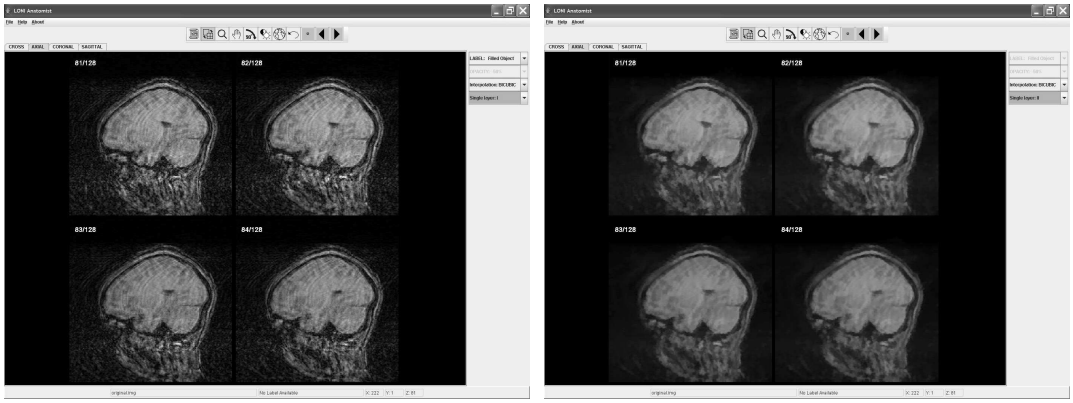


Figure 7.16: Some noisy brain MRI images and their corresponding denoised MRI images.

respectively. On the right hand side, we show the image with additive Gaussian noise with standard deviation $\sigma = 20$. The minimizers of the energy functional (7.12) for these noisy images are shown in Figure 7.13 to 7.15. Figure 7.13 shows the results for the salt-and-pepper noise without (left) and with (right) an user defined mask function. Results for the random-valued impulse noise and the additive Gaussian noise are given in Figure 7.14 and 7.15.

7.5.3 Example 3

Figures 7.16 show the denoising results of a 3D brain MRI image. The number of voxels are $128 \times 256 \times 256$. The computational time is approximately 354 mins.

CHAPTER 8

An Adaptive Level Set Method for Stefan Problems

8.1 Introduction

Crystal growth is a classical problem for phase transition. To experimentally visualize this transition process, we can drop a small solid seed into a bath of undercooled liquid. This seed will then expand and grow into a finger-like shape, or the so-called dendritic shape. Usually, this final shape depends strongly on the bath environment. For example, a slightly change in the liquid temperature will completely change the way how the crystal evolves. Understanding this transition process is important. For instant, in making a metal alloy, how the crystal evolves could affect the conductivity or the strength of the final alloy. A small change in the growth condition could result in a poor quality of alloy. It is, therefore, necessary to have a complete study in relating the growth process to the bath environment.

Numerical modeling provides an efficient way to study the crystal growth. Rather than doing a whole series of *real* experiment, we can understand how a particular parameter changes the growth process by varying the corresponding value in a computer program. Moreover, we can precisely control the growth environment in these *imaginary* experiments in computer, which is very chal-

lenging in a laboratory. However, there are still challenges in simulating the dendritic growth in a computer. Main difficulties include the numerical method should be able to solve the growth equation on an arbitrary domain with irregular boundaries; to evolve the interface between the solid phase and the liquid phase according to some physical laws; to solve the problem in a reasonable time, and most importantly, to easily handle the topological changes of the interface.

The level set method [96, 23, 62, 47, 46] has been successfully introduced recently to model the growth of a crystal. The main idea is to represent the crystal interface by the zero level set of a level set function. This method can naturally handle any change in the topology of the solid boundary. However, one drawback of this method is that this finite difference method introduces excessive numerical dissipations in the calculations. In other words, to have an accurate solution, one may need to use a very fine mesh. This is not efficient in the sense that many grid points are far away from the interface and any numerical solution at these locations has little contribution to the motion of the zero level set.

This chapter aims to improve the computational efficiency of the above approach by cooperating the level set method with an adaptive moving grid method [64, 53, 108, 107]. Since this adaptive approach is based on structured quadrilateral grids, we can directly apply any standard finite difference method in solving all physical laws and the level set equation.

8.2 The Quasi-steady Stefan Problem

In this section, we briefly summarize the classical Stefan problem. A full description can be found in, for example, [23]. The classical Stefan problem is given

by

$$\begin{aligned} \frac{\partial T}{\partial t} &= k_s \nabla^2 T \quad \text{in} \quad \Omega_s \\ \frac{\partial T}{\partial t} &= k_l \nabla^2 T \quad \text{in} \quad \Omega_l \,, \end{aligned} \tag{8.1}$$

where T denotes the temperature in both the crystal Ω_s and the supercooled liquid Ω_l. These two phases are separated by the interface $\Sigma = \phi^{-1}(0)$ which is defined by the zero level set of a function ϕ. k_s and k_l are related to volumetric heat capacities and thermal diffusivities of the materials. On the interface Σ, we impose the Gibbs-Thomson relation for the temperature,

$$T = -\sigma\kappa - \mu v_n \,, \tag{8.2}$$

where σ is the surface tension coefficient, κ is the curvature of the interface, and μ is the molecular kinetic coefficient. v_n is the normal velocity of the interface motion and is defined by the jump in the normal derivative of the temperature across the interface,

$$v_n = [\mathbf{n} \cdot \nabla T] \text{ on } \Sigma \,, \tag{8.3}$$

where $[.]$ is defined as the jump across the interface, i.e. $[f] = f_s|_\Sigma - f_l|_\Sigma$. To update the location of the interface, we use this normal velocity and solve the following level set equation

$$\phi_t + v_n |\nabla \phi| = 0 \text{ in } \Omega \,. \tag{8.4}$$

To further simplify the problem, we assume that the diffusion coefficients k_s and k_l are large. This reduces the classical Stefan problem to the following

quasi-steady Stefan problem

$$\begin{aligned} \nabla^2 T = 0 \quad &\text{in} \quad \Omega \\ T = -\sigma\kappa - \mu v_n \quad &\text{on} \quad \Sigma \\ v_n = [\mathbf{n} \cdot \nabla T] \quad &\text{on} \quad \Sigma \\ \phi_t + v_n |\nabla \phi| = 0 \quad &\text{in} \quad \Omega\,. \end{aligned} \tag{8.5}$$

8.3 Numerical Method

In this chapter, we will focus only on two-dimensional cases. It is relatively straightforward to apply the same idea to higher dimensions. The numerical procedures compose of the following steps:

1. Adapt the grid according to the level set function ϕ [64, 107].
2. Solve the Laplace equation with the Gibbs-Thomson relation imposed on the crystal interface [80, 47].
3. Update the location of the interface by solving the level set equation.

In the following subsections, we explain each of these steps in detail.

8.3.1 Grid Adaptation

The adaptivity of the grid can be interpreted as a time dependent map between the physical domain (x, y) and the computational domain (ξ, η). Because the computational domain is fixed, we can also treat (ξ, η) as the index used in discretizing the physical space. This means $(\xi, \eta) \in \{1, 2, \ldots, n_\xi\} \times \{1, 2, \ldots, n_\eta\}$ with $\Delta\xi = \Delta\eta = 1$. We denote this desired mapping as $x = x(\xi, \eta)$ and $y = y(\xi, \eta)$.

Given a level set function ϕ^n defined on the grids $(x_{i,j}^n, y_{i,j}^n)$ at time t^n, we want to find a new set of structured grids $(x_{i,j}^{n+1}, y_{i,j}^{n+1})$ with grid points concentrated around the zero level set of ϕ. A simple idea is to minimize the following so-called Winslow's functional defined in the computational domain

$$\min_{(x,y)} E(x(\xi,\eta), y(\xi,\eta)) = \frac{1}{2}\int \omega \left(|\nabla_{\xi,\eta} x|^2 + |\nabla_{\xi,\eta} y|^2\right) d\xi d\eta \,, \tag{8.6}$$

where ω is called a monitor function which controls how the grids will be adapted. Numerically, it is found that

$$\omega(\phi) = 1 + \alpha \exp(-\phi^2/\sigma^2) \,, \tag{8.7}$$

with $\alpha = 10$ and $\sigma = 0.1$, is a reasonable choice. This monitor function is symmetric in $\phi = 0$, meaning that the grid points are symmetrically distributed around the zero level set. This function has its largest value when $\phi = 0$, therefore the concentration of the grid points decreases away from the zero level set. In [107], the authors used a Heaviside level set function instead of a signed distance level set function. They proposed

$$\omega(\psi) = \sqrt{1 + |\nabla_{x,y}\psi|^2} \,, \tag{8.8}$$

with $\psi = \text{sign}(\phi)$. In our case, the monitor function is chosen in a more natural way such that the distance property of the level set function is incorporated in the grid adaptation.

