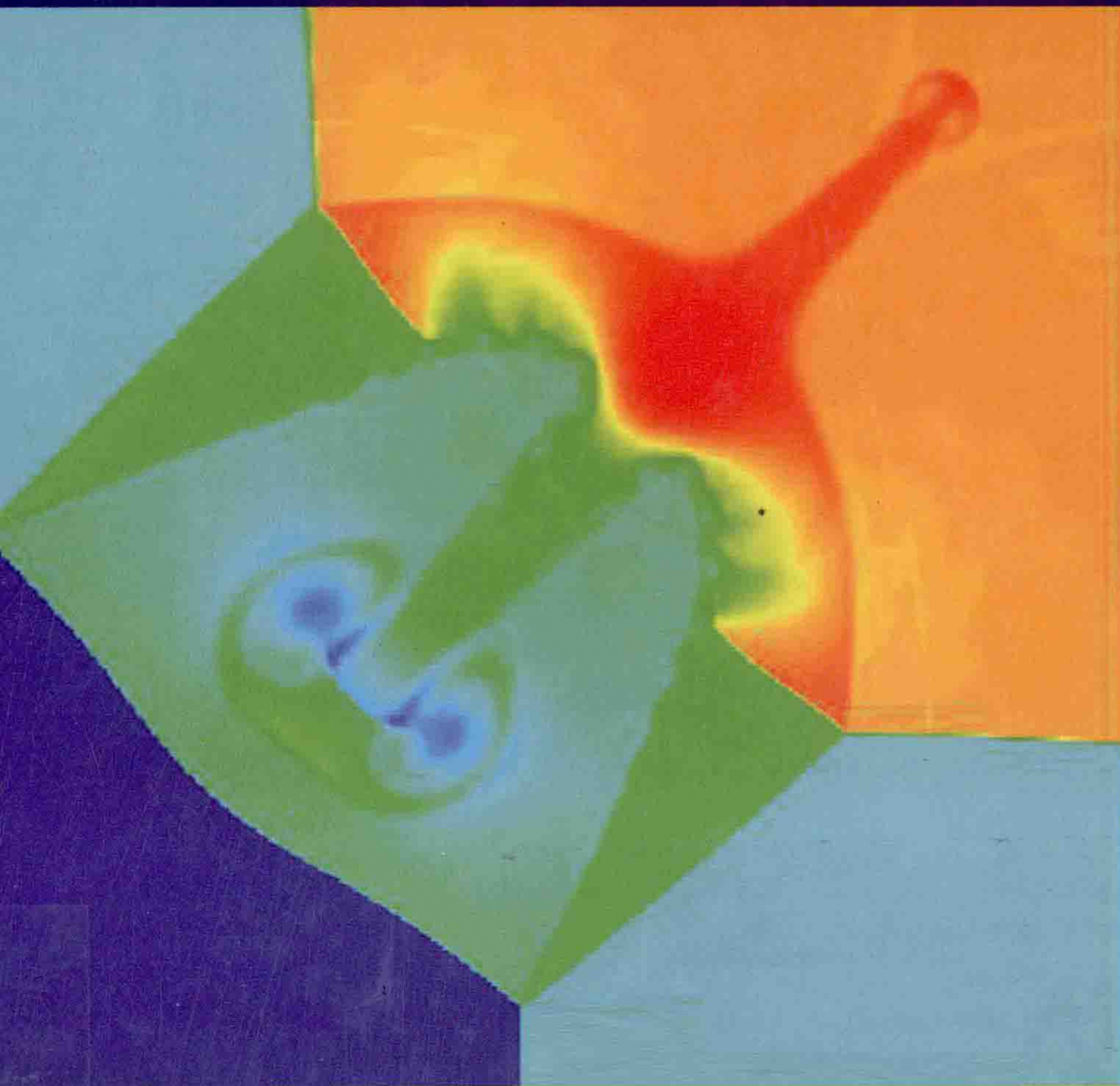


# NUMERICAL SOLUTION OF HYPERBOLIC PARTIAL DIFFERENTIAL EQUATIONS



**John A.  
Trangenstein**

CAMBRIDGE



# NUMERICAL SOLUTION OF HYPERBOLIC PARTIAL DIFFERENTIAL EQUATIONS

JOHN A. TRANGENSTEIN

Department of Mathematics, Duke University  
Durham, NC 27708-0320

 **CAMBRIDGE**  
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS  
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São  
Cambridge University Press  
The Edinburgh Building, Cambridge CB2 8RU, UK  
Published in the United States of America by Cambridge University Press

[www.cambridge.org](http://www.cambridge.org)  
Information on this title: [www.cambridge.org/9780521877275](http://www.cambridge.org/9780521877275)

© John A. Trangenstein 2009

This publication is in copyright. Subject to statutory exceptions  
and to the provisions of relevant collective licensing agreements,  
no reproduction of any part may take place without  
the written permission of Cambridge University Press.

First published 2009

Printed in the United Kingdom at the University Press, Cambridge

*A catalog record for this publication is available from the British Library*

ISBN 978-0-521-87727-5 Hardback

---

Cambridge University Press has no responsibility for the persistence or  
accuracy of URLs for external or third-party internet websites referred to  
in this publication, and does not guarantee that any content on those  
websites is, or will remain, accurate or appropriate.

All material contained within the CD-ROM is protected by copyright and other intellectual property rights.  
The customer acquires only the right to use the CD-ROM and does not acquire any other rights, express or implied, unless these are stated explicitly in a separate agreement.

To the extent permitted by applicable law, Cambridge University Press is not liable for any damage or loss of any kind resulting from the use of this product or from errors or omissions in the product, and in every case Cambridge University Press's liability shall be limited to the amount actually paid by the customer for the product.

---

