

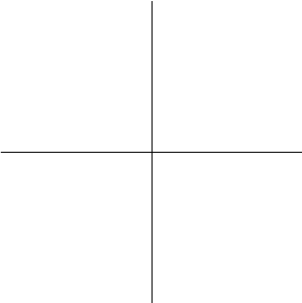


KTH Engineering Sciences

Fast Adaptive Numerical Methods for High Frequency Waves and Interface Tracking

JELENA POPOVIC

Doctoral Thesis
Stockholm, Sweden 2012



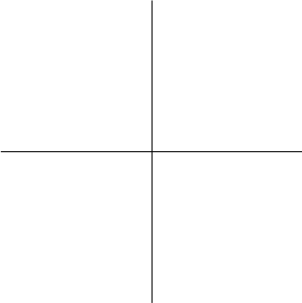
TRITA-NA 2012:13
ISSN 0348-2952
ISRN KTH/NA/-12/13-SE
ISBN 978-91-7501-577-4

KTH School of Engineering Sciences
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i numerisk analys måndagen den 10 december 2012 klockan 10.15 i Sal D2, Lindstedsvägen 5, éntreplan Kungliga Tekniska högskolan, Stockholm.

© Jelena Popovic, december 2012

Tryck: E-print, www.eprint.se



Abstract

The main focus of this thesis is on fast numerical methods, where adaptivity is an important mechanism to lowering the methods' complexity. The application of the methods are in the areas of wireless communication, antenna design, radar signature computation, noise prediction, medical ultrasonography, crystal growth, flame propagation, wave propagation, seismology, geometrical optics and image processing.

We first consider high frequency wave propagation problems with a variable speed function in one dimension, modeled by the Helmholtz equation. One significant difficulty of standard numerical methods for such problems is that the wave length is very short compared to the computational domain and many discretization points are needed to resolve the solution. The computational cost, thus grows algebraically with the frequency ω . For scattering problems with impenetrable scatterer in homogeneous media, new methods have recently been derived with a provably lower cost in terms of ω . In this thesis, we suggest and analyze a fast numerical method for the one dimensional Helmholtz equation with variable speed function (variable media) that is based on wave-splitting. The Helmholtz equation is split into two one-way wave equations which are then solved iteratively for a given tolerance. We show rigorously that the algorithm is convergent, and that the computational cost depends only weakly on the frequency for fixed accuracy.

We next consider interface tracking problems where the interface moves by a velocity field that does not depend on the interface itself. We derive fast adaptive numerical methods for such problems. Adaptivity makes methods robust in the sense that they can handle a large class of problems, including problems with expanding interface and problems where the interface has corners. They are based on a multiresolution representation of the interface, i.e. the interface is represented hierarchically by wavelet vectors corresponding to increasingly detailed meshes.

The complexity of standard numerical methods for interface tracking, where the interface is described by marker points, is $O(N/\Delta t)$, where N is the number of marker points on the interface and Δt is the time step. The methods that we develop in this thesis have $O(\Delta t^{-1} \log N)$ computational cost for the same order of accuracy in Δt . In the adaptive version, the cost is $O(\text{Tol}^{-1/p} \log N)$, where Tol is some given tolerance and p is the order of the numerical method for ordinary differential equations that is used for time advection of the interface.

Finally, we consider time-dependent Hamilton-Jacobi equations with convex Hamiltonians. We suggest a numerical method that is computationally efficient and accurate. It is based on a reformulation of the equation as a front tracking problem, which is solved with the fast interface tracking methods together with a post-processing step. The complexity of standard numerical methods for such problems is $O(\Delta t^{-(d+1)})$ in d dimensions, where Δt is the time step. The complexity of our method is reduced to $O(\Delta t^{-d} |\log \Delta t|)$ or even to $O(\Delta t^{-d})$.

Preface

This thesis consists of four papers and an introduction.

Paper I: J. Popovic and O. Runborg, "*Analysis of a Fast Method for Solving the High Frequency Helmholtz Equation in One Dimension*", BIT Numer. Math., vol. 51, pp. 721–755, 2011.

The author of the thesis contributed to the ideas, developed the numerical method, performed the numerical computations and wrote most of the manuscript.

Paper II: J. Popovic and O. Runborg, "*Adaptive Fast Interface Tracking Methods, Part I: Time Adaptivity*", Preprint, 2012.

The author of the thesis contributed to the ideas, developed the numerical method, performed the numerical computations and wrote most of the manuscript.

Paper III: J. Popovic and O. Runborg, "*Adaptive Fast Interface Tracking Methods, Part II: Spatial Adaptivity*", Preprint, 2012.

The author of the thesis contributed to the ideas, developed the numerical method, performed the numerical computations and wrote most of the manuscript.

Paper IV: J. Popovic and O. Runborg, "*Time Upscaling for Hamilton-Jacobi Equations*", Preprint, 2012.

The author of the thesis contributed to the ideas, performed all numerical computations except the computations in the last numerical example and wrote the manuscript.

Acknowledgments

First of all, I would like to express my gratitude for having the opportunity to do my doctoral thesis at the Department of Numerical Analysis at KTH. I have had a wonderful advisor, Prof. Olof Runborg, who spent a lot of time discussing scientific issues and helping me solve many mathematical problems. Thank you Olof for your patience, encouragement and support you have given me all throughout the years. I sincerely appreciate everything you have done to help me become a better researcher. I am very much obliged to my co-advisor, Prof. Jesper Ooppelstrup, for all the help, especially for advices and comments regarding this thesis. I would also like to thank Dr. Mikael Hanke, Beatrice Frock, Dr. Mohammad Motamed and Oana Marin for reading parts of my thesis.

A big 'Thank you!' to all my colleagues at the department. Special thanks to my friends Dr. Sara Zahedi, Dr. Murtazo Nazarov and Dr. Håkon Hoel. Sara, Murtazo, and Håkon, I am grateful for all your help and all the pleasant moments we shared. I would also like to thank Dr. Elias Jarlebring for taking time to discuss a very important last minute concern I had surrounding this thesis.

I also wish to thank families Nikolic and Minic for their love, help and support.

I am specially grateful to my parents for always being by my side and never letting me go down.

Many thanks to my husband Miki for his patience, understanding, and support during all the years we spent together, but especially during the last few months of my PhD studies. Thank you for believing in me and encouraging me every time I loose faith in myself.

Finally, many thanks to my sweet little Jana for making my life meaningful.

Koristim ovu priliku da se još jednom zahvalim svojim roditeljima na neizmernoj ljubavi i podršci tokom svih ovih godina. Hvala vam na svemu! Želim da se zahvalim i svom bratu i svojim rođacima i prijateljima u Srbiji sa kojima sam provela divne trenutke i koji su uvek uz mene. Posebno i veliko hvala najboljim ujacima na svetu, Draganu Kneževiću i dr Novici Kneževiću. Bez vaše pomoći, moj dolazak u Švedsku na studije ne bi ni bio moguć.

Contents

Preface	v
Acknowledgments	vii
Contents	viii
I Introductory Chapters	1
1 Introduction	3
2 High Frequency Wave Propagation Problems	7
2.1 Geometrical Optics	10
2.2 Numerical Methods	12
3 Numerical Methods with Weakly Frequency-Dependent Cost	15
3.1 Fast Method for Solving the High Frequency Helmholtz Equation in One Dimension	17
4 Interface Tracking	23
4.1 Standard Front Tracking	23
4.2 Level Set Methods	24
4.3 The Segment Projection Method	29
5 Multiresolution Interface Tracking	31
5.1 Multiresolution Description of the Interface	31
5.2 Governing Equations	34
5.3 The Basic Fast Numerical Method	35
5.4 Adaptive Multiresolution Front Tracking	37
6 Time Upscaling of Hamilton-Jacobi Equations	43
6.1 Hamilton-Jacobi Equations	43
6.2 Numerical methods	44
6.3 Time Upscaling Methods	45

CONTENTS

ix

6.4 Post-Processing in Time Upscaling of HJ Equations 47

7 Summary of Papers 51



7.1 Paper I: Analysis of a Fast Method for Solving the High Frequency
Helmholtz Equation in One Dimension 51

7.2 Adaptive Fast Interface Tracking Methods, Part I: Time Adaptivity 51

7.3 Adaptive Fast Interface Tracking Methods, Part II: Spatial Adaptivity 52

7.4 Time Upscaling for Hamilton-Jacobi Equations 52

Bibliography 53



Part I

Introductory Chapters

Chapter 1

Introduction

In this thesis, we propose fast and accurate numerical methods for the high frequency Helmholtz equation, interface propagation and time dependent Hamilton-Jacobi equations.

We begin by high frequency wave propagation problems, which are modeled by the Helmholtz equation. These problems arise in many applications. Currently the interest is driven by new applications in wireless communication (cell phones, bluetooth) and photonics (optical fibers, filters, switches). Simulation of high frequency waves is also used in more classical applications. Some examples in electromagnetism are antenna design and radar signature computation. In acoustics simulation is used for noise prediction, underwater communication and medical ultrasonography. One significant difficulty with numerical simulation of such problems is that the wavelength is very short compared to the computational domain and thus many discretization points are needed to resolve the solution. Consequently, the computational complexity of standard numerical methods grows algebraically with the frequency. Therefore, development of effective numerical methods for high frequency wave propagation problems is important.

Recently a new class of methods have been proposed, which, in principle, can solve the wave propagation problem at a cost almost *independent of the frequency* for a fixed accuracy. The new methods have been applied mostly to scattering problems with an impenetrable scatterer in constant media. These methods are significantly faster than direct methods. However, proving the low cost rigorously is rather difficult, although there are now a few precise proofs about computational cost and accuracy in terms of the frequency. For problems set in a domain with varying media, much less has been done. The first part of this thesis addresses such problems. We develop a fast method for the Helmholtz equation in a domain with a varying wave speed. The Helmholtz equation is split into two one-way wave equations which are then solved iteratively for a given tolerance. Moreover, we show rigorously that in one dimension the asymptotic computational cost of the method only grows slowly with the frequency, for fixed accuracy.

Next, we consider interface propagation problems where the interface moves by a velocity field that does not depend on the interface itself. We derive fast adaptive numerical methods for such problems. Problems with propagating interfaces occur in many applications. Some examples are wave propagation, multiphase flow, crystal growth, melting, epitaxial growth or flame propagation. Thus, development of fast and accurate numerical methods for such problems is important. The complexity of standard numerical methods for interface propagation, where the interface is represented by marker points, is $O(N/\Delta t)$, where N is the number of marker points on the interface and Δt is the time step. The computational cost of the methods that we develop in this thesis is $O(\Delta t^{-1} \log N)$ for the same order of accuracy in Δt . In the adaptive version, the cost is $O(\text{Tol}^{-1/p} \log N)$, where Tol is some given tolerance and p is the order of the numerical method for ordinary differential equations (ODEs) that is used for time advection of the interface. The methods are based on a multiresolution description of the interface. The interface is represented hierarchically by wavelet vectors corresponding to increasingly detailed meshes. Such representation makes it possible to use larger time steps for finer scales which leads to the reduced computational cost. The adaptivity makes methods robust in the sense that they can handle a large class of interfaces, including interfaces that have corners and interfaces whose length changes rapidly in time. However, they are only suitable for problems in which the velocity field does not depend on the interface itself. They are thus, for instance, not suitable for fronts propagating with a curvature-dependent speed.

Finally, we suggest a fast numerical method for time-dependent Hamilton-Jacobi equations with convex Hamiltonians. Applications include seismology, geometrical optics, mechanics and image processing. Standard numerical methods for Hamilton-Jacobi equations are explicit and use discretizations in time and space. The computational cost to obtain the solution at a fixed time T is of order $O(\Delta t^{-(d+1)})$ in d dimensions, where Δt is the time step. The computational cost of our method is $O(\Delta t^{-d})$ or $O(\Delta t^{-d} |\log \Delta t|)$ for the same order of accuracy. The method is based on the fact that the time dependent Hamilton-Jacobi equation can be solved by characteristics, which can be reformulated as a front tracking problem. It has two steps:

1. First, we reformulate the Hamilton-Jacobi equation as a front tracking problem and compute the multivalued solution using the fast interface tracking method proposed in the second part of the thesis. The cost of this part of the algorithm is $O(\Delta t^{-d})$ or $O(\Delta t^{-d} |\log \Delta t|)$ in d dimensions.
2. Second, we reconstruct the viscosity solution from the multivalued solution with an algorithm that has cost $O(\Delta x^{-d})$, where $\Delta x \sim \Delta t$ is the spatial discretization.

The thesis consists of two parts, the introduction and the included papers. The first part is organized as follows. In Chapter 2, we introduce the equations that are used to model high frequency wave propagation problems and discuss

standard numerical methods that are used to solve these problems. Chapter 3 introduces numerical methods for high frequency wave propagation problems with weakly frequency-dependent cost, including an outline of the method analyzed in paper 1. Equations and standard numerical methods for interface tracking problems are presented in Chapter 4. Numerical methods for interface tracking problems, based on multiresolution description of the interface, followed by a brief description of the fast interface tracking methods proposed in papers 2 and 3, are presented in Chapter 5. Chapter 6 describes numerical methods for Hamilton-Jacobi equations, including an outline of the fast method developed in the fourth paper. Finally, Chapter 7 concludes the first part of the thesis by a brief description of included papers.

Chapter 2

High Frequency Wave Propagation Problems

The basic equation that describes wave propagation problems mathematically is the scalar wave equation,

$$\frac{\partial^2}{\partial t^2} u(\mathbf{x}, t) - c(\mathbf{x})^2 \Delta u(\mathbf{x}, t) = 0, \quad (2.1)$$

where c is the speed of propagation, that can depend on x . The equation is subjected to the initial conditions

$$u(\mathbf{x}, 0) = f(\mathbf{x}), \quad u_t(\mathbf{x}, 0) = g(\mathbf{x}),$$

where f and g are given functions. The scalar wave equation is used e.g. in acoustics, where the evolution of the pressure and particle velocity is modeled.

Systems of wave equations are used to model elasticity and electromagnetism. The elastic wave equation is

$$\rho \mathbf{u}_{tt} = \mathbf{f} + (\lambda + 2\mu) \nabla(\nabla \cdot \mathbf{u}) - \mu \nabla \times (\nabla \times \mathbf{u}), \quad (2.2)$$

where λ and μ are the so-called Lamé parameters describing the elastic properties of the medium, ρ is the density, \mathbf{f} is the body force and \mathbf{u} is the displacement vector. Maxwell's equations are used to model problems in electromagnetics. They are given by

$$\begin{aligned} \frac{\partial \mathbf{D}}{\partial t} - \nabla \times \mathbf{H} &= -\mathbf{J} \\ \frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} &= 0 \\ \nabla \cdot \mathbf{D} &= \rho \\ \nabla \cdot \mathbf{B} &= 0, \end{aligned}$$

where \mathbf{H} is the magnetic field, \mathbf{J} is the current density, \mathbf{D} is the electric displacement, \mathbf{E} is the electric field, \mathbf{B} is the magnetic flux density and ρ is the electric charge density. In many situations, the elastic wave equation and the Maxwell equations can be reduced to sets of scalar wave equations for the individual components, but typically these equations couple, e.g. via boundary conditions.

Assuming time-harmonic waves, i.e. waves of the form

$$u(\mathbf{x}, t) = v(\mathbf{x})e^{i\omega t},$$

where ω is the frequency, the wave equation can be reduced to the Helmholtz equation,

$$\Delta v(\mathbf{x}) + \frac{\omega^2}{c(\mathbf{x})^2}v(\mathbf{x}) = f(\mathbf{x}). \quad (2.3)$$

Thus, the Helmholtz equation represents a time-harmonic form of the wave equation.