To minimize the above functional (8.6), we use the method of variations and derive the corresponding Euler-Lagrange equations

$$\nabla_{\xi,\eta} \cdot (\omega \nabla_{\xi,\eta} x) \;\; = \;\; 0$$

$$\nabla_{\xi,\eta} \cdot (\omega \nabla_{\xi,\eta} y) \;=\; 0\,. \tag{8.9}$$

These equations are elliptic and uncoupled. Therefore, they can be easily solved using any standard iterative scheme like SOR or PCG.

8.3.2 Coordinate Transformation

The next step in the above algorithm is to solve the Laplace equation in this new set of grid points (x^{n+1}, y^{n+1}). This can be done by the following coordinate transformation,

$$\begin{aligned} dx &= A d\xi + B d\eta \\ dy &= M d\xi + N d\eta\,, \end{aligned} \tag{8.10}$$

where

$$A = \frac{\partial x}{\partial \xi}\,,\; B = \frac{\partial x}{\partial \eta}\,,\; M = \frac{\partial y}{\partial \xi} \text{ and } N = \frac{\partial y}{\partial \eta} \tag{8.11}$$

are determined from the local grid information. Using this transformation and defining Δ to be the Jacobian of this transformation, we rewrite the gradient operator in the Cartesian coordinates to

$$\nabla_{x,y} = \frac{1}{\Delta}\left(-M\frac{\partial}{\partial \eta} + N\frac{\partial}{\partial \xi}, -B\frac{\partial}{\partial \xi} + A\frac{\partial}{\partial \eta}\right)\,. \tag{8.12}$$

Therefore, the Laplacian operator is replaced by the Beltrami operator given by

$$\begin{aligned} \nabla^2_{x,y} T \;=\; & \frac{1}{\Delta}\left[\frac{\partial}{\partial \xi}\left(\frac{(B^2+N^2)T_\xi - (AB+MN)T_\eta}{\Delta}\right)\right. \\ & \left. + \frac{\partial}{\partial \eta}\left(\frac{(A^2+M^2)T_\eta - (AB+MN)T_\xi}{\Delta}\right)\right]\,. \end{aligned} \tag{8.13}$$

Unlike the standard five-point difference for the Laplacian, the numerical approximation of the Beltrami operator at $T(\xi_i, \eta_j)$ uses a linear combination of the following 9 grid points centered at (ξ_i, η_j), i.e. $\{T_{i,j}$, $T_{i-1,j-1}$, $T_{i,j-1}$, $T_{i+1,j-1}$, $T_{i-1,j}$, $T_{i+1,j}$, $T_{i-1,j+1}$, $T_{i,j+1}$, $T_{i+1,j+1}\}$.

All computations away from the interface is straightforward. For grid points near the interface, however, we need to pay spacial care. Since we have a boundary condition imposed on the interface, we apply the Ghost Fluid Method [80, 47] to those cells near the zero level set. For example, consider the one-dimensional case where $\phi_{i-1}\phi_i > 0$ and $\phi_i\phi_{i+1} < 0$, i.e. the interface is located between the grid point at x_i and x_{i+1} but not between x_{i-1} and x_i. By the Ghost Fluid Method, the **Laplacian** of T at x_i can be approximated by

$$\nabla_h^2 T_i = \frac{T_{i-1} - 2T_i + T_{i+1}^*}{\Delta x^2}, \tag{8.14}$$

where T_{i+1}^* is the value extrapolated from $T(\phi^{-1}(0))$, T_i, and/or T_{i-1}.

In the two dimensional case with the Cartesian coordinates, we only need to check the above condition on $\phi_{i,j}$ with the 4 neighboring grid points $(x_{i\pm1}, y_j)$ and $(x_i, y_{j\pm1})$. However, with the adaptive grids, we need to consider a total of 8 points $(\xi_{i\pm1}, \eta_j)$, $(\xi_i, \eta_{j\pm1})$ and $(\xi_{i\pm1}, \eta_{j\pm1})$.

8.3.3 Temperature Extension

After we compute the temperature in both domains Ω_s and Ω_l, we need to compute the jump of their normal derivatives across the interface. This can be done using the following extension algorithm [3, 46].

1. Define $T^1 = \chi_{\Omega_1} T$ and $T^2 = \chi_{\Omega_2} T$, where $\chi_{\{\}}$ is the characteristic function and is defined to be 1 if (x, y) is in the subscripted domain and is 0 otherwise.

2. Define $T_n^1 = \chi_{\Omega_1}(\vec{n} \cdot \nabla T)$, $T_n^2 = \chi_{\Omega_2}(\vec{n} \cdot \nabla T)$.

3. Define $T_{nn}^1 = \chi_{\Omega_1}(\vec{n} \cdot \nabla T_n)$, $T_{nn}^2 = \chi_{\Omega_2}(\vec{n} \cdot \nabla T_n)$.

4. Extend T_{nn}^1 and T_{nn}^2 to Ω_2 and Ω_1, respectively, by solving

$$\begin{aligned} (T_{nn}^1)_\tau + H(\phi)\vec{n} \cdot \nabla T_{nn}^1 &= 0 \\ (T_{nn}^2)_\tau - H(-\phi)\vec{n} \cdot \nabla T_{nn}^2 &= 0\,, \end{aligned} \tag{8.15}$$

   where $H(.)$ is the Heaviside function.

5. Extend T_n^1 and T_n^2 to Ω_2 and Ω_1, respectively, by solving

$$\begin{aligned} (T_n^1)_\tau + H(\phi)(\vec{n} \cdot \nabla T_n^1 - T_{nn}^1) &= 0 \\ (T_n^2)_\tau - H(-\phi)(\vec{n} \cdot \nabla T_n^2 - T_{nn}^2) &= 0\,. \end{aligned} \tag{8.16}$$

6. Compute $v_n = T_n^1 - T_n^2$.

In [3, 46], the above equations are solved only for a few $\Delta\tau$ steps in the artificial time τ-direction. Due to the CFL restriction, the computational speed is relatively slow. To speed up the computation, we determine directly the steady state solution of the above equations by considering the corresponding time-independent version. This implies that equations (8.15) and (8.16) are replaced by

$$\begin{aligned} H(\phi)\vec{n} \cdot \nabla T_{nn}^1 &= 0 \\ -H(-\phi)\vec{n} \cdot \nabla T_{nn}^2 &= 0\,, \end{aligned} \tag{8.17}$$

and

$$\begin{aligned} H(\phi)(\vec{n}\cdot\nabla T_n^1 - T_{nn}^1) &= 0 \\ -H(-\phi)(\vec{n}\cdot\nabla T_n^2 - T_{nn}^2) &= 0\,, \end{aligned} \tag{8.18}$$

respectively. These *static Hamilton-Jacobi* equations can be solved using a fast sweeping method [111, 60, 59]. Here, we briefly describe the numerical procedures. Note that all equations are in the form of

$$uT_\xi + vT_\eta = R(\xi,\eta)\,. \tag{8.19}$$

Using upwind differencing, we get the following iterative scheme for $T_{i,j}$ with $1 \le i \le n_\xi$ and $1 \le j \le n_\eta$,

$$T_{i,j}^{k+1} = \frac{\left[\begin{array}{l} \Delta\eta\{(u-|u|)T_{i+1,j}^{k'} - (u+|u|)T_{i-1,j}^{k'}\}+ \\ \Delta\xi\{(v-|v|)T_{i,j+1}^{k'} - (v+|v|)T_{i,j-1}^{k'}\} - 2\Delta\xi\Delta\eta R_{i,j} \end{array}\right]}{-2(\Delta\eta|u| + \Delta\xi|v|)}\,, \tag{8.20}$$

for $k = 1, 2, \cdots$ and $k' = k$ or $k+1$, depending on the direction of the sweeping.