# NUMERICAL SOLUTION OF HYPERBOLIC PARTIAL DIFFERENTIAL EQUATIONS

This is a new type of graduate textbook, with both print and interactive electronic components (on CD). It is a comprehensive presentation of modern shock-capturing methods including both finite volume and finite element methods, covering the theory of hyperbolic conservation laws and the theory of the numerical methods.

Classical techniques for judging the qualitative performance of the schemes, such as modified equation analysis and Fourier analysis, are used to motivate the development of classical higher-order methods (the Lax–Wendroff process) and to prove results such as the Lax Equivalence Theorem.

The range of applications (shallow water, compressible gas dynamics, magnetohydrodynamics, finite deformation in solids, plasticity, polymer flooding and water/gas injection in oil recovery) is broad enough to engage most engineering disciplines and many areas of applied mathematics.

The solution of the Riemann problems for these applications is developed, so that the reader can use the theory to develop test problems for the methods, especially to measure errors for comparisons of accuracy and efficiency. The numerical methods involve a variety of important approaches, such as MUSCL and PPM, TVD, wave propagation, Lax–Friedrichs (aka central schemes), ENO and discontinuous Galerkin; all of these are discussed in one and multiple spatial dimensions. Since many of these methods depend on Riemann solvers, there is extensive discussion of the basic design principles of approximate Riemann solvers, and several computationally useful techniques. The final chapter contains a discussion of adaptive mesh refinement via structured grids.

The accompanying CD contains a hyperlinked version of the text which provides access to computer codes for all of the text figures. Through this electronic text students can:

- See the codes and run them, choosing their own input parameters interactively
- View the online numerical results as movies
- Gain an appreciation for both the dynamics of the problem application, and the growth of numerical errors
- Download and modify the code for use with other applications
- Study the code to learn how to structure their programs for modularity and ease of debugging

JOHN A. TRANGENSTEIN is Professor of Mathematics at Duke University, North Carolina