Another formulation of wave propagation problems is the *scattering problem*. Scattering problems concern propagation of waves that collide with some object. More precisely, let Ω be an object, a scatterer, that is illuminated by an incident wave field u^{inc} . Then, the wave field that is generated when u^{inc} collides with Ω is the scattered field u^s (see Figure 2.1). There are many examples of scattering problems. One example in nature is a rainbow that appears when the sunlight is scattered by rain drops. Another example is the scattering of light in air, which is the reason for the blue color of the sky. To describe the scattering problem mathematically, let us consider an impenetrable obstacle Ω in \mathbb{R}^3 and assume that the speed of propagation outside Ω is homogeneous, $c(\mathbf{x}) = 1$. The obstacle is illuminated by some known incident wave $u^{inc}(\mathbf{x}, t)$ which solves the constant coefficient Helmholtz equation pointwise, for example a plane wave $e^{i\omega\mathbf{x}\cdot\hat{s}}$ in direction \hat{s} . We also assume that $u^{tot} = u^{inc} + u^s = 0$ on the boundary of the obstacle $\partial\Omega$. Then, the scattering problem is to find the scattered field $u^s(\mathbf{x})$ such that it satisfies the Helmholtz equation

$$\Delta u^s(\mathbf{x}) + \omega^2 u^s(\mathbf{x}) = 0, \quad \mathbf{x} \in \mathbb{R}^3 \setminus \bar{\Omega} \quad (2.4)$$

with boundary condition

$$u^s(\mathbf{x}) = -u^{inc}(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega, \quad (2.5)$$

and the Sommerfeld radiation condition

$$\lim_{r \rightarrow \infty} r \left(\frac{\partial u^s}{\partial r} - i\omega u^s \right) = 0, \quad (2.6)$$

where $r = |\mathbf{x}|$, which guarantees that the scattered wave is outgoing. To solve this problem numerically, it is typically reformulated as a boundary integral equation. One version that is suitable for high frequency problems [17] is

$$\frac{1}{2} \frac{\partial u}{\partial n}(x) + \int_{\partial\Omega} \left(\frac{\partial \Phi(x, y)}{\partial n(x)} - i\eta \Phi(x, y) \right) \frac{\partial u}{\partial n}(y) ds(y) = f(x), \quad x \in \partial\Omega, \quad (2.7)$$

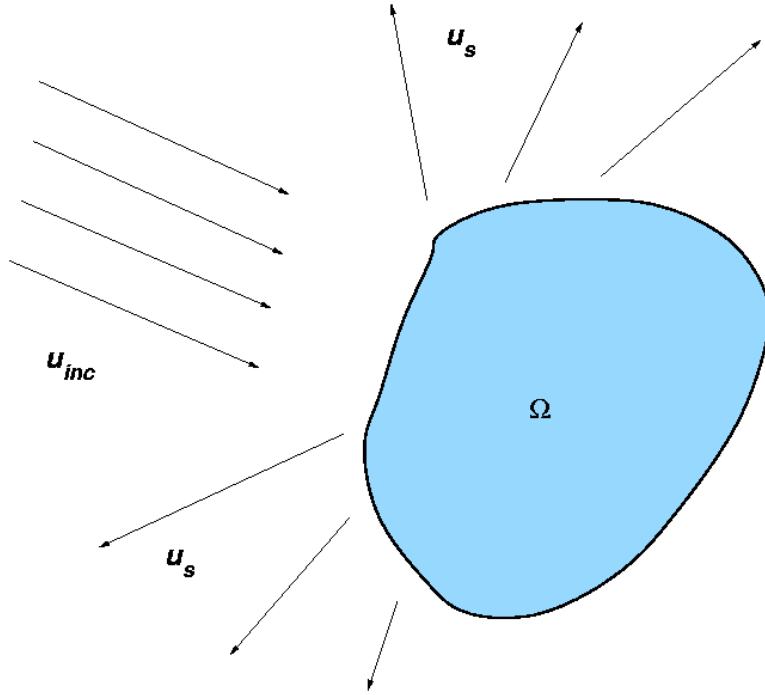


Figure 2.1: Scattering problem. An incident field u_{inc} collides with a scatterer Ω and an scattered field u_s is generated.

where

$$f(x) = \frac{\partial u^{inc}}{\partial n}(x) - i\eta u^{inc}(x),$$

$\eta = \eta(\omega) > 0$ is a coupling parameter that ensures the well-posedness of (2.7) and $\partial u/\partial n$ is to be determined. Here $\Phi(x, y, \omega)$ denotes the standard fundamental solution of the Helmholtz equation in \mathbb{R}^3 , which becomes increasingly oscillatory when $\omega \rightarrow \infty$.

One important difference between scattering problems in homogeneous media and the domain problems (2.1) and (2.3) with variable media is in the way the waves are scattered. In the domain problems the waves are, in principle, scattered whenever the wave speed changes, while in the scattering problems the waves are scattered only from the surface of the scatterer.

2.1 Geometrical Optics

Direct computation of high frequency wave propagation problems requires many discretization points/ elements to resolve the solution, and therefore it is very computationally expensive. Instead, asymptotic approximations, such as geometrical optics (GO), can be used. See for example [28, 78, 64]. The key idea in GO is that the highly oscillating solution is represented as a product of a slowly variable amplitude function A and an exponential function of the slowly variable phase function ϕ multiplied by the large parameter ω . Hence, instead of the oscillating wave field, the unknowns in GO are the phase and the amplitude, which vary on a much coarser scale than the full solution (see Figure 2.2). They are therefore easier to compute numerically, at a cost independent of the frequency. However, the approximation is only accurate for large frequencies. It typically requires that variations in the speed of propagation $c(x)$ are on a scale much larger than the wave length.

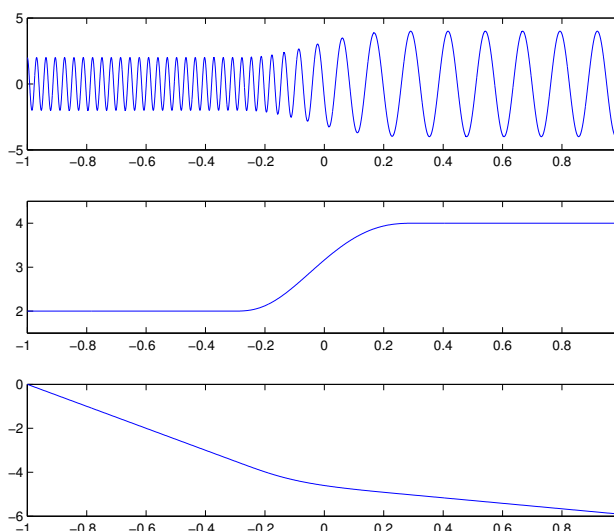


Figure 2.2: Real part of the solution of Helmholtz equation (top), the amplitude (middle) and the phase (bottom). The amplitude and the phase vary on a much coarser scale than the solution.

Consider the Helmholtz equation (2.3). In the GO approximation its solution

is sought in the form (asymptotic WKB expansion)

$$u(\mathbf{x}) = e^{i\omega\phi(\mathbf{x})} \sum_{k=0}^{\infty} A_k(\mathbf{x})(i\omega)^{-k}. \quad (2.8)$$

Substituting (2.8) in (2.3) and equating to zero the coefficients of equal powers of ω yields the Hamilton-Jacobi type *eikonal* equation for the phase ϕ

$$|\nabla\phi| = \frac{1}{c}, \quad (2.9)$$

and a system of linear *transport* equations

$$\begin{aligned} 2\nabla\phi \cdot \nabla A_0 + \Delta\phi A_0 &= 0 \\ \dots & \\ 2\nabla\phi \cdot \nabla A_{n+1} + \Delta\phi A_{n+1} &= \Delta A_n, \end{aligned} \quad (2.10)$$

for the amplitudes A_k , $k = 0, 1, \dots$. For large frequencies, A_k , $k > 0$ in (2.8) can be discarded, so only the transport equation for A_0 remains to be solved in (2.10) and $u(\mathbf{x}) \approx A_0(\mathbf{x})e^{i\omega\phi(\mathbf{x})}$.

If the eikonal equation (2.9) is solved by the method of characteristics, GO can be formulated as a system of ordinary differential equations (ODEs). Let $\mathbf{p} = \nabla\phi$ and define a Hamiltonian

$$H(\mathbf{x}, \mathbf{p}) = c(\mathbf{x})|\mathbf{p}|.$$

Then, instead of (2.9), we solve the following system of ODEs

$$\frac{d\mathbf{x}}{dt} = \nabla_{\mathbf{p}} H(\mathbf{x}, \mathbf{p}) = c(\mathbf{x}) \frac{\mathbf{p}}{|\mathbf{p}|}, \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (2.11)$$

$$\frac{d\mathbf{p}}{dt} = -\nabla_{\mathbf{x}} H(\mathbf{x}, \mathbf{p}) = -|\mathbf{p}|\nabla c(\mathbf{x}), \quad \mathbf{p}(0) = \mathbf{p}_0, \quad (2.12)$$

where $(\mathbf{p}(t), \mathbf{x}(t))$ is a bicharacteristic related to the Hamiltonian $H(\mathbf{x}, \mathbf{p})$. The parametrization t corresponds to the phase of the wave $\phi(x(t)) = \phi(x_0) + t$. In the special case when $H \equiv 1$, (2.11, 2.12) reduces to

$$\frac{d\mathbf{x}}{dt} = c(\mathbf{x})^2 \mathbf{p}, \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (2.13)$$

$$\frac{d\mathbf{p}}{dt} = -\frac{\nabla c}{c}, \quad \mathbf{p}(0) = \mathbf{p}_0, \quad |\mathbf{p}_0| = \frac{1}{c(\mathbf{x}_0)}. \quad (2.14)$$

There is also a system of ODEs for the amplitude.

Finally, one can introduce a kinetic model of GO. It is based on the interpretation that rays are trajectories of particles following Hamiltonian dynamics. Introduce the phase space $(t, \mathbf{x}, \mathbf{p})$ such that the evolution of particles in this space

is given by the system of equations (2.11)–(2.12). Letting $f(t, \mathbf{x}, \mathbf{p})$ be a particle density function, it will satisfy the Liouville equation

$$f_t + \nabla_p H \cdot \nabla_x f - \nabla_x H \cdot \nabla_p f = 0, \quad (2.15)$$

where $\nabla_p H$ and $\nabla_x H$ are given by (2.11)–(2.12), [7, 27]. With $H(\mathbf{x}, \mathbf{p}) = c(\mathbf{x})|\mathbf{p}| \equiv 1$ (2.15) reduces to the Vlasov-type equation

$$f_t + c^2 \mathbf{p} \cdot \nabla_x f - \frac{1}{c} \nabla_x c \cdot \nabla_p f = 0, \quad (2.16)$$

with initial data $f_0(\mathbf{x}, \mathbf{p})$ vanishing whenever $|\mathbf{p}| \neq 1/c$.

Asymptotic approximations, such as GO, cannot capture typical wave properties such as diffraction. Diffracted waves are produced when the incident field hits edges or vertices of the obstacle, or when the incident wave strikes the tangent points of the smooth scatterer (creeping waves). To overcome the problem with diffracted fields, asymptotic expansions with correction terms were proposed by Keller [45] in his geometrical theory of diffraction (GTD). GTD makes up for the GO solution on the boundary of the scatterer, where some terms of this solution vanish. GTD expansion of the solution accounts for the phase and the amplitude of the diffracted waves. It does so by means of several laws of diffraction which allow the use of the same principles as in GO to assign a field to each diffracted ray.

Another disadvantage of GO is that it fails when the amplitude A_0 is unbounded, at caustics where waves focus. One way to remedy this is to use Gaussian beams [60, 57, 4].

2.2 Numerical Methods

In this section we describe different numerical methods for computing high frequency wave propagation problems. We consider classical direct numerical methods as well as GO based methods.

Direct Numerical Methods

Direct numerical methods for wave propagation include finite difference methods, finite element methods and finite volume methods applied to (2.1) or (2.3). See, e.g. [39, 31, 8, 49] for an introduction to such methods. When applied to high frequency wave propagation problems, these methods require a certain number of grid points or elements per wavelength to maintain a fixed accuracy. Hence, if N is the number of grid points in each coordinate direction and ω is the frequency, one needs at least $N \sim \omega$.

When applied to the d -dimensional Helmholtz equation, the methods lead to sparse systems of equations of size N^d . Thus, direct methods, like Gaussian elimination, become computationally too expensive in higher dimensions. The alternative is to use iterative methods. However, since the systems of equations are

indefinite and ill-conditioned, iterative methods have a slow convergence rate. The convergence rate can be improved by preconditioning, but finding a good preconditioner for the Helmholtz equation is still a challenge [32]. See, however, [80] for some recent breakthroughs in this area.

Scattering problems can be solved by integral equations of the type (2.7). This is typically done with a collocation or Galerkin method, which again transforms the problem into a large linear system of equations, now of size N^{d-1} . However, the matrix in the system is dense and the computational cost of direct methods is therefore normally $\mathcal{O}(N^{3(d-1)})$. Thus, when ω is large, direct methods are not feasible. The alternative is to use iterative methods. Then, the computational cost can be reduced to $\mathcal{O}(N^{2(d-1)})$. However, for large ω this can still be expensive. Fast multipole methods can reduce the complexity to $\mathcal{O}(N^d \log N)$ for any fixed accuracy [61, 30].

Hence, the computational cost of direct numerical methods for all formulations grows algebraically with the frequency ω . Therefore, at sufficiently high frequencies ($\omega \gg 1$), direct numerical simulation is no longer feasible.

GO-based Methods

Methods based on the GO approximation are suitable for high frequency wave propagation problems. For different mathematical models of GO, there are various numerical methods.

PDE-based methods are used for the eikonal and transport equations (2.9) and (2.10). The equations are solved directly on a uniform grid to control the error. The eikonal equation is a Hamilton-Jacobi type equation and it has a unique viscosity solution, which can be computed numerically by finite difference methods, for example. Some of them are upwind-based methods like the fast marching method [66, 76] and the fast sweeping method [75, 82]. For time-dependent eikonal equations one can use high-resolution methods of ENO and WENO type [54]. However, at points where wavefronts collide and overlap, the viscosity solution is not enough because the eikonal and the transport equation describe only one unique wave at a time. Therefore, some other numerical techniques must be used. Some examples are the big ray tracing method [5], the slowness matching method [71] or the domain decomposition based on detecting kinks in the viscosity solution [6]. In these methods, the full solution is obtained by solving several eikonal equations.

Ray tracing methods [15, 43, 47, 72] are used to solve the ray equations (2.11)–(2.12). That system can be augmented with another system of ODEs for the amplitude. The ODEs are solved with standard numerical methods, such as second or fourth-order Runge-Kutta methods. The phase and the amplitude are calculated along the rays and interpolation must be used to obtain the solution in points other than the grid points. However in regions with diverging or crossing rays, this method is not efficient.

Methods based on the kinetic formulation of GO are phase space methods. The Liouville equation (2.15) has a large number of independent variables, which makes

computations very expensive. To overcome this problem, one can use wave-front methods and moment-based methods.

Wave-front methods include tracing of wave fronts in the phase space. They are based either on the kinetic formulation (2.16) or on the ray tracing formulation (2.13,2.14). Solutions including the formation of caustics can be computed by these methods. Some of the wave-front methods are the level set method [53] and the segment projection method [29, 73], which are PDE-based methods, and the wave front construction method [77], which is an ODE-based method. The wave front construction method introduced in [77] uses marker points to represent the interface. They are then evolved by (2.13,2.14). If the resolution of the interface deteriorates, new marker points are added to preserve the accuracy of the solution. These methods are described further in Chapter 4.

In moment-based methods, the kinetic transport equation set in the high-dimensional phase space $(t, \mathbf{x}, \mathbf{p})$ is approximated by a finite system of moment equations in the reduced space (t, \mathbf{x}) . In that way, the number of unknowns is decreased. For more details, see [7, 63].

Chapter 3

Numerical Methods with Weakly Frequency-Dependent Cost

The situation described in the previous chapter can be summarized as follows: For direct methods the computational cost grows with frequency for fixed accuracy, while for GO methods the accuracy grows with frequency for fixed computational cost. Unfortunately, the frequency range and accuracy requirement of many realistic problems is often intractable with either of these approaches.

Recently a new class of algorithms has been proposed that combines the cost advantage of GO methods with the accuracy advantage of direct methods. They are thus characterized by a computational cost that grows slowly with the frequency, while at the same time being accurate also for moderately high frequencies. It should be noted that one needs at least $\mathcal{O}(\omega)$ points to resolve the solution. If that many points are used, then the computational cost is, of course, at least $\mathcal{O}(\omega)$. In these methods, however, one does not resolve the solution, but computes instead GO-like solutions. Therefore, one can only expect to get the full solution in $\mathcal{O}(1)$ points.