Theoretically, the normal velocity v_n is defined only on the interface through equation (8.3). To update the level set equation, on the other hand, one needs to define a normal velocity everywhere in the computational domain. One way to find such a velocity is to extend the normal velocity perpendicular to the interface. At each point (x_i, y_j), we

1. determine the closest point on the interface to that point, given by

$$(x^*, y^*) = (x_i, y_j) - \phi(x_i, y_j)\vec{n}\,, \tag{8.21}$$

where $\vec{n}$ is the normal vector at the point (x_i, y_j).

2. use cubic interpolation to determine $v_n(x^*, y^*)$ and assign

$$v_n(x_i, y_j) = v_n(x^*, y^*)\,. \tag{8.22}$$

8.3.4 Updating the Level Set Function

Applying the coordinate transformation (8.10) to the level set equation, we have

$$\phi_t + \frac{v_n}{\Delta}\sqrt{a\phi_\xi^2 + 2b\phi_\xi\phi_\eta + c\phi_\eta^2} = 0\,, \tag{8.23}$$

where $a = B^2 + N^2$, $b = -(AB + MN)$, $c = A^2 + M^2$ and $\Delta = AN - BM$. This Hamilton-Jacobi equation is solved using a fifth order WENO-LF scheme in space [55] and a third order TVD-RK scheme in time [82].

8.3.5 Reinitialization

It is important to regularize the level set function in the evolution. For example, an over-flattened region could introduce large errors in the interpolation from determining the zero level set. To reduce these errors, we numerically preserve the signed-distance property of the level set function near the zero level set by solving the following reinitialization equation

$$\phi_\tau + S(\phi_0)(|\nabla\phi| - 1) = 0\,, \tag{8.24}$$

where $S(.)$ approximate the signum function. In the transformed coordinates, this equation reads

$$\phi_\tau + S(\phi_0)\left(\frac{1}{\Delta}\sqrt{a\phi_\xi^2 + 2b\phi_\xi\phi_\eta + c\phi_\eta^2} - 1\right) = 0\,. \tag{8.25}$$

Again, this Hamilton-Jacobi equation is solved using a fifth order WENO-LF scheme in space and a third order TVD-RK scheme in time. Few iterations in the τ-direction are sufficient to get a regular level set function near the zero level set.

8.3.6 A Predictor-Corrector Approach

Traditionally, a moving grid technique updates both an objective function and the mesh in the following way:

1. Given the grids (x^n, y^n) and the function $\phi^n(x^n, y^n)$ defined at time t^n, update ϕ^n to get $\phi^{n+1/2}(x^n, y^n)$. This step is denoted by the operator E_{x^n}.

2. With the function $\phi^{n+1/2}(x^n, y^n)$ fixed, adapt the grid to obtain (x^{n+1}, y^{n+1}). We denote this step by $A(\phi^{n+1/2})$.

3. Interpolate $\phi^{n+1}(x^{n+1}, y^{n+1})$ at (x^{n+1}, y^{n+1}). The interpolation procedure is denoted by $I_{x^{n+1}}$.

A problem with the above procedure is the excessive numerical dissipation in the operator $I_{x^{n+1}}$. To reduce this error, [107] introduced different types of interpolation scheme.

In this work, however, we design the following predictor-corrector approach to reduce the numerical dissipation in the interpolation.

1. The procedures E_{x^n} and $A(\phi^{n+1/2})$ are considered as a predictor step. The function $\phi^{n+1/2}(x^n, y^n)$ is considered as a prediction of the function values at the next time step t^{n+1}, and it is used **only** to give the new location of the grids.

2. A new operator $E_{x^n \to x^{n+1}}$ is introduced. This operator couples the level set equation with the grid motion driven by the velocities

$$F = \frac{dx}{dt} \text{ and } G = \frac{dy}{dt}\,, \tag{8.26}$$

in terms of the following transformation

$$\begin{aligned} dt &= d\tau \\ dx &= F d\tau + A d\xi + B d\eta \\ dy &= G d\tau + M d\xi + N d\eta\,, \end{aligned} \tag{8.27}$$

where the local grid information is given by

$$A = \frac{\partial x}{\partial \xi}\,,\ B = \frac{\partial x}{\partial \eta}\,,\ M = \frac{\partial y}{\partial \xi}, N = \frac{\partial y}{\partial \eta} \text{ and } \Delta = AN - BM\,. \tag{8.28}$$

Now the level set equation in this moving grid coordinates is given by

$$\phi_\tau - \frac{FN - GB}{\Delta}\frac{\partial \phi}{\partial \xi} - \frac{GA - FM}{\Delta}\frac{\partial \phi}{\partial \eta} + \frac{v_n}{\Delta}\sqrt{a\phi_\xi^2 + 2b\phi_\xi\phi_\eta + c\phi_\eta^2} = 0\,, \tag{8.29}$$

8.4 Numerical Examples

8.4.1 Example 1

The initial condition is a square of length 0.2. The domain of interested is given by $\Omega = [-1, 1]^2$. The temperature is given by $T = -0.0005\kappa$ on the interface and $T = -1$ on the outer boundary $\{x = \pm 1\} \cup \{y = \pm 1\}$. We plot the interfaces at time from $t = 0$ to 0.2 in increment of 0.025 in Figures 8.1-8.3. Figure 8.1 shows the numerical solution computed using fixed Eulerian grids. It clearly shows the grid anisotropy in the computation. This effect can, of course, be eliminated by increasing the total number of grids. However, comparing the solutions in the first row of Figure 8.2 with Figure 8.3, we can eliminate this anisotropy effect by simply using the proposed adaptive grid strategy.

8.4.2 Example 2

This example simulates crystal anisotropy in which the crystal grows in some preferred directions. The Gibbs-Thomson relation on the interface is given by [2]

$$T = -\sigma(\vec{n})\kappa - \mu(\vec{n})v_n \,, \tag{8.30}$$

with

$$\begin{aligned} \sigma(\theta) &= \sigma\left\{1 + A_s\left[\frac{8}{3}\sin^4\left(\frac{1}{2}m_s(\theta - \phi_s)\right) - 1\right]\right\} \\ \mu(\theta) &= \mu\left\{1 + A_k\left[\frac{8}{3}\sin^4\left(\frac{1}{2}m_l(\theta - \phi_k)\right) - 1\right]\right\} . \end{aligned} \tag{8.31}$$

Here m_s and m_k determine the mode of symmetry of the crystal, ϕ_s and ϕ_k determine the angles of the symmetry axis made with the x-axis, A_s and A_k are