**To James A. Rowe**



## Preface

Hyperbolic conservation laws describe a number of interesting physical problems in diverse areas such as fluid dynamics, solid mechanics, and astrophysics. Our emphasis in this book is on nonlinearities in these problems, especially those that lead to the development of propagating discontinuities. These propagating discontinuities can appear as the familiar shock waves in gases (the “boom” from explosions or super-sonic airplanes), but share many mathematical properties with other waves that do not appear to be so “shocking” (such as steep changes in oil saturations in petroleum reservoirs). These nonlinearities require special treatment, usually by methods that are themselves nonlinear. Of course, the numerical methods in this book can be used to solve linear hyperbolic conservation laws, but our methods will not be as fast or accurate as possible for these problems. If you are only interested in *linear* hyperbolic conservation laws, you should read about spectral methods and multipole expansions.

This book grew out of a one-semester course I have taught at Duke University over the past decade. Quite frankly, it has taken me at least 10 years to develop this material into a form that I like. I may tinker with the material more in the future because I expect that I will never be fully satisfied.

I have designed this book to describe both numerical methods and their applications. As a result, I have included substantial discussion about the analytical solution of hyperbolic conservation laws, as well as discussion about numerical methods. In this course, I have tried to cover the applications in such a way that the engineering students can see the mathematical structure that is common to all of these problem areas. With this information, I hope that they will be able to adapt new numerical methods developed for other problem areas to their own applications. I try to get the mathematics students to adopt one of the physical models for their computations during the semester, so that the numerical experiments can help them to develop physical intuition.



I also tried to discuss a variety of numerical methods in this text, so that you could see a number of competing ideas. This book does not try to favor a particular numerical scheme, and it does not serve as a user manual to a software package. It does have software available, to allow the reader to experiment with the various ideas. But the software is not designed for easy application to problems. Instead, I hope that the readers will learn enough from this book to make intelligent decisions on which scheme is best for their problems, as well as to implement that scheme efficiently.

There are a number of very good books on related topics. LeVeque's *Volume Methods for Hyperbolic Problems* [97] is one that covers the material well, describes several important numerical methods, but emphasizes the Godunov scheme over all. Other books are specialized for particular areas, such as Hirsch's *Numerical Computation of Internal and External Flows* [73], Peyret and Taylor's *Computational Methods for Fluid Flow* [131], Toro's *Computational Fluid Dynamics* [137] and Toro's *Riemann Solvers and Numerical Methods for Fluid Dynamics* [159]. These books contain very interesting material that is particular for fluid dynamics, and should not be ignored.

Because this text develops analytical solutions to several problems, it is easy to measure the errors in the numerical methods on interesting test problems. This relates to a point I try to emphasize in teaching the course, that it is not enough to use numerical computation to perform mesh refinement studies in order to check that the method is performing properly. Another topic in this text is that numerical methods can be compared for accuracy (error for a given mesh size) and efficiency (error for a given amount of computational time). Sometimes people have a bias toward higher-order methods, but this may not be the most cost-effective approach for many problems. Efficiency is tricky to measure, because programming issues can drive up computational time. I do not claim to have the most efficient version of any of the schemes in this text, so the efficiency comparisons should be taken "with a grain of salt."

The numerical comparisons produced some surprises for me. For example, I was surprised that approximate Riemann problem solvers often produce more accurate numerical results than Godunov methods than "exact" Riemann solvers. One surprise is that there is no clear best scheme or worst scheme in this text. I have omitted discussions of schemes that have fallen out of favor in the literature (for good reasons). There are some schemes that generally work better and some that often are less efficient than most, but all schemes have situations in which they perform well. The journal literature, of course, is full of examples of the latter behavior, since the authors get to choose computational examples that benefit their method.



During the past ten years, I have watched numerical methods evolve, computers gain amazing speed, and students struggle harder with programming. The evolution of the methods led me to develop the course material into a form that students could access online. In that way, I could insert additional text for ready access to the students. The speed of current desktop machines allows us to make some reasonably interesting computations during the semester, seeing in a few minutes what used to require overnight runs on supercomputers. During that time, however, the new operating systems have separated the students ever farther from programming details.