The main interest of the new methods has been for scattering problems [10, 48, 24, 1, 41]. These methods are based on the integral formulation of the Helmholtz equation (2.7). The key step is to write the surface potentials as a slowly varying function multiplied by a fast phase variation. Instead of approximating the unknown function $v := \partial u / \partial n$ directly, the following ansatz is used

$$v(x, \omega) \approx \omega e^{i\omega\phi(x)} V(x), \quad x \in \partial\Omega. \quad (3.1)$$

The basic idea is that, using asymptotic analysis, the phase function ϕ can be determined in such a way that $V(x)$ is much less oscillatory than the original unknown function v . More precisely, $\phi(x)$ is taken as the phase of the geometrical optics approximation. For instance, when u^{inc} is a plane wave in direction \hat{d} , then $\phi = x \cdot \hat{d}$. Hence, the ansatz becomes

$$v(x, \omega) = \omega e^{i\omega x \cdot \hat{d}} V(x, \omega), \quad (3.2)$$

where V now varies slowly with ω . Each side of (2.7) is then multiplied by $e^{-i\omega x \cdot \hat{d}}$ to obtain

$$\frac{1}{2}V + DV - i\eta SV = i(\omega \hat{d} \cdot \hat{n} - \eta), \quad (3.3)$$

where the operators S and D are defined as follows

$$SV(x) = \int_{\partial\Omega} \Phi(x, y, \omega) e^{i\omega(y-x) \cdot \hat{d}} V(y) dy, \quad (3.4)$$

$$DV(x) = \int_{\partial\Omega} \frac{\partial \Phi(x, y, \omega)}{\partial n(x)} e^{i\omega(y-x) \cdot \hat{d}} V(y) dy. \quad (3.5)$$

Thus, the problem becomes to find the amplitude $V(x, \omega)$ by solving the integral equation (3.3). The amplitude varies slowly away from the shadow boundaries and can be represented by a fixed set of grid points, i.e. independently of the frequency. The integral equation can be solved for example with the collocation method. In this method, one needs to compute integrals of type

$$\sum_{j=1}^N c_j \int_a^b \Phi(x_i, y, \omega) \varphi_j(y) ds(y) = f(x_i), \quad i = 1, \dots, N. \quad (3.6)$$

Although the amplitude is a slowly varying function, the integrals cannot be computed independently of the frequency by direct numerical methods, since their kernels are oscillatory (c.f. (3.4)–(3.5)). To overcome this problem, one can use hybrid methods, which, for instance, can be based on a collocation [40] or a Nyström approach [10, 9]. Since the values of the integrands and their derivatives in critical points make the only significant contributions to the oscillating integrals, an integration procedure based on localization around these points was introduced. Since the amplitude near shadow boundaries is varying rapidly, those regions are treated with special consideration. However, the method works mainly for problems with convex scatterers. For more information about these methods, as well as their extensions to non-convex scatterers see [14].

Rigorous proofs of the low cost of these methods is difficult, and requires detailed results on the asymptotic behavior of the exact solution near shadow boundaries. An example of a result, due to Dominguez, Graham and Smyshlyaev [24], concerns convex smooth scatterers in 2D. They show that for a Galerkin discretization with N unknowns, where the integrals corresponding to (3.4), (3.5) are computed exactly, the relative error in $V(y)$ can be estimated as

$$rel. err \sim k^\alpha \left(\left(\frac{k^{1/9}}{N} \right)^6 + e^{-Ck^{1/3}} \right),$$

where α is a small exponent less than one and $k := \omega/c$ is the wave number. Hence, if N is slightly larger than $k^{1/9}$ the error is controlled as k grows.

For full domain problems with variable wave speed, much less has been done. In some sense this is more difficult than the scattering problem in a homogeneous

medium, because the waves are reflected at all points where the wave speed changes, not only at the surface of a scatterer. One attempt at lowering the computational cost along the lines above has been to use "plane wave basis functions" in finite element methods [56, 34, 13]. The method can be seen as a discontinuous Galerkin method with a particular choice of basis functions. However, except in simple cases, these methods do not reduce the complexity more than by a constant factor. A related method was proposed by Giladi and Keller [34]. It is a hybrid numerical method for the Helmholtz equation, in which the finite element method is combined with GO. The idea is to determine the phase factor which corresponds to the plane wave direction a priori by solving the eikonal equation for the phase using ray tracing, and then to determine the amplitude by a finite element method, choosing asymptotically derived basis functions which incorporate the phase factor.

3.1 Fast Method for Solving the High Frequency Helmholtz Equation in One Dimension

In this thesis, we suggest and analyze a numerical method for the high frequency Helmholtz equation in a bounded, one-dimensional domain with a variable wave speed function. The method is based on wave splitting. The Helmholtz equation is split into one-way wave equations, with source functions which are solved iteratively for a given tolerance. The source functions depend on the wave speed function and on the solutions of the one-way wave equations from the previous iteration. The solution of the Helmholtz equation is then approximated by the sum of the one-way solutions at every iteration. To improve the computational cost, the source functions are thresholded and, in the domain where they are equal to zero, the one-way wave equations are solved with geometrical optics, with a computational cost independent of the frequency. Elsewhere, the equations are fully resolved with a Runge-Kutta method. We have been able to show rigorously in one dimension that the algorithm is convergent and that for fixed accuracy, the computational cost is just $O(\omega^{1/p})$ for a p -th order Runge-Kutta method. Numerical experiments indicate that the growth rate of the computational cost is much slower than a direct method, and can be close to the asymptotic rate.

We now give some more details of the method. We consider the 1D Helmholtz equation

$$u_{xx} + \frac{\omega^2}{c(x)^2}u = \omega f, \quad x \in (-L, L), \quad (3.7)$$

where ω is the frequency and $c(x)$ is the wave-speed function such that $\text{supp}(c_x) \subset (-L, L)$. It is augmented with the non-reflecting boundary conditions

$$u_x(-L) - i\omega u(-L) = -2i\omega A, \quad (3.8)$$

$$u_x(L) + i\omega u(L) = 0, \quad (3.9)$$

which also incorporate an incoming wave with amplitude A . At high frequencies, GO is a good approximation to the solution. We want to find a way to correct for

the errors it makes at lower frequencies. A natural idea would be to use the system of equations (2.10), which is obtained in GO when the solution is approximated by (2.8). However, the series in (2.8) does not converge, even in simple settings. It is only an asymptotic series. The main problem is that (2.9,2.10) only describes waves travelling in one direction. In reality, waves are reflected whenever $c_x \neq 0$. We therefore introduce the functions v and w to describe waves propagating in the right-going and the left-going directions, respectively. They satisfy the following one-way wave equations,

$$i\omega v + c(x)v_x - \frac{1}{2}c_x(x)v = \frac{\alpha(x)(v+w)}{2i\omega}, \quad (3.10)$$

$$i\omega w - c(x)w_x + \frac{1}{2}c_x(x)w = \frac{\alpha(x)(v+w)}{2i\omega}, \quad (3.11)$$

where

$$\alpha(x) = \frac{1}{2}cc_{xx} - \frac{1}{4}c_x^2. \quad (3.12)$$

The full solution is the sum of these functions, $u = v + w$. We now make a WKB type expansion of v and w in powers of ω ,

$$v = \sum_{n=0}^{\infty} r_n \omega^{-n}, \quad w = \sum_{n=0}^{\infty} s_n \omega^{-n}. \quad (3.13)$$

Then, (3.10) and (3.11) become

$$\sum_{n=0}^{\infty} \left(i\omega r_n + c(x)\partial_x r_n - \frac{1}{2}c_x(x)r_n - \frac{\alpha(x)}{2i}(r_{n-1} + s_{n-1}) \right) \omega^{-n} = 0,$$

$$\sum_{n=0}^{\infty} \left(i\omega s_n - c(x)\partial_x s_n + \frac{1}{2}c_x(x)s_n - \frac{\alpha(x)}{2i}(r_{n-1} + s_{n-1}) \right) \omega^{-n} = 0,$$

where $r_{-1} = s_{-1} = 0$. Defining $v_n = r_n \omega^{-n}$ and $w_n = s_n \omega^{-n}$ and setting each term to zero, we obtain for $x \in (-L, L)$ and $n \geq 0$

$$i\omega v_n + c(x)\partial_x v_n - \frac{1}{2}c_x(x)v_n = -\frac{1}{2i}f_n(x), \quad (3.14)$$

$$i\omega w_n - c(x)\partial_x w_n + \frac{1}{2}c_x(x)w_n = -\frac{1}{2i}f_n(x), \quad (3.15)$$

where

$$f_0(x) = \omega f(x), \quad f_{n+1}(x) = -\alpha(x)(v_n(x) + w_n(x)). \quad (3.16)$$

We then approximate $u(x) = v(x) + w(x)$ by $z_m(x)$, obtained by taking the first m terms in the sums in (3.13),

$$u(x) \approx z_m(x) = \sum_{n=0}^m (v_n(x) + w_n(x)). \quad (3.17)$$

3.1. FAST METHOD FOR SOLVING THE HIGH FREQUENCY HELMHOLTZ EQUATION IN ONE DIMENSION 19

We also need to specify initial conditions for (3.14) and (3.15). We have proved in the first paper that z_m will satisfy the boundary conditions (3.8) and (3.9) for all m if we let

$$v_n(-L) = \begin{cases} A, & n = 0 \\ 0, & n > 0 \end{cases}, \quad w_n(L) = 0, \quad \forall n. \quad (3.18)$$

To sum up, we solve (3.14) and (3.15) for $n = 0, 1, 2, \dots, m$ with the given initial conditions (3.18) and the source function f_n that is defined by (3.16). Then, the solution $u(x)$ of the Helmholtz equation (3.7) can be well approximated by z_m given in (3.17). For proof see Theorem 1 in the first paper. Moreover, in contrast to (2.8), the series (3.17) converges quickly for large ω .

In a direct implementation, the computational complexity of solving (3.14),(3.15) would grow algebraically with the frequency ω just as for the full Helmholtz equation, since the solution is oscillatory. To get around this we note that (3.14) and (3.15) can be simplified when $f_n = 0$. Then, using the ansatz $v_n = Ae^{i\omega\phi}$ in (3.14) we obtain equations for A and ϕ ,

$$\partial_x \phi = \frac{1}{c(x)}, \quad \partial_x A = \frac{c_x(x)}{2c(x)} A(x). \quad (3.19)$$

This is in fact GO, and can be solved at a cost independent of the frequency. Similar equations can be obtained when the ansatz is used in (3.15). Thus, the computational cost estimate can be improved by approximating the forcing functions with zero when they are small. That is exactly what we do in our method.

More precisely, let \hat{f}_n be the approximate forcing function, and \hat{v}_n and \hat{w}_n the corresponding approximate one-way wave equation solutions, with initial data (3.18) and let

$$\text{trunc}(x, \delta) = \begin{cases} 0, & |x| < \delta \\ x, & \text{otherwise} \end{cases}$$

be the truncation function. Then, we first calculate the forcing function from the approximate one-way solutions in the same way as before, by (3.16) with \hat{v}_n and \hat{w}_n instead of v_n and w_n and define \hat{f}_n as the thresholded version of f_n

$$\hat{f}_n(x) = \text{trunc}(f_n(x), \text{Tol}_n), \quad \text{Tol}_n = \frac{\omega \text{Tol}}{2^{n+1}L}, \quad (3.20)$$

for some tolerance Tol. We use GO where $\hat{f}_n = 0$ and a fully resolved ODE method elsewhere, see Figure 3.1.

For this case, also the solution $u(x)$ of the Helmholtz equation (3.7) can be well approximated by

$$u(x) \approx \hat{z}_m(x) := \sum_{n=0}^m (\hat{v}_n(x) + \hat{w}_n(x)). \quad (3.21)$$

For the proof, see Theorem 1 in the first paper. Thus, the system of one-way wave equations can be solved independently of the frequency in the part of the

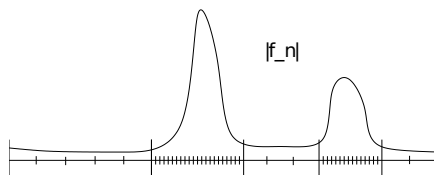


Figure 3.1: Function $|f_n|$. In a domain where $|f_n(x)| < \text{Tol}_n$, $\hat{f}_n(x) = 0$ and we can use GO to solve the system of one way wave equations, otherwise $\hat{f}_n(x) = f_n(x)$ and some p -th order ODE numerical scheme can be used.

domain where $\hat{f}_n = 0$. In the part of the domain where $\hat{f}_n \neq 0$, a direct ODE numerical method can be used. The solution of (3.7) is then approximated by (3.21). Moreover, we have shown that the size of the region where $\hat{f}_n \neq 0$ is $\mathcal{O}(1/\omega)$ and that this implies an almost frequency-independent computational complexity (see Theorem 1 in the first paper). More precisely, with the right choice of stepsize,

$$\Delta x_f \sim \frac{\text{Tol}^{1/p}}{\omega^{1+1/p}}, \quad (3.22)$$

the computational cost will be $\mathcal{O}(\omega^{1/p})$, where p is the order of the ODE scheme. This is a significant improvement compared to the direct methods, where the complexity is $\mathcal{O}(\omega)$. For a more detailed description of the method, see the first paper in this thesis.

Let us now show one example. Consider the non-symmetric function $c(x)$ that is shown in Figure 3.2 (top). The absolute value of the solution for $\omega = 16$ is plotted in Figure 3.2 (bottom). The oscillations are due to the reflected waves which would not be present in GO. The error, computational cost and the number of iterations needed to compute the solution using our method for $\text{Tol} = 0.1$, are shown in Figure 3.3. We use the 4-th order Runge-Kutta method to compute the solution in intervals where $|f_n(x)| > \text{Tol}_n$. The computational cost grows essentially as $\omega^{1/4}$, as predicted by the asymptotic theory.

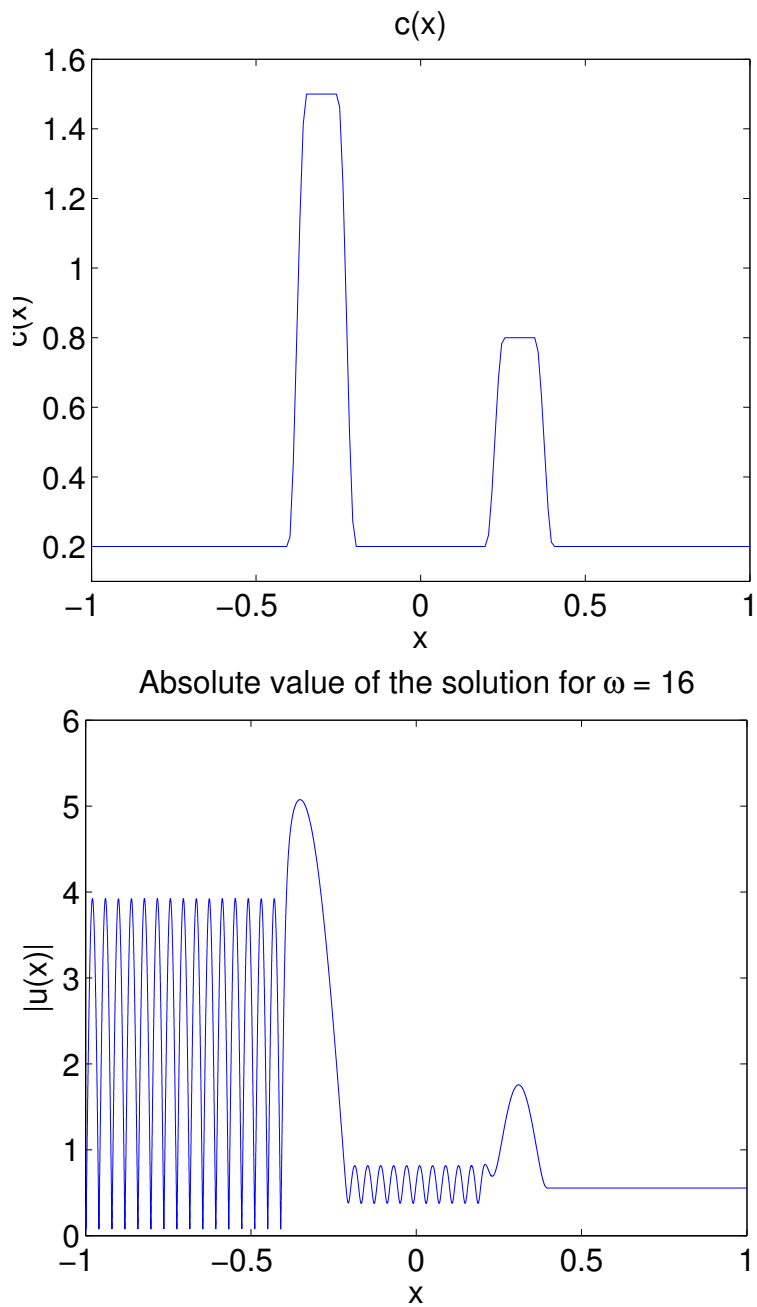


Figure 3.2: Function $c(x)$ (top). Absolute value of the solution of the Helmholtz equation for $\omega = 16$ (bottom).