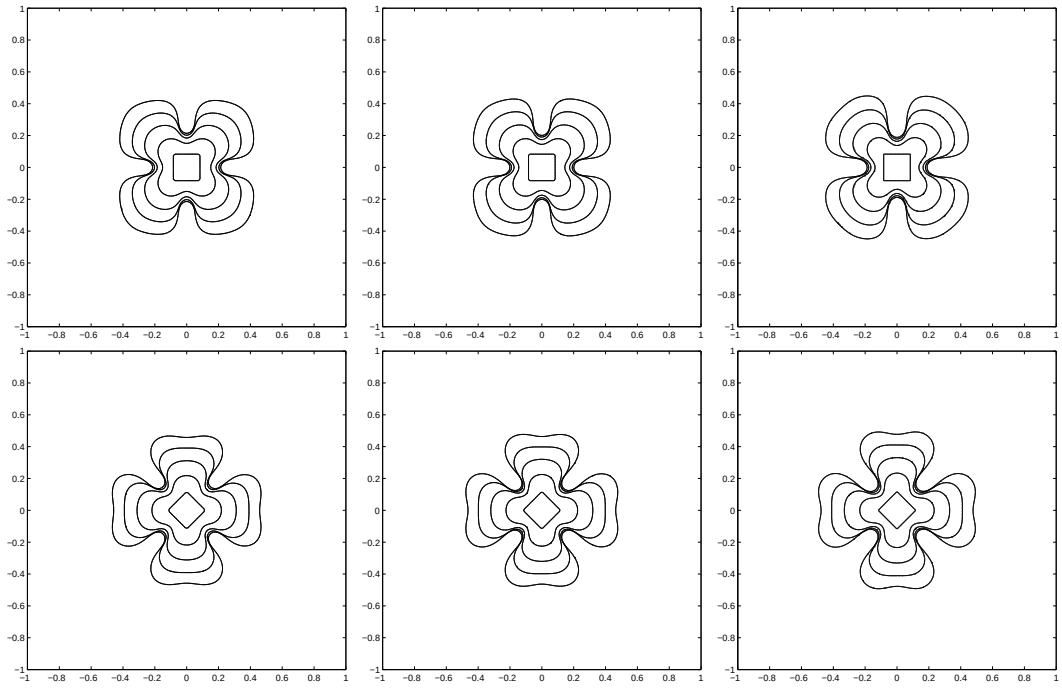


Figure 8.1: (Example 1) Evolution of the interface using 150-by-150, 200-by-200 and 400-by-400 uniform rectangular grids with two different orientations of the initial profile.

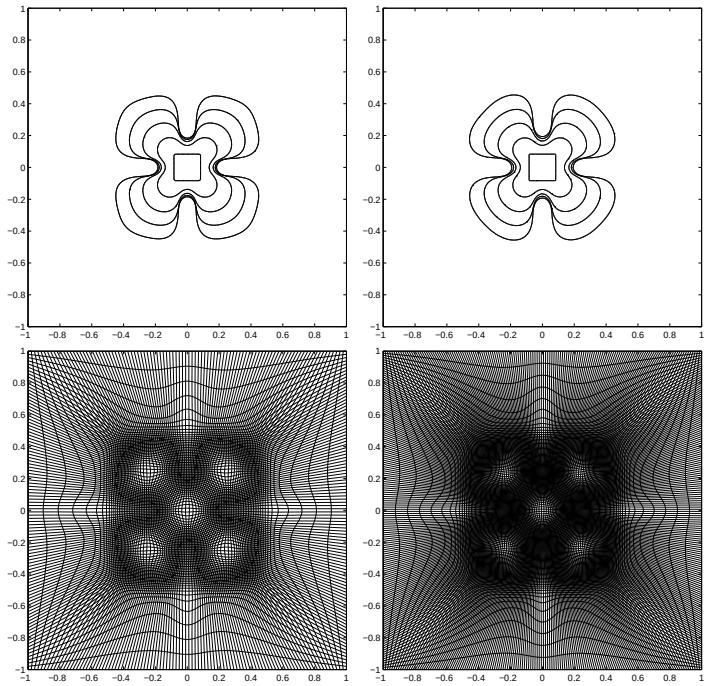


Figure 8.2: (Example 1) Evolution of the interface using 100-by-100 and 150-by-150 adaptive grids. The second row shows the grid points at the last time step.

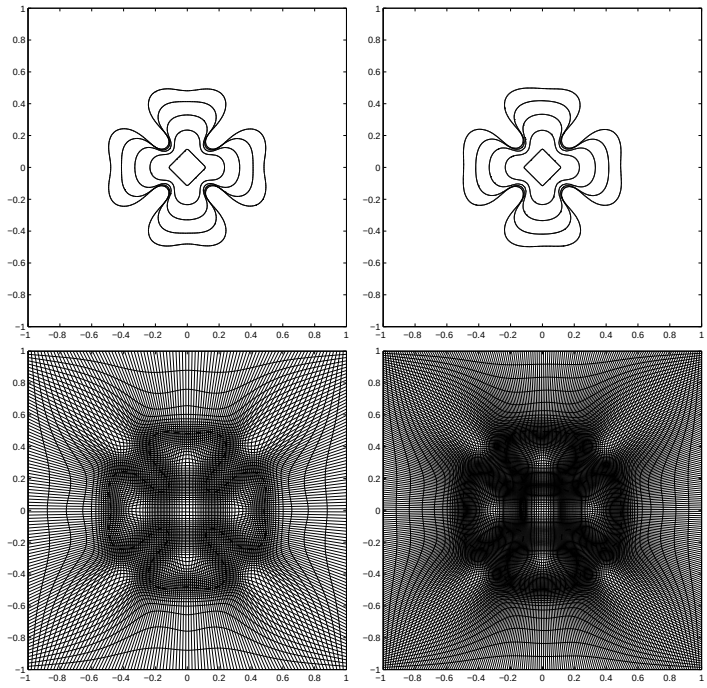


Figure 8.3: (Example 1) Evolution of the interface using 100-by-100 and 150-by-150 adaptive grids with another orientation for the initial seed. The second row shows the grid points at the last time step.

the strength of the anisotropy and θ is the angle made between the normal vector $\vec{n}$ and the x-axis. We have tried three different initial conditions.

(a) Initial seed is a square with sides 1/6. $A_s = A_k = 0.5$, $\sigma = \mu = 0.0005$ and $m_s = m_k = 4$.

(b) Initial seed is a square with sides 1/6 but rotated by $\pi/4$. $A_s = A_k = 0.5$, $\sigma = \mu = 0.0005$ and $m_s = m_k = 4$.

(c) Initial seed is four-folded given by

$$r = r_0 + p_0 \cos 4\theta \,, \tag{8.32}$$

with $r_0 = 0.05$ and $p_0 = 0.01$. $A_s = A_k = 0.5$, $\sigma = \mu = 0.00025$ and $m_s = m_k = 6$.

All solutions are plotted at increments of 0.025 from $t = 0$ to 0.2. The outer boundary condition $T = -1$ is imposed on $r = 0.95$.

Figure 8.4 shows the solutions using adaptive moving grids with setting (a). The mode of symmetry in the Gibbs-Thomson relation is 4. Comparing the solutions with Figure 8.5, the resolution of the solutions using 100-by-100 adaptive grids is similar to that of 200-by-200 uniform adaptive grids.

Solutions with the initial setting (b) are plotted in Figures 8.6 and 8.7. Similar to case (a), the uniform grids solutions are similar to that from our proposed adapted grids with approximately only one-forth the total number of grid points.

Figure 8.8 and 8.9 show the solutions with the initial setting (c). We now halve both the surface tension coefficient and the kinetic coefficient. We see that the growth is now becoming more sensitive to the initial profile and the resulting shape of the crystal is more complex. The mode of symmetry of the crystal is

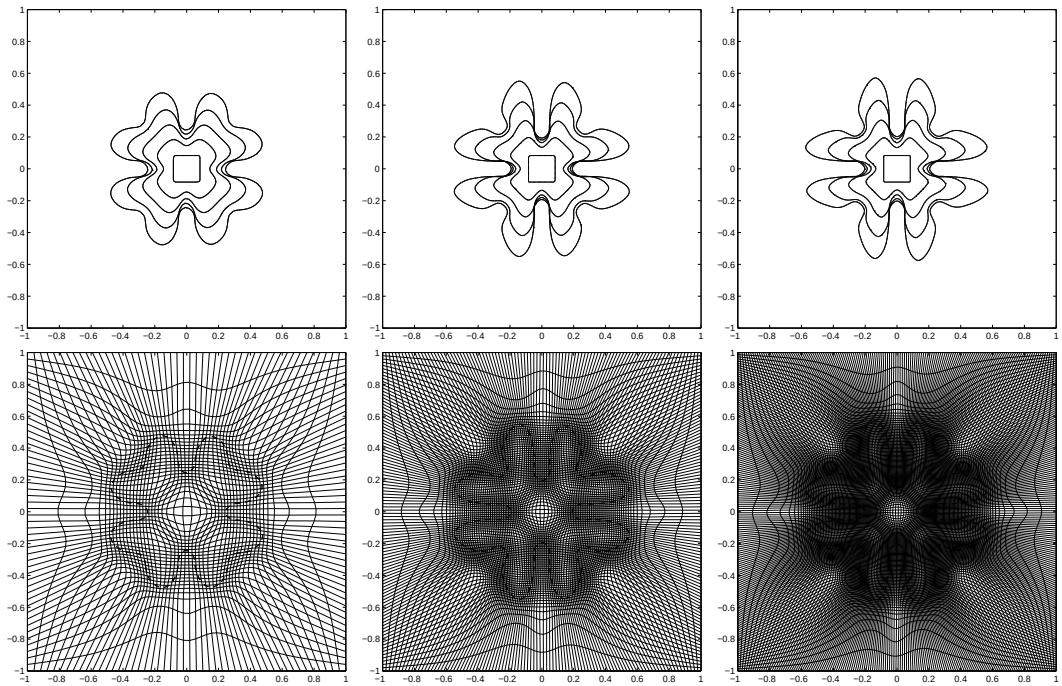


Figure 8.4: (Example 2a) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step.

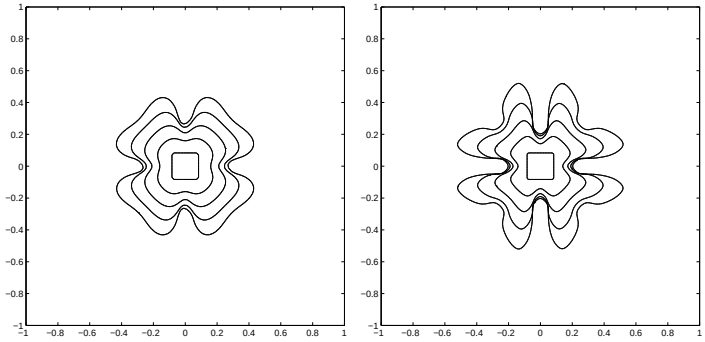


Figure 8.5: (Example 2a) 100-by-100 and 200-by-200 uniform rectangular grids.

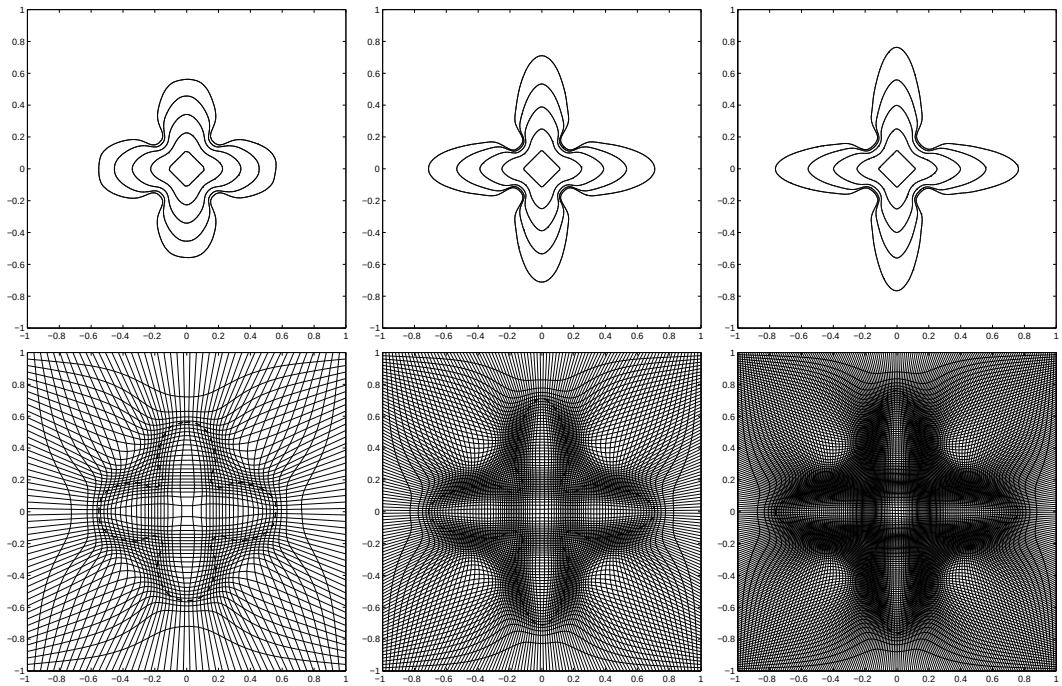


Figure 8.6: (Example 2b) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step.

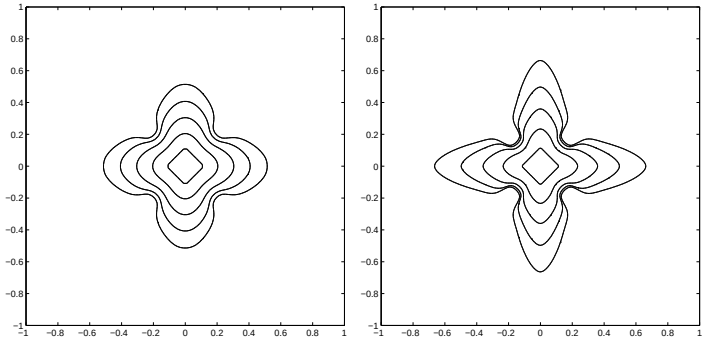


Figure 8.7: (Example 2b) 100-by-100 and 200-by-200 uniform rectangular grids.

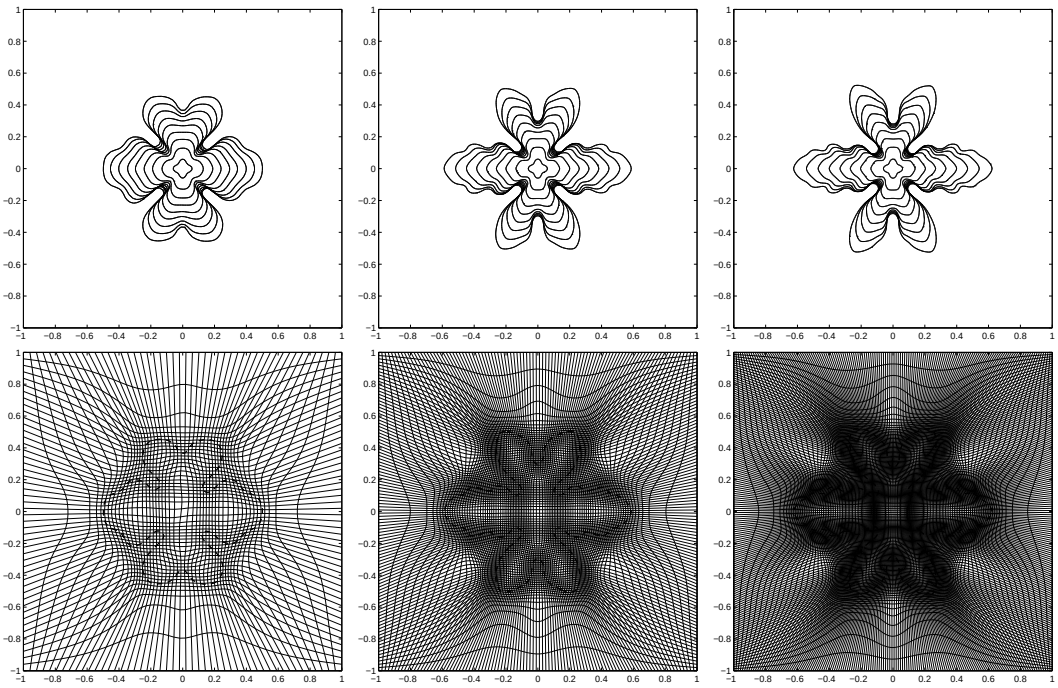


Figure 8.8: (Example 2c) 50-by-50, 100-by-100 and 150-by-150 adaptive grids. Second row shows the adaptive grids at the last time step.