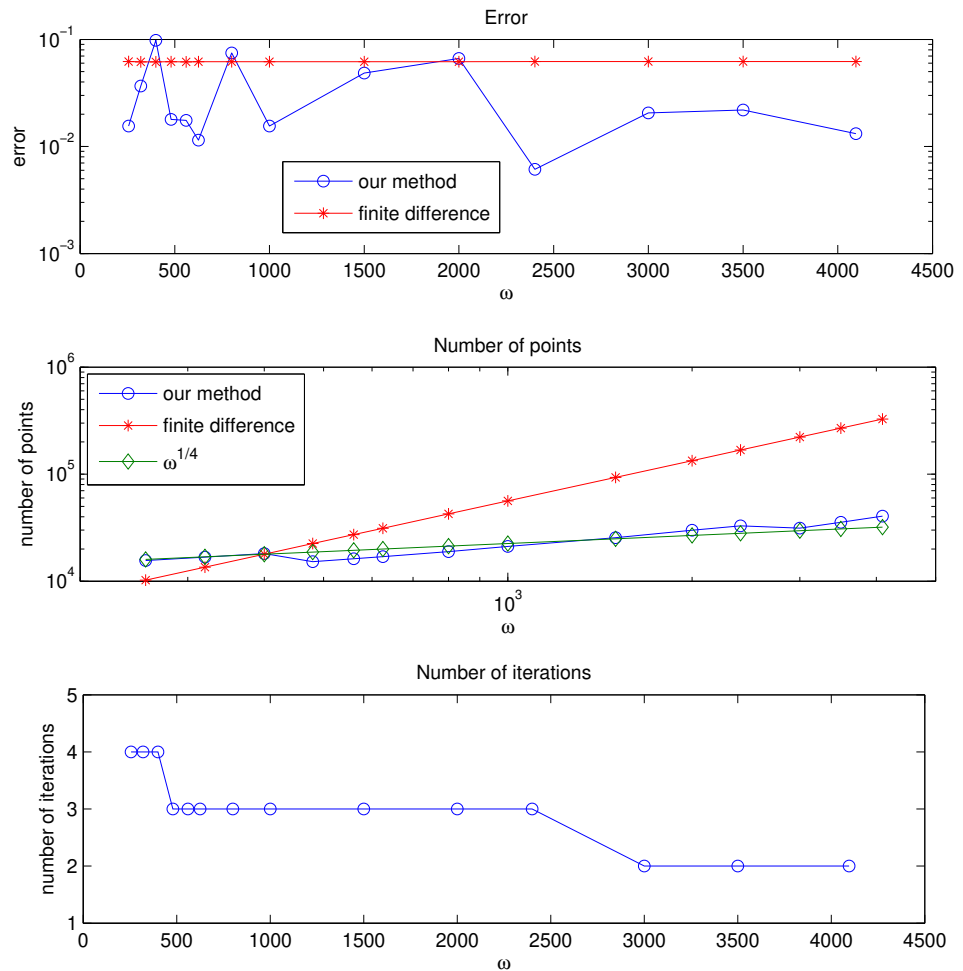


Figure 3.3: Comparison between the finite-difference method and our method with tolerance $\text{Tol} = 0.1$, for $256 \leq \omega \leq 4096$, when the speed function is given in Figure 3.2. The error (top), the computational cost (middle) and the number of iterations needed to compute the solution with our method (bottom). The 4-th order Runge-Kutta method is used to compute the solution in intervals where $|f_n(x)| > \text{Tol}_n$.

Chapter 4

Interface Tracking

Propagating interfaces occur in many applications. Wave propagation, multiphase flow, crystal growth, melting, epitaxial growth and flame propagation are some examples. Therefore, development of fast and accurate numerical methods for interface tracking is very important. In this chapter, we present the equations that are used to model propagation of interfaces in a time-varying velocity field. We also give an overview of existing numerical methods for the problem, and discuss their advantages and disadvantages.

We consider interfaces that move with some given velocity field that does not depend on the interface. If we assume that the interface can be parametrized so that it is described by the function $x(t, s) : \mathbb{R}^+ \times \mathbb{R}^q \rightarrow \mathbb{R}^d$ for a fixed time t , with the parametrization $s \in \Omega \subset \mathbb{R}^q$, then it satisfies the following ODE

$$\frac{dx(t, s)}{dt} = F(t, x(t, s)), \quad x(0, s) = \gamma(s), \quad s \in \Omega, \quad (4.1)$$

where $F(t, x) : \mathbb{R}^+ \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ is the given velocity field and $\gamma(s) : \mathbb{R}^q \rightarrow \mathbb{R}^d$ is the initial interface. Depending on how the interface is represented, different numerical methods are used to solve (4.1).

4.1 Standard Front Tracking

In standard front tracking methods, the interface is represented explicitly by a set of marker points, which are then propagated using some ODE numerical method, see Figure 4.2 (left). For a fixed time, a front in two dimensions, $d = 2$, $q = 1$, is usually reconstructed by line segments, and a front in three dimensions, $d = 3$, $q = 2$, by a triangulation.

Let us now consider the case $q = 1$. Marker points are then defined by

$$x_j(t) = x(t, s_j), \quad j = 0, \dots, N,$$

where $\{s_j\}$, $j = 0, \dots, N$ is a discretization of Ω . When $q = 2$ and $d \geq 3$, one uses a triangulation of Ω where s_j are the nodes of triangles covering Ω . Note that s_j are then vectors in \mathbb{R}^q .

The equation (4.1) becomes

$$\frac{dx_j(t)}{dt} = F(t, x_j(t)), \quad x_j(0) = \gamma(s_j), \quad j = 0, \dots, N. \quad (4.2)$$

It is solved using some standard ODE method with a time step Δt .

One difficulty with these methods is that they are not suitable for problems in which the length or the area of the interface increases with time. The distance between marker points then becomes larger and the interface cannot be accurately resolved. As an illustration, let us consider the following velocity field

$$F(t, \mathbf{x}) = \begin{bmatrix} \tanh(-y^2 + 0.25) \\ \sin(2\pi x) \end{bmatrix}, \quad (4.3)$$

and solve (4.1) until $T = 4$, with $\gamma(s)$ being a line segment on the x -axis between zero and one. The problem is solved for different number of points on the interface, $N = 2000$, $N = 5000$, $N = 8000$, and $N = 10000$. The solution is shown in Figure 4.1. It can be observed that the interface cannot be accurately resolved at the final time, even when $N = 10000$. A standard way to resolve this problem is to adaptively add points whenever the distance between two consecutive points becomes larger than some given tolerance, i.e. if for two points on the interface x_j and x_{j+1}

$$|x_j - x_{j+1}| > \text{Tol} \quad (4.4)$$

for some given tolerance Tol, add a new point between them. This can be done by interpolation. For an illustration, see Figure 4.2 (right).

Another drawback is that these methods become complicated for problems in which the interface changes topology (e.g. when two interfaces merge). In that case, the connectivity of marker points has to be changed correctly, which is rather complicated, although it can be done [36, 35].

Let us finally note that the cost to propagate one marker point with a time step Δt to a fixed time is $O(1/\Delta t)$. Consequently, the cost to propagate a front that is described by N marker points is $O(N/\Delta t)$. The accuracy is $O(\Delta t^p)$ for a p -th order ODE method.

4.2 Level Set Methods

The level set method was introduced by Sethian and Osher [53]. In this method, the interface is represented implicitly as a level set of a continuous function ϕ . There are two formulations, the boundary value formulation and the initial value formulation, [67].

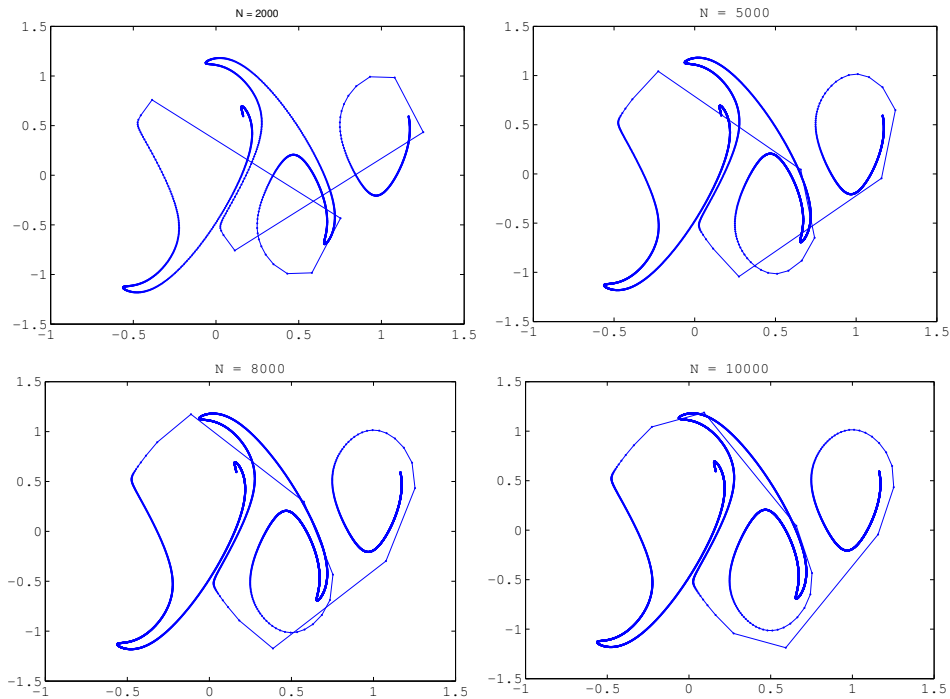


Figure 4.1: The interface at $T = 4$ obtained by moving N marker points. The velocity field is given by (4.3). The initial interface is the line segment between $x = 0$ and $x = 1$ on the x -axis.

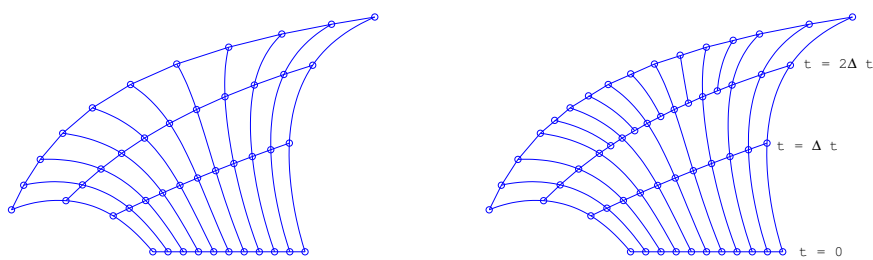


Figure 4.2: Interface construction. The interface is represented by marker points that are propagated by some numerical ODE method. When the points are too far apart, new marker points are inserted.

In the initial value formulation, we assume that the level set value of a point on the front with path $x(t, s)$ is zero, i.e.,

$$\phi(t, x(t, s)) = 0. \quad (4.5)$$

Differentiating with respect to t , we obtain

$$\phi_t + \dot{x} \cdot \nabla \phi = 0. \quad (4.6)$$

Assuming that the front moves in the normal direction with speed c , i.e.

$$F = c \frac{\nabla \phi}{|\nabla \phi|},$$

the initial value formulation is

$$\phi_t + c|\nabla \phi| = 0, \quad \text{given } \phi(x, t = 0). \quad (4.7)$$

For an illustration, see Figure 4.3.

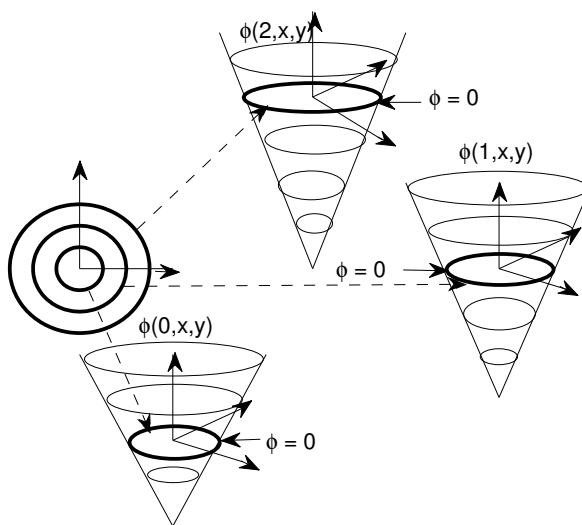


Figure 4.3: Transformation of front motion into an initial value problem.

To obtain the boundary value formulation, one lets $\phi(x(t, s))$ be the arrival time t of the front. To derive the equation, we thus assume that

$$\phi(x(t, s)) = t. \quad (4.8)$$

Differentiating (4.8) with respect to t , we obtain

$$\dot{x} \cdot \nabla \phi = 1.$$

Again, assuming that c is the speed in the normal direction, the boundary value formulation is given by

$$|\nabla \phi|c = 1, \quad \phi = 0 \text{ on } \gamma. \quad (4.9)$$

For an illustration, see Figure 4.4. If the speed c depends only on the position, $c = c(x)$, then the equation (4.9) becomes the eikonal equation.

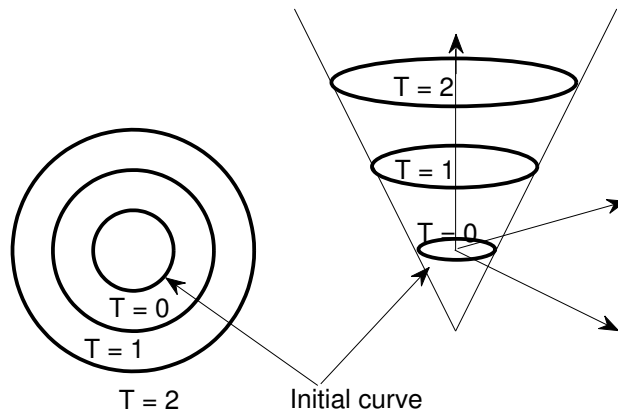


Figure 4.4: Transformation of front motion into a boundary value problem.

When \dot{x} only depends on x and $\nabla \phi$, equations (4.7) and (4.9) can be written in the form of Hamilton-Jacobi equations

$$\alpha \phi_t + H(\mathbf{x}, \nabla \phi) = 0, \quad (4.10)$$

where the Hamiltonian H and α , for the initial value problem, are given by

$$H(\mathbf{x}, \nabla \phi) = c \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2}, \quad \alpha = 1 \quad (4.11)$$

and, for the boundary value problem, by

$$H(\mathbf{x}, \nabla \phi) = c \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2} - 1, \quad \alpha = 0. \quad (4.12)$$

In general, \dot{x} can also depend on higher derivatives of ϕ , to model for instance fronts propagating with curvature-dependent speed [53].

To solve the initial value problem (4.7) numerically, one can use upwind numerical methods for Hamilton-Jacobi equations. The schemes are explicit and use techniques from hyperbolic conservation laws to solve Hamilton-Jacobi equations. Higher order methods for the initial value formulation include methods of ENO (essentially non-oscillatory) and WENO (weighted essentially non-oscillatory) type proposed by Osher and Shu in [54].

Methods used for the boundary value formulation (4.9) include the fast marching methods introduced by Sethian in [66]. The main idea in the fast marching methods is to use only upwind values to construct the solution ϕ in (4.9). In other words, to compute the solution at a new grid point, only the previously computed values at the neighboring grid points are used. The algorithm starts from the boundary data, and calculates the solution outward from the boundary in a computationally efficient way. The computational cost is $O(N \log N)$ where N is the total number of grid points. With some modifications, the cost can be reduced to $O(N)$, see for example [79]. The order of accuracy of the method is one. Constructing higher order fast marching methods is complicated, but possible [3]. Other methods with $O(N)$ complexity include fast sweeping methods [44, 82, 74] and group marching methods [46].

In general, the level set methods have a nice property, namely that the equations (4.7) and (4.9) are the same regardless of the dimension of the problem. Moreover, they do not change, even if the interface changes topology. Hence, topology changes, like interface merging for example, are easily handled by these methods. This property is very important in practical applications. Another advantage is that the error is controlled by the choice of the spatial discretization and, in the case of the initial value formulation, by the time discretization. Consequently, the methods are suitable for expanding interfaces.

On the other hand, since the interface is represented as the level set of a higher dimensional function ϕ , the dimension of the problem is always increased by one, leading to higher computational cost. Moreover, the initial value formulation requires that the time step Δt satisfies a CFL condition with respect to the maximum velocity over the whole domain, i.e. $\max_{\Omega} c \Delta t < \Delta x$, where Δx is the step used in the spatial discretization. Thus, the main problem of this approach is the computational cost. For example, the cost to solve (4.7) is $O(N/\Delta t)$ in any dimension, where N is the total number of grid points. However, often, the cost can be reduced by using local level set methods such as the narrow band level set method introduced by Chopp in [16]. The main idea of this method is to perform calculations only near the zero level set. The computational cost is then decreased to $O(N^{(d-1)/d} \log N/\Delta t)$, where N is the total number of grid points and d is the dimension of the problem. For more details, see [2, 52, 55, 69, 70].

4.3 The Segment Projection Method

In the segment projection method, the interface is divided into several parts, called segments. The segments overlap, and are chosen in such a way that they can be expressed as functions of one variable. For example, when the interface is a curve with points (x, y) , they are either expressed by $(f_j(y), y)$ or $(x, g_j(x))$ for some functions $f_j(y)$ and $g_j(x)$, where j is the index of the segment. The f_j and g_j functions are discretized on a uniform Eulerian grid in the y and x -direction, respectively. The information about their connectivity also has to be provided. For an illustration, see Figure 4.5. Hence, the interface is represented by marker points as in the standard front tracking methods, but the discretization is Eulerian as in the level set methods.

The front is moved by evolving the segments by PDEs given by the equations of motion. After every time step, the segments have to be reinitialized and the information about their connectivity has to be updated. Handling topology changes is a bit simpler than in the standard front tracking methods, but still complicated. Moreover, it is difficult to extend this method to higher dimensions. For a more detailed description of the method and its applications see [29, 73].