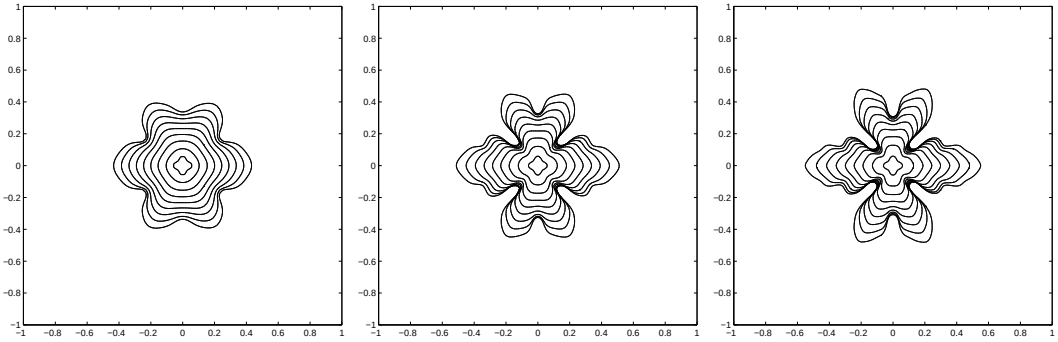


Figure 8.9: (Example 2c) 100-by-100, 150-by-150 and 200-by-200 uniform rectangular grids.

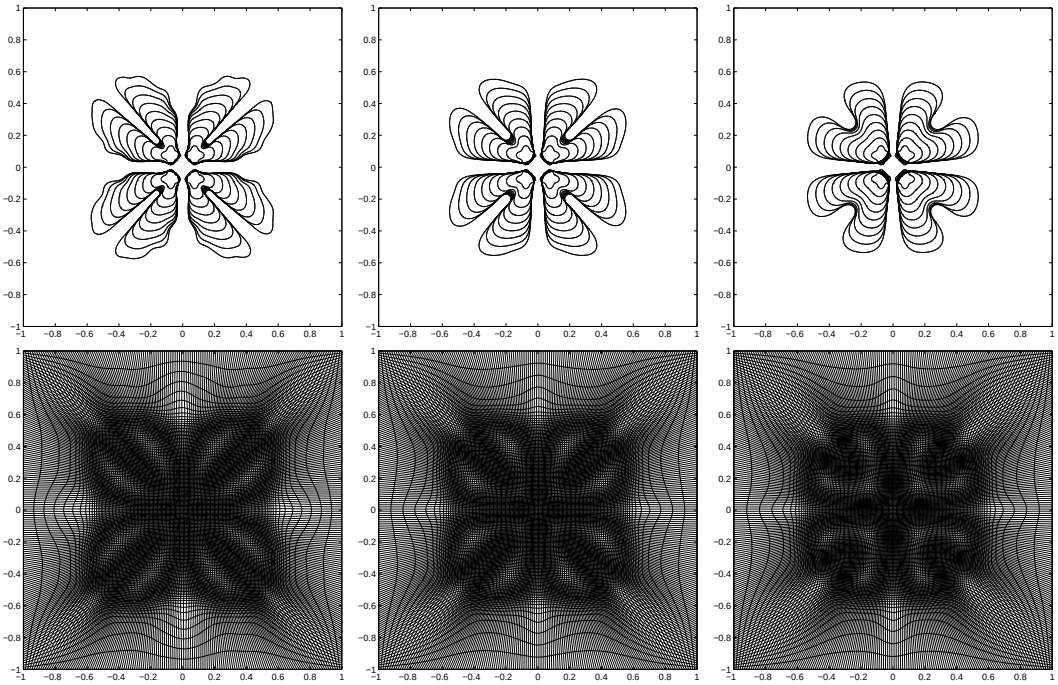


Figure 8.10: (Example 3) Four four-fold initial seeds with (left to right) $\sigma = 0.0001, 0.00025$ and 0.0005 in the Gibbs-Thomson condition. The corresponding grid points at the last time step are shown on the bottom.

increased to 6. Again, we obtain a similar evolution of the crystal using only one-forth of grid points in our proposed adaptive method than that from the uniform grids.

8.4.3 Example 3

This last example models the growth of four four-folded seeds with each seed defined by (8.32). These seeds are initially centered at $(\pm 0.075, \pm 0.075)$ and $(\pm 0.075, \mp 0.075)$. The Gibbs-Thomson relation on the interfaces are given by $T = -\sigma\kappa$. Temperature on the outer boundary $r = 0.95$ is defined as $T = -1.0$. Solutions at $t = 0$ to 0.1 using 150-by-150 grid points are shown in Figure 8.10.

References

[1] D. Adalsteinsson and J.A. Sethian. A fast level set method for propagating interfaces. *J. Comput. Phys.*, 118:269–277, 1995.

[2] R. Almgren. Variational algorithms and pattern formation in dendritic solidification. *J. Comput. Phys.*, 106:337–354, 1993.

[3] T. Aslam. A partial differential equation approach to multidimensional extrapolation. *J. Comput. Phys.*, 193:349–355, 2004.

[4] G. Bal, J. B. Keller, G. Papanicolaou, and L. Ryzhik. Transport theory for acoustic waves with reflection and transmission at interfaces. *Wave Motion*, 30:303–327, 1999.

[5] J. D. Benamou. Big ray tracing: multivalued travel time field computation using viscosity solutions of the eikonal equations. *J. Comput. Phys.*, 128:463–474, 1996.

[6] J.-D. Benamou. Direct solution of multi-valued phase-space solutions for Hamilton-Jacobi equations. *Comm. Pure Appl. Math.*, 52:1443–1475, 1999.

[7] J. D. Benamou. An introduction to Eulerian geometrical optics (1992 - 2002). *J. Sci. Comp.*, 19:63–93, 2003.

[8] J. Berryman. Stable iterative reconstruction algorithm for nonlinear traveltime tomography. *Inverse Problems*, 6:21–42, 1990.

[9] J. Berryman. Analysis of approximate inverses in tomography I. resolution analysis of common inverses. *Optimization and Engineering*, 1:87–115, 2000.

[10] J. Berryman. Analysis of approximate inverses in tomography II. iterative inverses. *Optimization and Engineering*, 1:437–473, 2000.

[11] T. N. Bishop, K. P. Bube, R. T. Cutler, R. T. Langan, P. L. Love, J. R. Resnick, R. T. Shuey, D. A. Spindler, and H. W. Wyld. Tomographic determination of velocity and depth in laterally varying media. *Geophysics*, 50:903–923, 1985.

[12] M. Boue and P. Dupuis. Markov chain approximations for deterministic control problems with affine dynamics and quadratic costs in the control. *SIAM J. Numer. Anal.*, 36:667–695, 1999.

[13] X. Bresson, S. Esedoḡlu, P. Vandergheynst, J.P. Thiran, and S. Osher. Global minimizers of the active contour/snake model. *UCLA CAM Report (05-04)*, 2005.

[14] K. P. Bube and R. T. Langan. Hybrid l^1-l^2 minimization with applications to tomography. *Geophysics*, 62:1183–1195, 1997.

[15] R. Burridge. Asymptotic evaluation of integrals related to time-dependent fields near caustics. *SIAM J. Appl. Math.*, 55:390–409, 1995.

[16] R.H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, 1995.

[17] V. Cerveny, I. A. Molotkov, and I. Psencik. *Ray method in seismology.* Univerzita Karlova press, 1977.

[18] R. Chan, C.-W. Ho, and M. Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on Image Processing*, 14:1479–1485, 2005.

[19] R. Chan, C. Hu, and M. Nikolova. An iterative procedure for removing random-valued impuse noise. *IEEE Signal Processing Letters*, 11:921–924, 2004.

[20] T.F. Chan, S. Esedoḡlu, and M. Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *UCLA CAM Report (04-54)*, 2004.

[21] T.F. Chan and S.H. Kang. Error analysis for image inpainting. *UCLA CAM Report (04-72)*, 2004.

[22] T.F. Chan and J. Shen. Mathematical models of local non-texture inpainting. *SIAM J. Appl. Math.*, 62:1019–1043, 2001.

[23] S. Chen, B. Merriman, S. Osher, and P. Smereka. A simple level set method for solving stefan problems. *J. Comput. Phys.*, 135, 1997.

[24] L.-T. Cheng, H. Liu, and S. J. Osher. High frequency wave propagation in Schrodinger equations using the level set method. Preprint (www.levelset.com), 2003.

[25] L.-T. Cheng, S. J. Osher, and J. Qian. Level set based eulerian methods for multivalued traveltimes in both isotropic and anisotropic media. In *73st Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1801–1804. Soc. Expl. Geophys., Tulsa, OK, 2003.

[26] J. F. Claerbout. *Fundamentals of geophysical data processing.* McGraw-Hill, 1976.

[27] R. A. Clarke, B. Alazard, L. Pelle, D. Sinuquet, P. Lailly, F. Delprat-Jannaud, and L. Jannaud. 3D traveltime reflection tomography with multi-valued arrivals. In *71st Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1601–1604. Soc. Expl. Geophys., Tulsa, OK, 2001.

[28] M. G. Crandall, L. C. Evans, and P. L. Lions. Some properties of viscosity solutions of hamilton-jacobi equations. *Trans. Amer. Math. Soc.*, 282:487–502, 1984.

[29] M. G. Crandall and P. L. Lions. Viscosity solutions of Hamilton-Jacobi equations. *Trans. Amer. Math. Soc.*, 277:1–42, 1983.

[30] M. G. Crandall and P. L. Lions. Two approximations of solutions of Hamilton-Jacobi equations. *Math. Comp.*, 43:1–19, 1984.

[31] F. A. Dahlen, S.-H. Hung, and G. Nolet. Frechet kernels for finite-frequency traveltimes– I. theory. *Geophys. J. Int.*, 141:157–174, 2000.

[32] F. Delprat-Jannaud and P. Lailly. Reflection tomography: how to handle multiple arrivals? *J. Geophys. Res.*, 100:703–715, 1995.

[33] K. A. Dines and R. J. Lytle. Computerized geophysical tomography. *Proc. IEEE*, 67:1065–1073, 1979.

[34] O. Dorn, H. Bertete-Aguiree, J. G. Berryman, and G. C. Papanicolaou. A nonlinear inversion method for 3d electromagnetic imagaing using adjoint fields. *Inverse Problems*, 15:1523–1558, 1999.

[35] B. Engquist, E. Fatemi, and S. Osher. Numerical resolution of the high frequency asymptotic expansion of the scalar wave equation. *J. Comput. Phys.*, 120:145–155, 1995.

[36] B. Engquist and O. Runborg. Multi-phase computations in geometrical optics. *J. Comput. Appl. Math.*, 74:175–192, 1996.

[37] B. Engquist and O. Runborg. Computational high frequency wave propagation. *Acta Numerica*, 12:181–266, 2003.

[38] B. Engquist, O. Runborg, and A-K Tornberg. High frequency wave propagation by the segment projection method. *J. Comp. Phys.*, 178:373–390, 2002.

[39] S. Esedoḡlu and J. Shen. Digital inpainting based on the Mumford-Shah-Euler image model. *European Journal of Applied Mathematics*, 13:353–370, 2002.

[40] L. C. Evans. Towards a quantum analogue of weak KAM theory. Preprint, 2002.

[41] M. Falcone and R. Ferretti. Semi-lagrangian schemes for hamilton-jacobi equations, discrete representation formulae and godunov methods. *J. Comput. Phys.*, 175:559–575, 2002.

[42] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid mathod). *J. Comput. Phys.*, 152:457–492, 1999.

[43] S. Fomel and J. Sethian. Fast phase space computation of multiple traveltimes. *Proc. Nat. Aca. Sci.*, 99:7329–7334, 2002.

[44] F. Friedlander. *Sound pulses*. Cambridge University Press, 1958.

[45] I.M. Gelfand and S.V. Fomin. *Calculus of Variation*. Prentice-Hall, 1963.

[46] F. Gibou and R. Fedkiw. A fourth order accurate discretization for the laplace and heat equations on arbitrary domains, with applications to the stefan problem. *J. Comput. Phys.*, 202:577–601, 2005.

[47] F. Gibou, R. Fedkiw, R. Caflisch, and S. Osher. A level set approach for the numerical simulation of dendritic growth. *J. Sci. Comput.*, 19:183–199, 2003.

[48] L. Gosse. Using K-branch entropy solutions for multivalued geometric optics computations. *J. Comput. Phys.*, 180:155–182, 2002.

[49] L. Gosse and P. A. Markowich. Multiphase semiclassical approximations of an electron in a one-dimensional crystalline lattice. *J. Comput. Phys.*, 197:387–417, 2004.

[50] S. Gray and W. May. Kirchhoff migration using eikonal equation traveltimes. *Geophysics*, 59:810–817, 1994.

[51] Jerry M. Harris, Richard C. Nolen-Hoeksema, Robert T. Langan, Mark Van Schaack, Spyros K. Lazaratos, and James W. Rector III. High-resolution crosswell imaging of a west texas carbonate reservoir: Part I-project summary and interpretation. *Geophysics*, 60:667–681, 1995.

[52] T. Y. Hou, Z. Li, S. J. Osher, and H.-K. Zhao. A hybrid method for moving interface problems with applications to the hele-shaw flows. *J. Comput. Phys.*, 134:236–252, 1997.

[53] W. Huang and R.D. Russell. Moving mesh stragegy based on a gradient flow equation for two-dimensonal problems. *SIAM J. Sci. Comput.*, 20(3):998–1015, 1999.

[54] David W. Hyndman and Jerry M. Harris. Traveltime inversion for the geometry of aquifer lithologies. *Geophysics*, 61:1728–1737, 1996.

[55] G. S. Jiang and D. Peng. Weighted ENO schemes for Hamilton-Jacobi equations. *SIAM J. Sci. Comput.*, 21:2126–2143, 2000.

[56] S. Jin and X. Li. Multi-phase computations of the semi-classical limit of the Schrodinger equation and related problems. *Physica D*, 182:46–85, 2003.

[57] S. Jin and S. Osher. A level set method for the computation of multivalued solutions to quasi-linear hyperbolic PDEs and Hamilton-Jacobi equations. *Commun. Math. Sci.*, 1:575–591, 2003.

[58] B.R. Julian and D. Gubbins. Three-dimensional seismic ray tracing. *J. Geophys.*, 43:95–113, 1977.

[59] C. Y. Kao, S. J. Osher, and Y.-H. Tsai. Fast sweeping method for static Hamilton-Jacobi equations. *SIAM J. Num. Anal.*, 42:2612–2632, 2005.

[60] C.Y. Kao, S.J. Osher, and J. Qian. Lax-Friedrichs sweeping schemes for static Hamilton-Jacobi equations. *J. Comp. Phys.*, 196:367–391, 2004.

[61] J. B. Keller and R. M. Lewis. Asymptotic methods for partial differential equations: the reduced wave equation and Maxwell's equations. *Surveys in Applied Mathematics*, 1:1–82, 1995.

[62] Y.T. Kim, N. Goldenfeld, and J. Dantzig. Computation of dentritic microstructures using a level set method. *Physical Review E*, 62:2471–2474, 2000.

[63] S. N. Kruzkov. Generalized solutions of nonlinear first order equations with several independent variables. II. *Math. USSR-Sb.*, 1:93–116, 1967.

[64] P. Kumpp and S. Steinberg. *Fundamentals of Grid Generation*. CRC Press, 1993.

[65] S. Leung and S. Osher. Fast global minimization of the active contour model with tv-inpainting and two-phase denoising. *Proceeding of the 3rd IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision*, pages 149–160, 2005.

[66] S. Leung and J. Qian. A transmission tomography problem based on multiple arrivals from paraxial liouville equations. *Expanded Abstract for the SEG 75th Annual Meeting*, 2005.

[67] S. Leung and J. Qian. An adjoint state method for 3d transmission traveltime tomography using first arrival. *Commun. Math. Sci.*, 4:249–266, 2006.

[68] S. Leung and J. Qian. Transmission traveltime tomography based on paraxial liouville equations and level set formulations. 2006.

[69] S. Leung, J. Qian, and S. Osher. A level set method for three-dimensional paraxial geometrical optics with multiple sources. *Commun. Math. Sci.*, 2(4):643–672, 2004.

[70] P. L. Lions. *Generalized solutions of Hamilton-Jacobi equations*. Pitman Advanced Publishing Program, 1982.

[71] X. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115:200–212, 1994.

[72] D. Ludwig. Uniform asymptotic expansions at a caustic. *Comm. Pure Appl. Math.*, XIX:215–250, 1966.