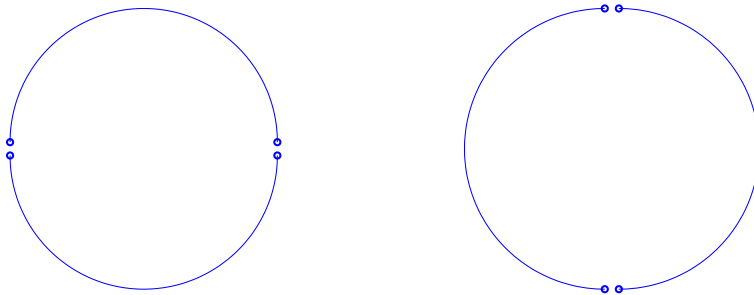


Figure 4.5: Segment structure when the interface is a circle. Left: y -segments, $(x, g_j(x))$, right: x -segments $(f_j(y), y)$.

Chapter 5

Multiresolution Interface Tracking

In this chapter, we describe interface tracking methods, where a multiresolution approximation of the interface is used for the interface description. This kind of methods are the topics of the second and the third papers of this thesis.

5.1 Multiresolution Description of the Interface

In multiresolution front tracking methods, the interface is represented hierarchically by increasingly detailed meshes. Starting from a representation on the finest level by a mesh that consists of marker points, the mesh is then coarsened level by level by removing points. For each removed point, a wavelet vector is computed as the difference between the point and a so-called predicted point, which only depends on the coarser level mesh. In that way, only points from the last coarse discretization and the wavelet vectors need to be saved. This representation is then used for the front tracking.

More precisely, assume that $\gamma(s)$ is twice continuously differentiable with respect to s , and non-self intersecting. Moreover, assume that the finest mesh consists of $N = 2^J$ discretization points, and for $0 \leq j \leq J$ define the marker points $x_{j,k} = \gamma(s_{j,k})$, $s_{j,k} = k2^{-j}$, $k = 0, \dots, 2^j$ on the curve γ . Then, the j th level multiresolution approximation of the curve is defined by the polyline

$$\gamma_j = \bigcup_{0 \leq k < 2^j - 1} l(x_{j,k}, x_{j,k+1}),$$

where $l(x_{j,k}, x_{j,k+1})$ is the line segment between points $x_{j,k}$ and $x_{j,k+1}$. Note that since $s_{j,2k} = s_{j-1,k}$, we have $x_{j,2k} = x_{j-1,k}$ for $k = 0, \dots, 2^{j-1}$, and the polyline γ_{j-1} therefore consists of every second point in γ_j . The wavelet vectors $w_{j,2k+1}$ for $k = 0, \dots, 2^{j-1} - 1$ are then defined as follows.

1. Compute the so-called predicted point by the midpoint formula, $x_{j,2k+1}^* = (x_{j-1,k} + x_{j-1,k+1})/2$,

2. Define $w_{j,2k+1}$ as the difference between the actual point and the predicted point, $w_{j,2k+1} = x_{j,2k+1} - x_{j,2k+1}^*$.

Hence, given $x_{j-1,k}$ and $w_{j,2k+1}$, we can reconstruct γ_j by first computing $x_{j,2k+1}^*$ and then adding $w_{j,2k+1}$. We repeat this for each level. Finally, the zeroth level multiresolution representation of γ is $\gamma_0 = l(x_{0,0}, x_{0,1})$. This is illustrated in Figure 5.1. Thus, to construct a multiresolution approximation of a curve, we first make a fine discretization of the interface, and then define wavelet vectors by

$$w_{j,2k+1} = x_{j,2k+1} - (x_{j-1,k} + x_{j-1,k+1})/2. \quad (5.1)$$

In this way, the curve γ can be uniquely represented, with a level of detail J , by the edge points $x_{0,0}$ and $x_{0,1}$ and the wavelet vectors $w_{j,2k+1}$, $j = 1, \dots, J$, $k = 0, \dots, 2^{j-1} - 1$. Note that one can reconstruct the curve's points $x_{j,k}$ on level J from the edge points and the wavelet vectors by reversing the previously described process. The cost is $O(N)$, where $N = 2^J$.

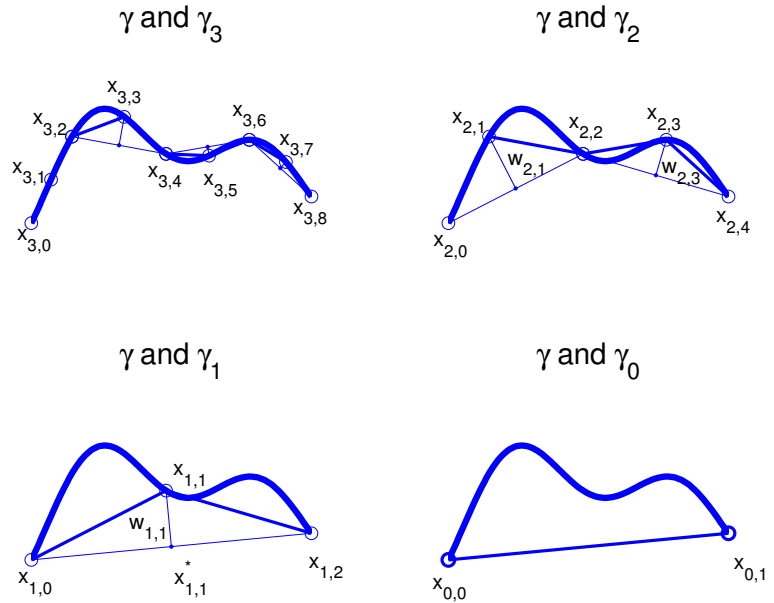


Figure 5.1: Constructing the polyline γ_j which represents the multiresolution approximation of the curve γ .

Remark 1. *In the special case when the wavelet vectors $w_{j,2k+1}$ are normal to the line segment $l(x_{j-1,k}, x_{j-1,k+1})$, only the lengths of wavelet vectors and edge points need to be saved. This approximation is known as a normal mesh, [38].*

So far we have used the midpoint formula to compute the predicted points. It is a special case of a process called subdivision. Subdivision is a procedure to iteratively create smooth curves and surfaces from an initial mesh [12, 20, 21, 25]. More precisely, let $\mathbf{x} = \{x_k\}$ be a sequence. A local, stationary subdivision scheme is then characterized by a bounded linear operator $S : \ell_\infty \mapsto \ell_\infty$, defined by a finite sequence $\mathbf{a} = \{a_k\}$ as follows:

$$(S\mathbf{x})_k = \sum_{\ell} a_{k-2\ell} x_\ell.$$

To build a smooth function, we start from a sequence \mathbf{x}_0 and associate to it a function $f_0(x)$ which is piecewise linear and interpolates $x_{0,k}$ on the integer grid, $f_0(k) = x_{0,k}$. We can then apply the subdivision scheme S iteratively and define \mathbf{x}_j for all $j > 0$ by

$$\mathbf{x}_{j+1} = S\mathbf{x}_j.$$

Similar to γ_j above, each \mathbf{x}_j defines a piecewise linear function f_j that interpolates $x_{j,k}$ in $s_{j,k} = k2^{-j}$. For many S these functions f_j converge to a smooth limit function.

A subdivision scheme is *interpolating* if $(S\mathbf{x})_{2k} = x_k$. Interpolating schemes are described by the mask that is used to compute odd (in k) points. For example, a two-point subdivision scheme is the midpoint scheme that is defined by the mask $[1/2, 1/2]$, i.e. $(S\mathbf{x})_{2k+1} = (x_k + x_{k+1})/2$.

The *order* of an interpolating subdivision scheme S is the largest q such that $SP(\mathbf{k}) = P(\mathbf{k}/2)$ for all polynomials P of degree $p < q$. We always assume that q is at least one, so that $S\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is a sequence which has all entries equal to 1. For example, the order of the two-point subdivision scheme is $q = 2$.

The approximation of the interface can be improved if some higher order subdivision scheme is used to calculate the predicted points. Some examples of higher order subdivision schemes that are also called the Lagrange subdivision schemes are:

- "4-point" subdivision scheme, order $q = 4$,

$$\text{mask} = \left[\frac{1}{16} \{-1, 9, 9, -1\} \right],$$

- "6-point" subdivision scheme, order $q = 6$,

$$\text{mask} = \left[\frac{1}{256} \{3, -25, 150, 150, -25, 3\} \right],$$

- "8-point" subdivision scheme, order $q = 8$,

$$\text{mask} = \left[\frac{1}{2048} \{-5, 49, -245, 1225, 1225, -245, 49, -5\} \right].$$

When the interface propagates, i.e. $x = x(t, s)$, we construct the wavelet vectors in the same way. Let $s_{j,k} = k2^{-j}$. Then, the curve $x(t, s)$ can be described by

$$x_{j,k}(t) = x(t, s_{j,k}), \quad 0 \leq k \leq 2^j.$$

Note that still $x_{j,2k}(t) = x_{j-1,k}(t)$. Let $\mathbf{x}_j(t) = \{x_{j,k}(t)\}$ and $\mathbf{w}_j(t) = \{w_{j,k}(t)\}$. The wavelet vectors are now defined by the relation

$$\mathbf{w}_j(t) = \mathbf{x}_j(t) - S\mathbf{x}_{j-1}(t), \quad (5.2)$$

where S is a subdivision operator. This is done recursively, as before, and gives a multiresolution description of the interface in terms of $\mathbf{x}_0(t)$ and $\mathbf{w}_j(t)$, $j = 1, \dots, J$, if the finest mesh has 2^J points.

Remark 2. Note that for a fixed t , (5.2) is equivalent to (5.1) if S is the 2-point subdivision scheme. Indeed, the relation (5.2) written component-wise becomes

$$w_{j,2k+1} = x_{j,2k+1} - S(\{x_{j-1,l}\}_{l=k}^{k+1}) = x_{j,2k+1} - (x_{j-1,k} + x_{j-1,k+1})/2.$$

5.2 Governing Equations

Let us now describe the equations that are to be solved in the front propagation problem, if the interface is represented by the described multiresolution approximation. Inserting (5.2) in (4.1), we get

$$\frac{d\mathbf{w}_{j+1}}{dt} = F(t, \mathbf{x}_{j+1}(t)) - SF(t, \mathbf{x}_j(t)) = F(t, S\mathbf{x}_j(t) + \mathbf{w}_{j+1}(t)) - SF(t, \mathbf{x}_j(t)).$$

Setting

$$G(t, \mathbf{x}, \mathbf{w}) = F(t, S\mathbf{x} + \mathbf{w}) - SF(t, \mathbf{x}),$$

we have the following system of ODEs

$$\frac{d\mathbf{w}_{j+1}(t)}{dt} = G(t, \mathbf{x}_j(t), \mathbf{w}_{j+1}(t)), \quad \frac{d\mathbf{x}_0(t)}{dt} = F(t, \mathbf{x}_0(t)), \quad (5.3)$$

which together with (5.2) describes the dynamics of the system. For the midpoint subdivision scheme, (5.3) can be written component-wise, with some abuse of notation, as

$$\frac{dw_{j+1,2k+1}(t)}{dt} = G(t, x_{j,k}(t), x_{j,k+1}(t), w_{j+1,2k+1}(t)), \quad (5.4)$$

where

$$G(t, x_L, x_R, w) = F\left(t, \frac{x_L + x_R}{2} + w\right) - \frac{F(t, x_L) + F(t, x_R)}{2}.$$

5.3 The Basic Fast Numerical Method

Let us now introduce the Basic fast numerical method for interface propagation. The method is based on a multiresolution representation of the interface described above, and exploits the following property of wavelet vectors. If S is a linear, stationary and interpolatory subdivision operator of order q and

$$x(t, s) \in C^{q+\ell}([0, T] \times [0, 1]; \mathbb{R}^d),$$

then

$$\left| \frac{d^\ell \mathbf{w}_j(t)}{dt^\ell} \right| \leq C_\ell(T) 2^{-jq}. \quad (5.5)$$

Hence, wavelet vectors and their derivatives decay as 2^{-jq} , where q is the order of the subdivision scheme used to compute predicted points, and j is the level [22, 65, 62]. This means that the finer scales evolve more slowly than the coarser scales, and in a numerical ODE solver, longer time steps can be used to evolve the wavelet vectors from higher levels. This is exactly the idea behind the method presented in [65]. The method uses a time step doubling technique to solve equations (5.3). This technique includes a time step that is uniform inside each level j , but differs for different levels. More precisely, the smallest time step, called the reference time step Δt , is chosen at level zero and then doubled at every level, so that the time step at level j is $\Delta t_j = 2^j \Delta t$. In order to describe the method in more details, we first introduce the following notation. The numerical approximation is denoted by

$$t_j^n = n \Delta t_j, \quad x_{j,k}^n \approx x_{j,k}(t_j^n), \quad w_{j,k}^n \approx w_{j,k}(t_j^n).$$

Furthermore, let S be a linear interpolatory subdivision of even order q , i.e.

$$(Sx_j)_{2k+1} = \sum_{i=1}^q a_i x_{k-q/2+i}, \quad (Sx_j)_{2k} = x_{j,k}, \quad (5.6)$$

where $a_i, i = 1, \dots, q$ are coefficients¹. Since we are interested in computing the solution up to time T , we also define the integer M_j such that $T = M_j \Delta t_j$. Let I_j be the index set $0, \dots, 2^j$ and \bar{I}_j the index set $1, \dots, 2^j - 1$. Then $x_{j,k}$ is defined for all $k \in I_j$, and $w_{j,k}$ for all $k \in \bar{I}_j$. These values are related via the reconstruction

$$\begin{aligned} x_{j+1,2k+1}^n &= (Sx_j^{2n})_{2k+1} + w_{j+1,2k+1}^n, & 2k+1 \in \bar{I}_j, \\ x_{j+1,2k}^n &= x_{j,k}^{2n}, & k \in I_j. \end{aligned} \quad (5.7)$$

Note that the time levels of the points \mathbf{x}_j are $2n$ since the time step doubling technique is used and the time step at the level j is twice the time step at the level $j+1$.

¹See masks for subdivision schemes in the previous section.

The forward Euler method is as follows. On the zeroth level, for $t_0^{n+1} \leq T$ and $k \in \bar{I}_1$

$$x_{0,k}^{n+1} = x_{0,k}^n + \Delta t_0 F(t_0^n, x_{0,k}^n)$$

where $n = 0, 1, \dots, M_0 - 1$ and $t_0^{M_0} = T$.

At level $j > 0$, for $t_j^{n+1} \leq T$ and $2k + 1 \in \bar{I}_j$

$$w_{j,2k+1}^{n+1} = w_{j,2k+1}^n + \Delta t_j G(t_j^n, \{x_{j-1,l}^{2n}\}_{l=k-q/2+1}^{k+q/2}, w_{j,2k+1}^n) \quad (5.8)$$

where $n = 0, 1, \dots, M_j - 1$ and $t_j^{M_j} = T$, c.f. (5.4). Note that q surrounding points $x_{j-1,k-q/2+1}^{2n}, \dots, x_{j-1,k+q/2}^{2n}$ are needed to calculate $(Sx_{j-1})_{2k+1}$ in (5.3). Instead of the forward Euler scheme, one can use higher order Runge-Kutta schemes. Note that the order of the subdivision scheme has to be higher than the order of the ODE scheme for accuracy reasons, see below.

Let us now give the cost and the approximation error estimate for the methods above. The cost at each level is simply the number of unknowns multiplied by the number of time steps, where the number of unknowns is 2^j and the number of time steps is $\lfloor T/\Delta t_j \rfloor$. The total propagation cost is the sum of the costs on every level, i.e.

$$\sum_{j=0}^J 2^j \left\lfloor \frac{T}{\Delta t_j} \right\rfloor \sim T \sum_{j=0}^J \frac{2^j}{2^j \Delta t} \sim O\left(\frac{\log N}{\Delta t}\right). \quad (5.9)$$

The cost of the method is then the sum of the total propagation cost and the cost for reconstruction of the interface at the final time, which is $O(N)$. This gives the cost $O(\log N/\Delta t + N)$. Hence, this is significantly improved as compared to the standard front tracking methods and the narrow band level set methods, where the cost is $O(N/\Delta t)$, assuming N marker points.