[73] V. P. Maslov and M. V. Fedoriuk. *Semi-classical approximation in quantum mechanics*. D. Reidel Publishing Company, 1981.

[74] S. M. Minkoff. A computationally feasible approximate resolution matrix for seismic inverse problems. *Geophys. J. Internat.*, 126:345–359, 1996.

[75] R. Montelli, G. Nolet, F. A. Dahlen, G. Masters, E. R. Engdahl, and S.-H. Hung. Finite-frequency tomography reveals a variety of plumes in the mantle. *Science*, 303:338–343, 2004.

[76] R. Montelli, G. Nolet, G. Masters, F. A. Dahlen, and S.-H. Hung. Global P and PP traveltime tomography: rays versus waves. *Geophys. J. Int.*, 158:637–654, 2004.

[77] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math.*, 42:577–685, 1998.

[78] M. J. P. Musgrave. *Crystal acoustics*. Holden-Day, 1970.

[79] S. Osher, L.-T. Cheng, M. Kang, H. Shim, and Y-H Tsai. Geometric optics in a phase space based level set and Eulerian framework. *J. Comput. Phys.*, 179:622–648, 2002.

[80] S. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, New York, 2003.

[81] S. J. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79:12–49, 1988.

[82] S. J. Osher and C. W. Shu. High-order Essentially NonOscillatory schemes for Hamilton-Jacobi equations. *SIAM J. Num. Anal.*, 28:907–922, 1991.

[83] D. Peng, B. Merriman, S. Osher, H. K. Zhao, and M. Kang. A pde-based fast local level set method. *J. Comput. Phys.*, 155:410–438, 1999.

[84] F. Poupaud and C. Ringhofer. Semi-classical limits in a crystal with external potentials and effective mass theorems. *Commun. Partial Diff. Eq.*, 21:1897–1918, 1996.

[85] J. Qian, L.-T. Cheng, and S.J. Osher. A level set based Eulerian approach for anisotropic wave propagations. *Wave Motion*, 37:365–379, 2003.

[86] J. Qian and S. Leung. A level set method for paraxial multivalued traveltimes. *J. Comput. Phys.*, 197:711–736, 2004.

[87] J. Qian and S. Leung. A local level set method for paraxial multivalued geometric optics. *SIAM J. Sci. Comput.*, 28:206–223, 2006.

[88] J. Qian and W. W. Symes. Adaptive finite difference method for traveltime and amplitude. *Geophysics*, 67:167–176, 2002.

[89] J. Qian and W. W. Symes. Finite-difference quasi-P traveltimes for anisotropic media. *Geophysics*, 67:147–155, 2002.

[90] J. Qian, Y-T Zhang, and H-K. Zhao. Fast sweeping methods for eikonal equations on triangulated domains. Submitted to SIAM J. Numer. Analy., 2004.

[91] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM J. Num. Anal.*, 29:867–884, 1992.

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

[93] S. J. Ruuth, B. Merriman, and S. J. Osher. A fixed grid method for capturing the motion of self-intersecting interfaces and related PDEs. *J. Comput. Phys.*, 151:836–861, 1999.

[94] A. Sei and W. W. Symes. Gradient calculation of the traveltime cost function without ray tracing. In *65th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1351–1354. Soc. Expl. Geophys., Tulsa, OK, 1994.

[95] A. Sei and W. W. Symes. Convergent finite-difference traveltime gradient for tomography. In *66th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1258–1261. Soc. Expl. Geophys., Tulsa, OK, 1995.

[96] J. Sethian and J. Strain. Crystal growth and dendritic solidification. *J. Comput. Phys.*, 98:231–253, 1992.

[97] J. A. Sethian. *Level set methods*. Cambridge Univ. Press, second edition, 1999.

[98] C. W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In B. Cockburn, C. Johnson, C.W. Shu, and E. Tadmor, editors, *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, volume 1697, pages 325–432. Springer, 1998. Lecture Notes in Mathematics.

[99] C. W. Shu and S. J. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes. *J. Comput. Phys.*, 77:439–471, 1988.

[100] J. Steinhoff, M. Fan, and L. Wang. A new Eulerian method for the computation of propagating short acoustic and electromagnetic pulses. *J. Comput. Phys.*, 157:683–706, 2000.

[101] J. Strain. Semi-lagrangian methods for level set equations. *J. Comput. Phys.*, 151:498–533, 1999.

[102] M. Sussman, P. Smereka, and S. J. Osher. A level set approach for computing solutions to incompressible two-phase flows. *J. Comput. Phys.*, 114:146–159, 1994.

[103] W. W. Symes. Mathematics of reflection seismology. In *Annual Report, The Rice Inversion Project, (http://www.trip.caam.rice.edu/)*. Rice University, 1995.

[104] W. W. Symes. A slowness matching finite difference method for traveltimes beyond transmission caustics. In *68th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1945–1948. Soc. Expl. Geophys., 1998.

[105] W. W. Symes and J. Qian. A slowness matching Eulerian method for multivalued solutions of eikonal equations. *J. Sci. Comp.*, 19:501–526, 2003.

[106] W. W. Symes, R. Versteeg, A. Sei, and Q. H. Tran. Kirchhoff simulation, migration and inversion using finite difference traveltimes and amplitudes. In *Annual Report, The Rice Inversion Project, (http://www.trip.caam.rice.edu/)*. Rice University, 1994.

[107] H.K. Tang, T. Tang, and Zhang P. An adaptive mesh redistribution method for nonlinear hamilton-jacobi equations in two- and three-dimensions. *J. Comput. Phys.*, 188, 2003.

[108] J.F. Thompson, B.K. Soni, and N.P. Weatherill. *Handbook of Grid Generation*. CRC Press, 1999.

[109] L. Thomsen. Weak elastic anisotropy. *Geophysics*, 51:1954–1966, 1986.

[110] A. N. Tikhonov and V. Y. Arsenin. *On the solution of ill-posed problems*. John Wiley and Sons, New York, 1977.

[111] R. Tsai, L.-T. Cheng, S. J. Osher, and H. K. Zhao. Fast sweeping method for a class of Hamilton-Jacobi equations. *SIAM J. Numer. Anal.*, 41:673–694, 2003.

[112] G. Uhlmann. Travel-time tomography. *J. Korean Math. Soc.*, 38:711–722, 2001.

[113] G. Uhlmann and S. Hansen. Recovering acoustic and elastic parameters from travel-times. In *International Mechanical Engineering Congress and Exhibition, Proceedings*, page 32149. ASME, 2002.

[114] J. van Trier and W. W. Symes. Upwind finite-difference calculation of traveltimes. *Geophysics*, 56:812–821, 1991.

[115] J. Vidale. Finite-difference calculation of travel times. *Bull., Seis. Soc. Am.*, 78:2062–2076, 1988.

[116] J. E. Vidale and H. Houston. Rapid calculation of seismic amplitudes. *Geophysics*, 55:1504–1507, 1990.

[117] V. Vinje, E. Iversen, K. Åstebøl, and H. Gjøystdal. Estimation of multivalued arrivals in 3d models using wavefront construction - part 1. *Geophys. Prosp.*, 44:819–842, 1996.

[118] V. Vinje, E. Iversen, and H. Gjoystdal. Traveltime and amplitude estimation using wavefront construction. *Geophysics*, 58:1157–1166, 1993.

[119] John K. Washbourne, James W. Rector, and Kenneth P. Bube. Crosswell traveltime tomography in three dimensions. *Geophysics*, 67:853–871, 2002.

[120] B. S. White. The stochastic caustic. *SIAM J. Appl. Math.*, 44:127–149, 1984.

[121] L. Zhang. *Imaging by the wavefront propagation method*. PhD thesis, Stanford University, Stanford, CA94305, 1993.

[122] H. K. Zhao. Fast sweeping method for eikonal equations. *Math. Comp.*, 74:603–627, 2005.

[123] H.-K. Zhao, T. Chan, B. Merriman, and S. J. Osher. A variational level set approach for multiphase motion. *J. Comput. Phys.*, 127:179–195, 1996.

[124] H.K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and nonparametric shape reconstruction from unorganized points using variational level set method. *Computer Vision and Image Understanding*, 80:295–319, 2000.