Let us now consider the approximation error. Here we will use the result (5.5) from [62]. Let $\tau_{j,k}^n$ be the local truncation error in time step n for wavelet coefficient k at level j , which we here simply assume is given by the $p + 1$ order derivative of the exact solution for a p -th order method, with $p = 1$ for the forward Euler. Then, if $N = 2^J$ and assuming stability, the global error ε_J is formally estimated as

$$\begin{aligned} \varepsilon_J &\leq \sum_{j=0}^J \sum_{n=0}^{\lfloor T/\Delta t_j \rfloor} \max_{k \in \bar{I}_j} |\tau_{j,k}^n| \leq \sum_{j=0}^J \sum_{n=0}^{\lfloor T/\Delta t_j \rfloor} \max_{k \in \bar{I}_j} \left| \Delta t_j^{p+1} \frac{d^{p+1} w_{j,k}}{dt^{p+1}} \right| \\ &\stackrel{(5.5)}{\leq} CT \Delta t^p \sum_{j=0}^J 2^{(p-q)j}. \end{aligned} \quad (5.10)$$

The accuracy is then $O(\Delta t^p)$ when $p < q$. This is proved rigorously in [62]. Moreover, when $p > q$, we have

$$\varepsilon_J \sim \Delta t^p 2^{(p-q)J} = \Delta t^p N^{p-q}.$$

Assuming $N \sim \Delta t^{-1}$, one then obtains

$$\varepsilon_J \sim \Delta t^q. \quad (5.11)$$

Hence, from (5.10) and (5.11) it follows that the accuracy of the Basic method is $O(\Delta t^{\min(p,q)})$ assuming that the number of points on the interface $N \sim \Delta t^{-1}$. Hence, the error can be bounded independently of N . For instance, we can take $p = 1$ for forward Euler with the midpoint subdivision scheme ($q = 2$).

In the case of the fourth-order Runge-Kutta method ($p = 4$), we should choose at least the 6-point subdivision scheme ($q = 6$) in order to get an error of order four, that can be bounded independently of the number of points N .

A drawback of this method is that it assumes that the wavelet coefficients change approximately uniformly in time within a level, and thus uses the same time step for all wavelet vectors within a level. This is, for instance, not suitable for problems in which the interface has corners, since in that case the wavelet vectors decay more slowly than 2^{-qj} in a neighborhood of the corner. It is also not suitable if the velocity field F varies strongly along the interface. Another problem, similar to standard front tracking, occurs if the interface changes its length significantly, which often happens in practice. Then it cannot be accurately resolved. Hence, more refined methods have to be used for such problems.

5.4 Adaptive Multiresolution Front Tracking

In this thesis, we suggest numerical methods for interface tracking that remedy some of the disadvantages of the basic multiresolution front tracking methods described above. They keep the same description of the interface but are more robust than the Basic method in the sense that they can handle the cases when wavelet vectors on the same level change at very different rates, which happens for instance when the interface has corners. We also introduce space adaptivity, to deal with expanding interfaces.

Time Adaptivity

For problems in which the wavelet vectors do not change uniformly within a level, we suggest a time adaptive method. We solve equations (5.3) using a time adaptive forward Euler method or a time adaptive Runge-Kutta method. To compute the predicted points, we use the two-point subdivision scheme in the first case, and the six-point scheme in the latter case. The time step is automatically chosen based on the size of the local truncation error. We thus use the same adaptive solver on each level, and let it detect the appropriate time step in this way, and not based on the asymptotic decay in (5.5). The method we propose, thus falls into the general category of multirate or multiadaptive ODE methods. See, for instance [51, 33, 18, 37]. Our method is tailored to the wavelet ODE system. It is a proof of concept that shows the potential of using this kind of adaptive methods. More

elaborate multirate and multiadaptive methods for this problem can doubtless be developed.

When all wavelet vectors decay at the same speed, then the adaptive method behaves essentially as the Basic time step doubling method. In problems where wavelet vectors change at different rates within a level, when the front has corners, for instance, then the time doubling technique would either give the wrong results or become too expensive, since the time step has to be small near the corners, and a very small reference time step has to be chosen. Our method remedies this problem by adjusting the time step locally for each wavelet vector or edge point according to the change of the velocity field. Hence, the method is suitable both for problems where the interface has corners, and for problems where the velocity field varies strongly along the interface. See Figures 5.3 - 5.4 for a comparison between our method and the Basic method in a problem where the interface has corners. The interface shown in Figure 5.3 (left) is propagated until time $T = 1$ using the Basic method and the time adaptive method introduced in this thesis. The error and the computational cost of both methods are shown in Figure 5.4. It can be observed that the computational cost of our method is much lower than the computational cost of the Basic method. Moreover, the error of the Basic method grows fast with the number of points N , while the error of our method remains bounded independently of N .

A major additional difficulty in our method is that a recursive interpolation has to be done due to the variable time step. Let us explain this on a simple example. Assume the 2-point subdivision scheme. Assume also that we have computed all wavelet vectors $w_{j,k}$ and the corresponding points $x_{j,k}$. Let them be given at time levels $t_{j,k}$. Here, $t_{j,k}$, $w_{j,k}$ and $x_{j,k}$ are vectors, eg. $t_{j,k} = \{t_{j,k}^n\}_{n \geq 0}$ and $t_{j,k}^n$ for a fixed n is called a time level. Note that $w_{j,k}$ and $x_{j,k}$ have the same time levels $t_{j,k}$, but time levels for different j or k differ. Assume that we want to compute $w_{j+1,2k+1}(\Delta t)$ for a fixed k . Then we need to solve (5.3) in $[0, \Delta t]$. No matter which ODE solver we use, we need points $x_{j,k}(\Delta t)$ and $x_{j,k+1}(\Delta t)$. Since the time step is chosen locally, it usually happens that approximations of $x_{j,k}(\Delta t)$ and $x_{j,k+1}(\Delta t)$ are not available from previous computations, see Figure 5.2. Hence, we have to interpolate. The problem is that we cannot interpolate $x_{j,k}(t_{j,k})$ and $x_{j,k+1}(t_{j,k+1})$ directly, since that would decrease the accuracy of our method. Note that the spacing of time levels is chosen based on the rate of change of the wavelet vectors $w_{j,k}(t)$; the point values $x_{j,k}(t)$ change much faster. Thus, we interpolate the corresponding wavelet vectors $w_{j,k}(t_{j,k})$ and $w_{j,k+1}(t_{j,k+1})$, and then compute $x_{j,k}(\Delta t)$ and $x_{j,k+1}(\Delta t)$ by (5.2). For example, we compute $x_{j,k}(\Delta t)$ by

$$x_{j,k}(\Delta t) = w_{j,k}(\Delta t) + \frac{(x_{j-1, \frac{k-1}{2}}(\Delta t) + x_{j-1, \frac{k-1}{2}+1}(\Delta t))}{2}. \quad (5.12)$$

However, since we cannot interpolate $x_{j-1,k}(t_{j-1,k})$ either, we have to repeat this process recursively. We thus need to interpolate $w_{j,k}(t_{j,k})$ all the way to $j = 1$. Then, $x_{0,0}$ and $x_{0,1}$ appear in (5.12) and these we can interpolate accurately, since they are computed directly by the adaptive method. The interpolation may thus,

at worse, involve J steps if J is the finest level. This is still just an $O(\log N)$ cost, however, and does not significantly alter the previous complexity estimates. For details, see the second paper in this thesis.

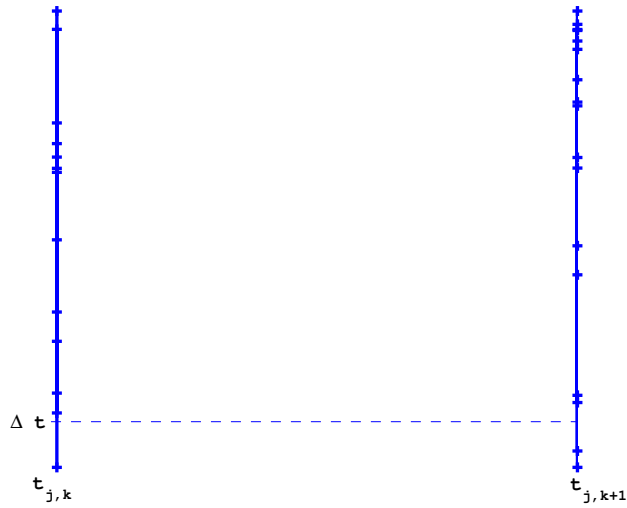


Figure 5.2: Time levels $t_{j,k}$ and $t_{j,k+1}$ at which $x_{j,k}(w_{j,k})$ and $x_{j,k+1}(w_{j,k+1})$ are given, respectively. If $x_{j,k}(\Delta t)$ or $x_{j,k+1}(\Delta t)$ are needed, we have to interpolate.

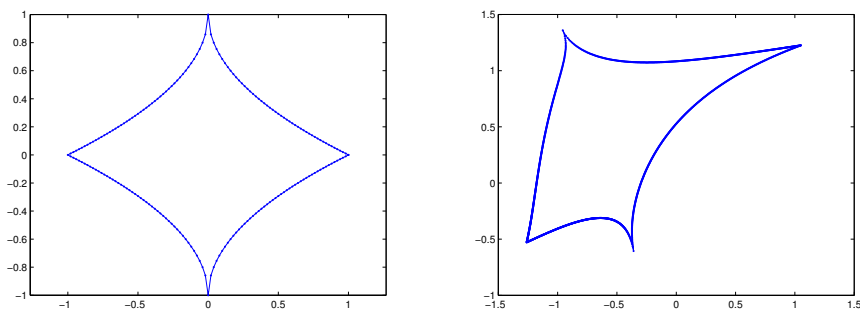


Figure 5.3: The initial interface (left) and the interface at $T = 1$ (right).

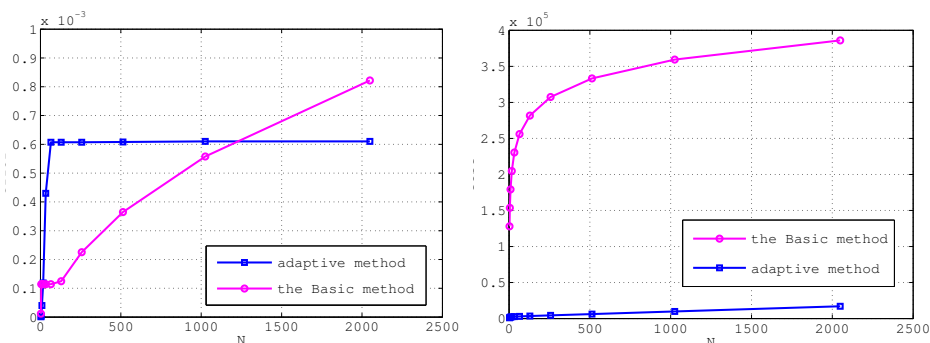


Figure 5.4: Error (left) and cost (right) as a function of number of points on the interface N . The reference time step in the Basic method is $2 \cdot 10^{-5}$. The initial curve is shown in Figure 5.3 (left).

Spatial Adaptivity

For problems in which the length of the interface changes with time, we suggest a space and time adaptive method. The time adaptivity is done in the same way as above. The goal of the space adaptivity is the same as in standard front tracking: adding points adaptively to maintain a good resolution of the front. In the multiresolution setting, this is however more difficult. The space adaptive method that we propose uses the length of the wavelet vectors as a sensor. More precisely, if the size of a wavelet vector is larger than some given tolerance Tol , new points/wavelet vectors are added. Unfortunately, this is not as straightforward as it sounds. Let us explain the way space adaptivity is done by a simple example. Keep in mind that we also have time adaptivity, and thus use recursive interpolation, described above. Moreover, two different wavelet vectors are basically never given on the same time levels. Assume the situation of Figure 5.5. Time levels are represented by dots, and the specific time level at which the size of the wavelet vector becomes larger than a given tolerance Tol is represented by a square and denoted by $t_{j,k}^*$. If the sizes of all wavelet vectors $w_{j,k}^n$ are less than Tol , we define $t_{j,k}^* := -1$. Assume, as before, the two point subdivision scheme and that we have computed all wavelet vectors up to level j . Then, for every pair k and $k+1$ at level j , we check if $t_{j,k}^*$ or $t_{j,k+1}^*$ is non-negative. If that is the case, we add a new wavelet vector $w_{j+1,2k+1}$ at $t = \max(t_{j,k}^*, t_{j,k+1}^*)$ and compute it for $t \in [\max(t_{j,k}^*, t_{j,k+1}^*), T]$, where T is the final time.

As an illustration, in Figure 5.6 we show interfaces at the final time $T = 4$ obtained by the standard space-time adaptive method, and our space-time adaptive multiresolution based method when the velocity field is given by (4.3).

In the standard space-time adaptive method, points are propagating instead of wavelet vectors as in our method. Solutions obtained by both methods are good,

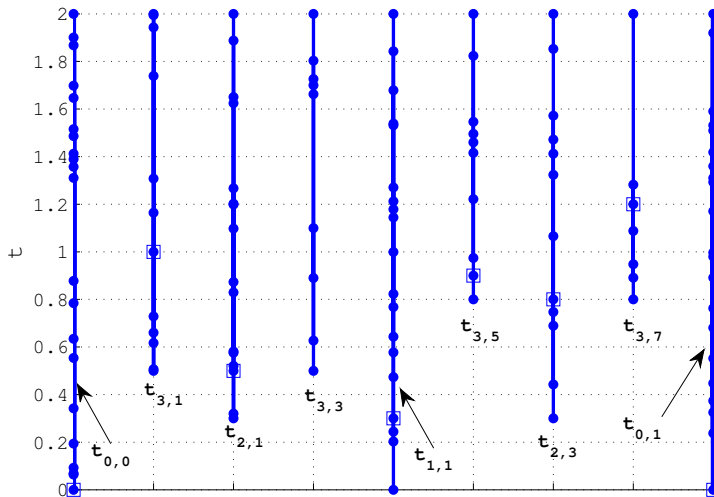


Figure 5.5: Time levels $t_{j,k}$. Dots represent the time levels and squares represent a time level at which the size of the wavelet vector $w_{j,k}$ becomes larger than a given tolerance.

but the computational cost of our method is more than ten times lower than the cost of the standard method. More precisely, the total number of time steps in the standard method is 797775, while the total number of time steps in our method is 77289. However, both methods give better results than the standard method without adaptivity, c.f. Figure 4.1.

At least for smooth problems, these methods have $O(\text{Tol} \log N)$ error, where Tol is some given tolerance and N is the number of points on the interface. The error can be reduced to $O(\text{Tol})$ if the tolerance is halved with every level, i.e. if the tolerance at level j is $\text{Tol}/2^j$. The complexity is as low as the complexity of the method described in [65], i.e. it is $O(\log N/\text{Tol}^{1/p} + N)$. The $O(\log N/\text{Tol}^{1/p})$ is the cost to propagate the interface with an adaptive ODE method of order p . The $O(N)$ part of the complexity is the cost to reconstruct the interface from the wavelet vectors.

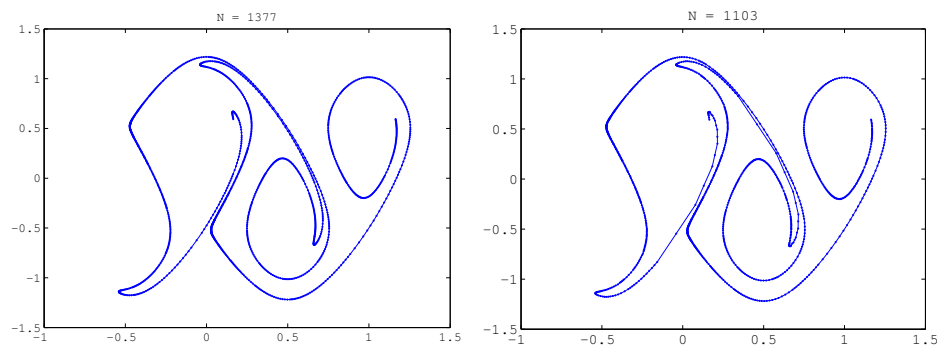


Figure 5.6: The interface obtained by the standard space-time adaptive method (left), and the interface obtained by our space-time adaptive method (right).

Chapter 6

Time Upscaling of Hamilton-Jacobi Equations

6.1 Hamilton-Jacobi Equations

Hamilton-Jacobi (HJ) equations arise in many applications such as geometrical optics, mechanics, seismology and image processing. Hence, the development of efficient numerical methods for HJ equations is very important. In this thesis, we consider the HJ initial value problem of the form

$$\phi_t + H(\mathbf{x}, \nabla\phi) = 0, \quad \mathbf{x} \in \mathbb{R}^n, \quad t \in (0, T] \quad (6.1)$$

$$\phi(0, \mathbf{x}) = \phi_0(\mathbf{x}), \quad (6.2)$$

where ϕ_0 is a known function. The function $H(\mathbf{x}, \nabla\phi)$ is called the Hamiltonian function. We assume that it is convex in its second argument. Solutions of these equations are usually not unique, and do not satisfy the equations in the classical sense. The class of unique solutions of the Hamilton-Jacobi equations, called viscosity solutions, was established by Crandall and Lions [19].

The problem (6.1) can be reformulated as a front tracking problem by introducing the bicharacteristics $\mathbf{x}(t, \mathbf{s})$ and $\mathbf{p}(t, \mathbf{s})$ for the Hamiltonian $H(\mathbf{x}, \mathbf{p})$, which results in the system of ODEs

$$\begin{aligned} \frac{\partial \mathbf{x}}{\partial t} &= H_{\mathbf{p}}(\mathbf{x}, \mathbf{p}), \\ \frac{\partial \mathbf{p}}{\partial t} &= -H_{\mathbf{x}}(\mathbf{x}, \mathbf{p}). \end{aligned} \quad (6.3)$$

with initial conditions

$$\mathbf{x}(0, \mathbf{s}) = \mathbf{s}, \quad \mathbf{p}(0, \mathbf{s}) = \nabla\phi_0(\mathbf{s}).$$

Moreover, we define the *phase space solution* $\varphi(t, \mathbf{s})$ by

$$\frac{\partial \varphi}{\partial t} = H_{\mathbf{p}}(\mathbf{x}, \mathbf{p})^T \mathbf{p} - H(\mathbf{x}, \mathbf{p}), \quad \varphi(0, \mathbf{s}) = \phi_0(\mathbf{s}). \quad (6.4)$$

Then, $\phi(t, \mathbf{x}(t, \mathbf{s})) = \varphi(t, \mathbf{s})$ as long as the solution $\phi(t, \mathbf{x})$ to (6.1) is smooth. For non-smooth solutions $\phi(t, \mathbf{s})$ the equivalence between that solution and the phase space solution is not valid, since the solution becomes multivalued; in general there are several values \mathbf{s}_j such that $\mathbf{x}(t, \mathbf{s}_1) = \mathbf{x}(t, \mathbf{s}_2)$. For convex Hamiltonians, the correct value of ϕ is then obtained by taking the minimum value of the multivalued phase solution φ [11], i.e.

$$\phi(t, x) = \min_{\mathbf{x}(t, \mathbf{s})=x} \varphi(t, \mathbf{s}). \quad (6.5)$$

6.2 Numerical methods

Numerical methods for these problems have already been introduced in the previous chapters. For (6.1) they include schemes of ENO and WENO type [53, 54, 42, 81, 50]. For problems in which (6.1) is solved by the method of characteristics, numerical methods include ray tracing methods [15, 43, 47, 72] and wave front methods [77]. The method that we propose in this thesis is based on the fact that the method of characteristics for HJ equations can be reformulated as the front tracking problem. In fact, we can consider $(\mathbf{x}(t, \mathbf{s}), \mathbf{p}(t, \mathbf{s}), \varphi(t, \mathbf{s}))$ as a front parametrized by $\mathbf{s} \in \mathbb{R}^n$ propagating in \mathbb{R}^{2n+1} . That problem is then solved by fast interface tracking methods proposed in [58, 59, 65]. This provides the multivalued solution $\mathbf{x}(t, \mathbf{s}), \phi(t, \mathbf{s})$ in a fast way. The viscosity solution is then calculated in a post-processing step by taking the minimum value of the multivalued solution, as in (6.5). Hence, our method has the following steps:

1. First, reformulate (6.1) as a front tracking problem and compute the multivalued solution using fast interface tracking.
2. Second, reconstruct the viscosity solution from the multivalued solution with an algorithm that has cost $O(\Delta x^{-d})$ in d dimensions, where $\Delta x \sim 1/N$ and N is the number of discretization points in every spatial direction.

If we use the time step doubling technique proposed in [62] for the fast interface tracking, the computational cost of the first step in the algorithm is $O((\Delta t)^{-d})$ or $O((\Delta t)^{-d} |\log \Delta t|)$ in d dimensions, where Δt is the reference time step (see Section 5.3 in the previous chapter). Thus, the total computational cost of the algorithm is $O(\Delta t^{-d})$ or $O(\Delta t^{-d} |\log(\Delta t)|)$ if $\Delta t \sim \Delta x$, where Δt and Δx are as above. That is a significant improvement compared to the $O(\Delta t^{-(d+1)})$ cost of standard finite-difference methods. One way to see this is that standard explicit methods are constrained by the CFL condition, so that $\Delta t \sim \Delta x$. We want to point out here that it is in general possible to bypass the CFL condition, and thus decrease the computational cost significantly compared to standard numerical methods that are inhibited by that requirement. Numerical methods that have the possibility to go around the CFL condition without reducing accuracy are called *time upscaling* methods. They will be described in the following section.

6.3 Time Upscaling Methods

Time upscaling methods have been proposed to reduce the computational complexity of direct simulation of time dependent PDEs. The aim is to reduce the extra cost incurred by the time-stepping.

Let us consider a d -dimensional problem, and assume N discretization points in every spatial direction and M discretization points in time. For explicit methods, due to the stability (CFL) and accuracy requirements, it is necessary to have $M \sim N^r$ where $r = 1$ for hyperbolic problems and $r = 2$ for parabolic problems. Then, the computational cost for a time interval of $\mathcal{O}(1)$ is $\mathcal{O}(N^{d+r})$. With time upscaling, the computational cost can be reduced to $\mathcal{O}(N^d \log N)$ or even to $\mathcal{O}(N^d)$ while maintaining the same accuracy.

For the advection and parabolic equations with spatially varying coefficients, time upscaling methods are proposed in [26]. The authors suggest using the fast wavelet transform together with truncation. Consider the following evolution equation

$$\begin{aligned} \partial_t u + L(x, \partial_x)u &= 0, & x \in \Omega \subset \mathbb{R}^d, & \quad t > 0, \\ u(x, 0) &= u_0(x), \end{aligned}$$

where L is a differential operator. After discretization, and collecting the unknowns in a vector $u \in \mathbb{R}^{N^d}$, the equation becomes

$$\begin{aligned} u^{n+1} &= Au^n, \\ u^0 &= u_0, \end{aligned} \tag{6.6}$$

where A is a $N^d \times N^d$ matrix approximating the operator $\partial_t + L$. The computational complexity of computing u^n is of order $\mathcal{O}(N^{d+r})$. Clearly, (6.6) is equivalent to

$$u^n = A^n u_0. \tag{6.7}$$

Then, repeated squaring of A can be used to compute the solution of (6.7) in $\log_2 M$ steps for $M = 2^m$, where m is an integer, i.e. one can compute $A, A^2, A^4, \dots, A^{2^m}$ in $m = \log_2 M$ steps. Since the matrix A is sparse, the cost of squaring is $\mathcal{O}(N^d)$. However, the later squarings involve almost dense matrices, and the computational cost is then $\mathcal{O}(N^{3d})$. The total cost becomes $\mathcal{O}(N^{3d} \log M)$, which is more expensive than to solve (6.6) directly. Instead, Engquist, Osher and Zhong in [26] suggest a wavelet representation of A which can decrease the computational complexity of the repeated squaring algorithm. More precisely, the solution of (6.7) can be computed in $m = \log_2 M$ steps in the following way

$$\begin{aligned} B &:= SAS^{-1}, \\ B &:= \text{TRUNC}(B^2, \epsilon) \quad (\text{iterate } m \text{ steps}) \\ u^n &:= S^{-1}BSu^0. \end{aligned} \tag{6.8}$$

The matrix S corresponds to a fast wavelet transform, and the truncation operator TRUNC sets all elements of A that are less than ϵ to 0. It is obvious that the algorithm (6.8) is equivalent to (6.7) for $\epsilon = 0$, and it was shown in [26] that there exists a small enough ϵ such that the result of the algorithm (6.8) is close to (6.7). Through truncation, the algebraic problem with dense matrices is transformed to a problem with sparse matrices. The authors also showed that the cost to compute the 1D hyperbolic equation can be reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(N(\log N)^3)$ for a fixed accuracy. Furthermore, for d -dimensional parabolic problems the computational complexity can be reduced from $\mathcal{O}(N^{d+2})$ to $\mathcal{O}(N^d(\log N)^3)$.

A similar technique is proposed in [23] by Demanet and Ying. They consider the 2D wave equation

$$\begin{aligned} u_{tt} - c^2(x)\Delta u &= 0, & x \in [0, 1]^2 \\ u(x, 0) &= u_0, & u_t(x, 0) = u_1, \end{aligned}$$

with periodic boundary conditions. They assume $c(x) \in C^\infty$ to be positive and bounded away from zero and propose to transform the wave equation into a first order system of equations,

$$v_t = Av, \quad v(t=0) = v_0,$$

where $v = (u_t, u_x)$ and A is a 2-by-2 matrix of operators. The system is then solved up to time $t = T$ using wave atoms, which provide a sparse representation of the matrix, combined with repeated squaring and truncation, as above. They proved rigorously that the computational cost of upscaled timestepping on a N by N grid is between $\mathcal{O}(N^{2.25} \log N)$ and $\mathcal{O}(N^3 \log N)$, depending on the initial data and the structure of $c(x)$, which is lower than $\mathcal{O}(N^3 \log N)$ computational cost of pseudo-spectral methods. The key of the proof is to show that the solution operator e^{At} remains almost sparse for all t in the wave atom basis.

Another fast time upscaling method for the one-dimensional wave equation is proposed by Stolk [68]. The idea here is first to rewrite the wave equation as a system of one-way wave equations, then transform them into wavelet bases and solve these using multiscale stepping, which means that the fine spatial scales are solved with longer time steps, and the coarse spatial scales are solved with the shorter time steps. The computational cost is only $\mathcal{O}(N)$ for fixed accuracy.

Fast interface tracking methods that we propose in this thesis are similar in style to the method proposed by Stolk [68]. Hence, our methods can be considered as time upscaling for interface tracking. Moreover, the method proposed in this chapter for HJ equations can be regarded as a time upscaling method for HJ equations.

Remark 3. *There is an interesting connection between time upscaling methods and methods with weakly frequency dependent cost introduced in Chapter 3. When the one-dimensional wave equation is solved with the time upscaling methods with repeated squaring, the solution is given on a grid that is coarse in time and dense in spatial direction, see Figure 6.1. One can also solve the wave equation by solving*

Helmholtz equations for each frequency component. This is like taking a Fourier transform in time. A spatial discretization with N points then corresponds to N Helmholtz equations for N frequency components. If the Helmholtz equation is solved with a frequency independent method, then the total cost is $\mathcal{O}(N)$, but the solution is obtained only in $\mathcal{O}(1)$ spatial points. After an inverse FFT for each point at a cost of $\mathcal{O}(N \log N)$, the solution of the wave equation is obtained on a dense grid in time but coarse in space at a total cost $\mathcal{O}(N \log N)$. Hence, by using methods with frequency-independent cost, one can obtain the solution on a coarse grid in space, but fine in time, with the same cost as in time upscaling, see Figure 6.1.

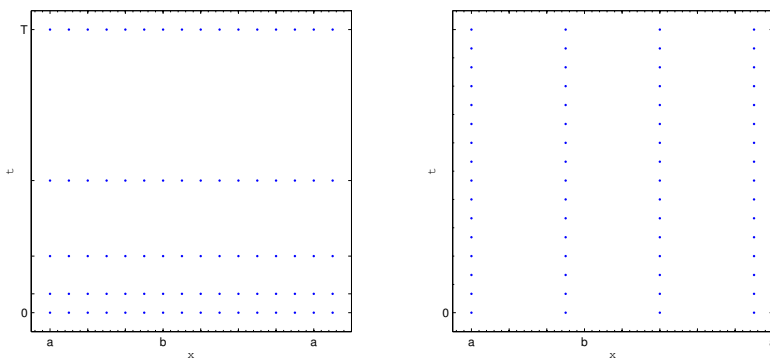


Figure 6.1: In time upscaling methods, the solution is given on a coarse grid in time but dense in space (left). Using frequency-independent numerical methods, the solution could be obtained on a grid that is coarse in space, but dense in time (right).

6.4 Post-Processing in Time Upscaling of HJ Equations

Let us now describe the post-processing step of our fast method for HJ equations. For simplicity, we consider the one-dimensional case. Assume thus that we have a multivalued solution $x(T, s)$, $\varphi(T, s)$ at a fixed time T . We want to compute the viscosity solution $\phi(T, \xi)$. Before we describe the method, let us introduce the notation. The numerical approximation is denoted by

$$s_j = j\Delta s, \quad x_j(t) \approx x(t, s_j), \quad \varphi_j(t) \approx \varphi(t, s_j)$$

and

$$\xi_i = i\Delta \xi, \quad \phi_i(t) = \phi(t, \xi_i).$$

Then, our method of computing the single-valued solution $\phi(T, x)$ from the multi-valued solution $\varphi(T, s)$ is as follows. Assume that we have x_j and φ_j for $s_j = j\Delta s$, $j = 0, \dots, J$, and that we want to get the solution ϕ_i on a regular grid $\xi_i = i\Delta\xi$, $i = 0, \dots, I$. To do this we can loop through the s_j indices and record the value of φ_j for ϕ_i , where ξ_i is the point closest to x_j , but only if the value of φ_j is smaller than the value already assigned to ϕ_i . Note that since $\xi_i = i\Delta\xi$ is a regular grid, finding the point ξ_i closest to x_j can be done quickly by $i := \text{round}(x(t, s_j)/\Delta\xi)$, $\xi_i = i\Delta\xi$. Hence, the cost for this step is $O((\Delta x)^{-1})$. The post-processing algorithm for 1D is described by Algorithm 1.

Algorithm 1 Post Processing Algorithm for Examples in 1D

```

Choose  $I$  and calculate  $\Delta\xi = \frac{x_J - x_0}{I}$ 
Define  $\xi_i = x_0 + i\Delta\xi$ ,  $i = 0, \dots, I$ 
Define  $\phi_i$ ,  $i = 0, \dots, I$  so that  $\phi_i > \max_j(\varphi_j)$ ,  $\forall i$ 
for  $j = 1$  to  $J$  do
   $i = \text{round}(x_j/\Delta\xi)$ 
  if  $\varphi_j < \phi_i$  then
     $\phi_i = \varphi_j$ 
  end if
end for
return  $\{\phi_i\}$ 

```

The solution obtained by Algorithm 1 is first order accurate. The order of approximation can be improved by first interpolating the multivalued solution at the desired points, and then taking the minimum. Details of that algorithm are given in the last paper of this thesis. Description of the algorithm in two dimension is also given in the last paper of the thesis.

Let us now show one numerical example. Let the Hamiltonian $H(x, \nabla\phi)$ and the initial function $\phi_0(x)$ be given by

$$H(x, \phi_x) = \frac{\phi_x^2}{2}, \quad \phi_0(x) = -\cos(3\pi x), \quad (6.9)$$

for $x \in [-1, 1]$. The solutions of (6.1) at $T = 0.1$, obtained by the Lax-Friedrichs method and our method are shown in Figure 6.2 (top). The multivalued solution $(x(T, s), \varphi(T, s))$ at $T = 0.1$ and the reconstructed viscosity solution $\phi(\xi)$ are plotted in Figure 6.2 (bottom). It can be observed that the viscosity solution obtained by our method and the viscosity solution obtained by the Lax-Friedrichs scheme agree, i.e. the viscosity solution is well approximated by our method.

Remark 4. A straightforward way to obtain the approximation of the viscosity solution $\phi(T, \xi_i)$ would be to loop through ξ_i , find intervals $[x_j, x_{j+1}]$ such that $\xi_i \in [x_j, x_{j+1}]$, interpolate $\varphi(T, x_j)$ at points ξ_i and assign the minimum of the interpolated values for fixed i to $\phi(T, \xi_i)$. The cost of such a method would, however, be of order $O(\Delta x^{-2})$, which would increase the total cost of our method.

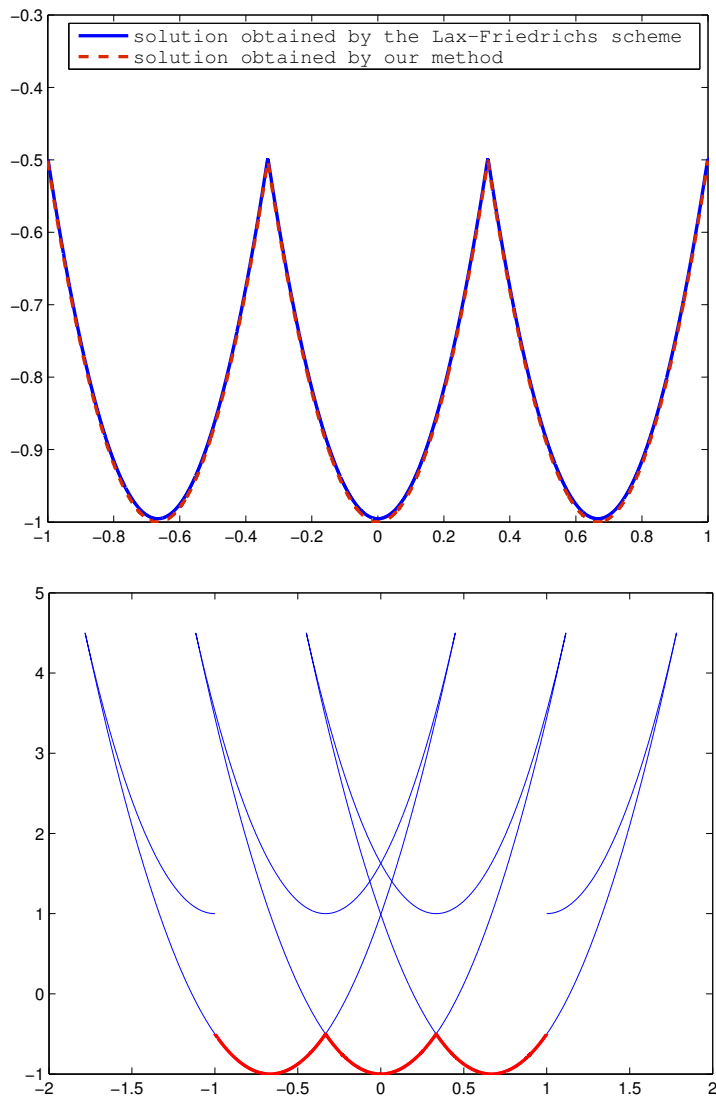


Figure 6.2: Top: The viscosity solution $\phi(x)$ at $T = 0.1$ obtained by the Lax-Friedrichs method (blue) and the viscosity solution $\phi(\xi)$ obtained by our method (red). Bottom: The multivalued solution $(x(T, s), \varphi(T, s))$ (blue) and the reconstructed viscosity solution $\phi(\xi)$ (red). The solution of our method agrees with the solution obtained by the Lax-Friedrichs method. The Hamiltonian $H(x, \phi_x)$ and $\phi_0(x)$ are given by (6.9).

Chapter 7

Summary of Papers

7.1 Paper I: Analysis of a Fast Method for Solving the High Frequency Helmholtz Equation in One Dimension

In this paper we consider the one dimensional high frequency Helmholtz equation with a variable wave speed function. We propose and analyze a fast numerical method for this problem. The method is based on wave splitting. More precisely, the Helmholtz equation is split into two one-way wave equations with source functions that are then solved iteratively for a given tolerance. We show rigorously in one dimension that the algorithm is convergent and that for fixed accuracy, the computational cost depends weakly on the frequency.

This paper is published in BIT Numerical Mathematics.

7.2 Adaptive Fast Interface Tracking Methods, Part I: Time Adaptivity

In this paper, we derive a time adaptive method for interface tracking that is based on a multiresolution description of the interface. The method is an extension of the method presented in [65], where the time step doubling technique is used for time evolution of the interface. The method that we suggest in this paper can be used for larger class of problems than the method proposed in [65] and is thus more robust. Time adaptivity allows for individual choice of time steps for each point on the interface, so that larger time steps can be used without loss of accuracy. Moreover, the computational cost is low. A disadvantage of the method is that it cannot handle problems where the length of the interface changes rapidly in time.

7.3 Adaptive Fast Interface Tracking Methods, Part II: Spatial Adaptivity

In this paper, we propose a space-time adaptive method for interface tracking that is based on a multiresolution description of the interface. The method is an extension of the time adaptive interface tracking method introduced in the previous paper so that it can also handle problems with expanding interfaces. The method has low computational cost and is suitable for a large class of problems.

7.4 Time Upscaling for Hamilton-Jacobi Equations

In this paper, we propose fast and accurate numerical methods for time dependent Hamilton-Jacobi equations with convex Hamiltonians. The method uses the fact that the problem can be solved by the method of characteristics, which can be reformulated as a front tracking problem. We first solve the front tracking problem using the fast methods for interface tracking proposed in [65] and in the second and the third papers of this thesis. In that way a multivalued solution is obtained. Next, we reconstruct the viscosity solution from the multi-valued solution in a fast way. The computational cost is $O(\Delta t^{-d} |\log \Delta t|)$ or $O(\Delta t^{-d})$ in d dimensions, where Δt is the time step. This is a significant improvement compared to the $O(\Delta t^{-(d+1)})$ computational cost of standard numerical methods for such problems.

Bibliography

- [1] T. Abboud, J.-C. Nédélec, and B. Zhou. Méthode des équations intégrales pour les hautes fréquences. *C.R. Acad. Sci. Paris.*, 318:165–170, 1994.
- [2] D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *J. Comp. Phys.*, 118(2):269–277, 1995.
- [3] S. Ahmed, S. Bak, J. McLaughlin, and Daniel Renzi. A third order accurate fast marching method for the eikonal equation in two dimensions. *SIAM J. Sci. Comput.*, 33(5):2402–2420, 2011.
- [4] V. M. Babic and T. F. Pankratova. On discontinuities of Green’s function of the wave equation with variable coefficient. *Problemy Matem. Fiziki., Leningrad University, Saint-Petersburg*, 6, 1973.
- [5] J.-D. Benamou. Big ray tracing: Multivalued travel time field computation using viscosity solutions of the eikonal equation. *J. Comput. Phys.*, 128(4): 463–474, 1996.
- [6] J.-D. Benamou. Direct computation of multi valued phase-space solutions for Hamilton-Jacobi equations. *Commun. Pure Appl. Math.*, 52(11):1443–1475, 1999.
- [7] Y. Brenier and L. Corrias. A kinetic formulation for multibranch entropy solutions of scalar conservations laws. *Ann. Inst. Henri Poincaré*, 15(2):169–190, 1998.
- [8] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 2008.
- [9] O. P. Bruno and C. A. Geuzaine. An $O(1)$ integration scheme for three-dimensional surface scattering problems. *J. Comput. Appl. Math.*, 204:463–476, 2007.
- [10] O. P. Bruno, C. A. Geuzaine, J. A. Monro Jr, and F. Reitich. Prescribed error tolerances within fixed computational times for scattering problems of arbitrarily high frequency: the convex case. *Philos. Transact. A. Math. Phys. Eng. Sci.*, 362(1816):629–645, 2004.

- [11] Piermarco Cannarsa and Carlo Sinestrari. *Semiconcave functions, Hamilton-Jacobi equations, and optimal control*. Progress in Nonlinear Differential Equations and their Applications, 58. Birkhäuser Boston Inc., Boston, MA, 2004. ISBN 0-8176-4084-3.
- [12] A. S. Cavaretta, W. Dahmen, and C. A. Micchelli. Stationary subdivision. *Memoirs Amer. Math. Soc.*, 93(493), 1991.
- [13] O. Cessenat and B. Despres. Application of an ultra weak variational formulation of elliptic PDEs to the two-dimensional Helmholtz problem. *SIAM J. Numer. Anal.*, 35(1):255–299, 1998.
- [14] S. N. Chandler-Wilde, I. G. Graham, S. Langdon, and E. A. Spence. Numerical-asymptotic boundary integral methods in high-frequency acoustic scattering. *Acta Numerica*, 21:89–305, 2012.
- [15] V. Červený, I. A. Molotkov, and I. Psencik. Ray methods in seismology. *Univ. Karlova Press*, 1977.
- [16] D. L. Chopp. Computing minimal surfaces via level set curvature flow. *Jour. of Comp. Phys.*, 106:77–91, 1993.
- [17] D. Colton and R. Kress. *Inverse Acoustic and Electromagnetic Scattering Theory*, volume 93. Applied Mathematical Sciences, Springer-Verlag, Berlin, 1998.
- [18] E. M. Constantinescu and A. Sandu. Multirate timestepping methods for hyperbolic conservation laws. *J. Sci. Comput.*, 22:239–278, 2007.
- [19] M. G. Crandall and P. L. Lions. Two approximations of solutions of Hamilton-Jacobi equations. *Mathematics of Computation*, 43:1–19, 1984.
- [20] I. Daubechies and J. C. Lagarias. Two-scale difference equations I. Existence and global regularity of solutions. *SIAM J. Math. Anal.*, 22(5):1388–1410, 1991.
- [21] I. Daubechies and J. C. Lagarias. Two-scale difference equations II. Local regularity, infinite products of matrices and fractals. *SIAM J. Math. Anal.*, 23(4):1031–1079, 1992.
- [22] I. Daubechies, O. Runborg, and W. Sweldens. Normal multiresolution approximation of curves. *Constr. Approx.*, 20:399–463, 2004.
- [23] L Demanet and L. Ying. Wave atoms and time upscaling of wave equations. To appear in *Numer.Math.*, 2007.
- [24] V. Dominguez, I. G. Graham, and V. P. Smyshlyaev. A hybrid numerical-asymptotic boundary integral method for high-frequency acoustic scattering. *Numer. Math.*, 106:471–510, 2007.

- [25] N. Dyn, D. Levin, and J. Gregory. A 4-point interpolatory subdivision scheme for curve design. *Comput. Aided. Geom. Des.*, 4:257–268, 1987.
- [26] B. Engquist, S. Osher, and S. Zhong. Fast wavelet based algorithms for linear evolution equations. *SIAM J. Sci. Comput.*, 15(4):755–775, 1994.
- [27] B. Engquist and O. Runborg. Multiphase computations in geometrical optics. *J. Comput. Appl. Math.*, 74:175–192, 1996.
- [28] B. Engquist and O. Runborg. Computational high frequency wave propagation. *Acta Numerica*, 12:181–266, 2003.
- [29] B. Engquist, O. Runborg, and A.-K. Tornberg. High-frequency wave propagation by the segment projection method. *J. Comput. Phys.*, 178:373–390, 2002.
- [30] B. Engquist and L. Ying. Fast directional multilevel algorithms for oscillatory kernels. *SIAM Journal on Scientific Computing*, 29(4):1710–1737, 2007.
- [31] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. *Computational Differential Equations*. Studnetlitteratur, Lund, 1996.
- [32] Y. A. Erlagga. Advances in iterative methods and preconditioners for the helmholtz equation. *Arch. Comput. Methods Eng.*, 15:37–66, 2008.
- [33] C. Gear and D. Wells. Multirate linear multistep methods. *BIT*, 24:484–52, 1984.
- [34] E. Giladi and J. B. Keller. A hybrid numerical asymptotic method for scattering problems. *Comput. Phys.*, 174:226–247, 2001.
- [35] J. Glimm, J. W. Grove, X. L. Li, K.-M. Shyue, Y. Zeng, and Q. Zhang. Three - dimesional front tracking. *SIAM J. Sci. Comput.*, 19(3):703–727, 1998.
- [36] J. Glimm, E. Isaacson, D. Marchesin, and O. McBryan. Front tracking for hyperbolic systems. *Adv. Appl. Math.*, 2:91–119, 1981.
- [37] M. Günther and P. Rentorp. Mutirate ROW methods and latency of electric circiuts. *Appl. Numer. Math.*, 13:83–102, 1993.
- [38] I. Guskov, K. Vidimce, W. Sweldens, and P. Schroder. Normal meshes. *Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 259–268, 2000.
- [39] B. Gustafsson. *High Order Difference Methods for Time Dependent PDE*. Springer-Verlag Berlin Heidelberg, 2008.
- [40] D. Huybrechs and S. Vanderwalle. A sparse discretization for integral equation formulations of high frequency scattering problems. *SIAM J. Sci. Comput.*, 29:2305–2328, 2007.

- [41] D. Huybrechts and S. Vandewalle. A sparse discretisation for integral equation formulations of high-frequency scattering problems. *Comm. SIAM J. Sci. Comput.*, 29:2305–2328, 2007.
- [42] G. S. Jiang and D. Peng. Weighted ENO schemes for Hamilton-Jacobi equations. *SIAM J. Sci. Comput.*, 21:2126–2143, 2000.
- [43] B. R. Julian and D. Gubbins. Three-dimensional seismic ray tracing. *J. Geophys. Res.*, 43:95–114, 1977.
- [44] C. Y. Kao, S. Osher, and J. Qian. Lax-Friedrichs sweeping scheme for static Hamilton-Jacobi equations. *J. Comput. Phys.*, 196(1):367–391, 2004.
- [45] J. Keller. Geometrical theory of diffraction. *J. Opt. Soc. Amer*, 52, 1962.
- [46] S. Kim. An $\mathcal{O}(n)$ level set method for eikonal equations. *SIAM J. Sci. Comput.*, 22(6):2178–2193, 2000.
- [47] R. T. Langan, I. Lerche, and R. T. Cutler. Tracing of rays through heterogeneous media: An accurate and efficient procedure. *Geophysics*, 50:1456–1465, 1985.
- [48] S. Langdon and S. N. Chandler-Wilde. A wave number independent boundary element method for an acoustic scattering problem. *SIAM J. Numer. Anal.*, 43:2450–2477, 2006.
- [49] Randall. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2006.
- [50] C. T. Lin and E. Tadmor. High resolution nonoscillatory central schemes for Hamilton-Jacobi equations. *SIAM J. Sci. Comput.*, 21:2163–2186, 2000.
- [51] A. Logg. Multi-adaptive time integration. *Appl. Num. Math.*, 48:339–354, 2004.
- [52] R. Malladi, J. A. Sethian, and B. C. Vemuri. Evolutionary fronts for topology-independent shape modeling and recovery. In *Third European Conference on Computer Vision, Stockholm, Sweden, Lecture Notes in Computer Science*, volume 800, pages 3–13, 1994.
- [53] S. J. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.
- [54] S. J. Osher and C.-W. Shu. High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations. *SIAM J. Numer. Anal.*, 28(4):907–977, 1991.
- [55] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang. A PDE based fast local level set method. *J. Comput. Phys.*, 2(155):410–438, 1999.

- [56] E. Perrey-Debain, O. Laghrouche, P. Bettess, and J. Trevelyan. Plane-wave basis finite elements and boundary elements for three-dimensional wave scattering. *Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.*, 362(1816): 561–577, 2004.
- [57] M. M. Popov. A new method of computation of wave fields using gaussian beams. *Wave Motion*, 4:85–97, 1982.
- [58] J. Popovic and O. Runborg. Adaptive fast interface tracking methods, part I: Time adaptivity. preprint (2012).
- [59] J. Popovic and O. Runborg. Adaptive fast interface tracking methods, part II: Spatial adaptivity. preprint (2012).
- [60] J. Ralston. Gaussian beams and the propagation of singularities, Studies in Partial Differential Equations. *MAA Studies in Mathematics*, 23:206–248, 1983.
- [61] V. Rokhlin. Sparse diagonal forms for translation operators for the Helmholtz equation in two dimensions. *Appl. Comput. Harmon. Anal.*, 5(1):36–67, 1998.
- [62] O. Runborg. Analysis of high order fast interface tracking methods. preprint (2012).
- [63] O. Runborg. Some new results in multiphase geometrical optics. *M2AN Math. Model. Numer. Anal.*, 34:1203–1231, 2000.
- [64] O. Runborg. Mathematical models and numerical methods for high frequency waves. *Communications in Computational Physics*, 2(5):827–880, 2007.
- [65] O. Runborg. Fast interface tracking via a multiresolution representation of curves and surfaces. *Commun. Math. Sci.*, 7(2):365–398, 2009.
- [66] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proc. Nat. Acad. Sci. U.S.A.*, 93(4), 1966.
- [67] J. A. Sethian. *Level set methods and fast marching methods*. Cambridge University Press, 2006.
- [68] Christiaan C. Stolk. A fast method for linear waves based on geometrical optics. *SIAM J. Numer. Anal.*, 47:1168–1194, 2009.
- [69] J. Strain. Tree methods for moving interfaces. *J. Comput. Phys.*, 2(151): 616–648, 1999.
- [70] J. Strain. A fast modular semi-Lagrangian method for moving interfaces. *J. Comput. Phys.*, 2(161):512–536, 2000.

- [71] W. W. Symes and J. Qian. A slowness matching Eulerian method for multivalued solutions of eikonal equations. *SIAM J. Sci. Comput.*, 19:501–526, 2003.
- [72] C. H. Thurber and W. L. Ellsworth. Rapid solution of ray tracing problems in heterogeneous media. *Bull. Seism. Soc. Amer.*, 70:1147–1148, 1980.
- [73] A.-K. Tornberg and B. Engquist. The segment projection method for interface tracking. *Comm. Pure Appl. Math.*, 56(1):47–79, 2003.
- [74] R. Tsai, R.-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.
- [75] Y. R. Tsai, L. T. Cheng, S. Osher, and H. K. Zhao. Fast sweeping algorithms for a class of Hamilton-Jacobi equations. *SIAM Journal on Numerical Analysis*, 41(2):673–694, 2003.
- [76] J. N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Trans. Automat. Control*, 40(9):1528–1538, 1995.
- [77] V. Vinje, E. Iversen, and H. Gjøystdal. Traveltime and amplitude estimation using wavefront construction. *Geophysics*, 58(8):1157–1166, 1993.
- [78] G. B. Whitham. *Linear and nonlinear waves*. John Wiley & Sons, Inc., 1974.
- [79] L. Yatziv, A. Bartsaghi, and G. Sapiro. $o(n)$ implementation of the fast marching algorithm. *J. Comput. Phys.*, 212(2):393–399, 2006.
- [80] L. Ying and B. Engquist. Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers. *Multiscale Model. Simul.*, 9(2):686–710, 2011.
- [81] Y. T. Zhang and C. W. Shu. High-order WENO schemes for Hamilton-Jacobi equations on triangular meshes. *SIAM J. Sci. Comput.*, 24(3):1005–1030, 2003.
- [82] H. Zhao. Fast sweeping method for eikonal equations. *Math. Comput.*, 74: 603–627, 2005.