As I gained experience with online text generation, I started to ask if it would be possible to develop an interactive text. First, I wanted students to be able to view the example programs while they were reading the text online. Next, I wanted students to be able to examine links to information available on the web. Then, I decided that it would be really nice if students could perform “what if” experiments within the text, by running numerical methods with different parameters and seeing the results immediately. Because I continue to think that only “real” programming languages (*i.e.*, C, C<sup>++</sup> and Fortran) should be used for the material such as this, I rejected suggestions that I rewrite the programs in Matlab or Java. Eventually, our department systems programmer, Andrew Schretter, found a way to make things work for me, provided that I arrange for all parameter entry through graphical user interfaces. Our senior systems programmer, Yunliang Yu, did a lot of the development of the early form of the graphical user interface. One of my former graduate students, Wenjun Ying, programmed carefully the many cases for the marching cubes algorithm for visualizing level surfaces in three dimensions. I am greatly indebted to Andrew, Wenjun and Yunliang for their help.

This text is being published in two forms: traditional paper copy and a PDF file on a companion CD. The electronic form of the text contains links between equations and theorem references and the original statements. Similar links lead to bibliographic citations or to occurrences of key words in the index. There are electronic links in the online text to source code and executables on the CD. This allows students to view computer implementations of the algorithms developed in the book, and to perform “what if” experiments with program and model parameters. However, since the text is the same for both versions of the book, this means that the paper text contains instructions to click on electronic links.

The graphical user interface (GUI) makes it easy for students to change parameters (and, in fact, to see all of the input parameters). The GUI also complicates the online programs. There is a danger that students may think that they have program GUIs in order to solve these problems. That is not my intent. I have provided several example programs in the online version of chapter 2 to show students



how they can write simple programs (that produce data sets for post processing), or slightly more complex programs (that display numerical results during computation to look like movies), or very sophisticated programs (that use input parameters). I would be happy if all students could program successfully in the first style. After all, CLAWPACK is a very successful example of the simple and direct style of programming.

It is common that students in this class are taking it in order to learn programming in Fortran or C++, as much as they want to learn about the numerical methods. Both of these languages have advantages and disadvantages. Fortran is very good with arrays (subscripts can start at arbitrary values, which is useful for “ghosting” many methods) and has a very large set of intrinsic functions (for example, `max` and `min` with more than two arguments for slope limiters). Fortran is also good with memory allocation, or with pointers in general. I use C++ to manage memory allocation, and for all interactive graphics, including GUIs. When I select numerical methods through a GUI, then I set values for function parameters and pass those as arguments to Fortran routines. I do not recommend such mixing for novice programmers. On the other hand, students who want to extend their programming skills can find several interesting techniques in the codes.

I do try to emphasize **defensive programming** when I teach courses in scientific computing. By this term, I mean the use of programming practices that make it easier to prevent or identify programming errors. It is often possible to catch the use of uninitialized variables, the access of memory out of bounds, and memory leaks. The mixed-language programs all use the following defensive techniques. First, floating-point traps are enabled in unoptimized code. Second, floating-point array values are initialized to IEEE infinity. Third, a memory debugger is used to track memory allocation by overloading operator `new` in C++. When the program makes an allocation request, the memory debugger gets even more space on the heap, and puts special bit patterns into the space before and after the user memory. As a result, the programmer can ask the memory debugger to check individual pointers or all pointers for writes out of bounds. This memory debugger is very fast and does not add significantly to the overall memory requirements. The memory debugger also informs the programmer about memory leaks, providing information about where the unfreed pointer was allocated.

Unfortunately, mixing Fortran and C++ allows the possibility of triggering programming errors. For example, declaring a Fortran subroutine to have a return value in a C++ `extern "C"` block can lead to stack corruption. I do not know of a good defensive programming technique for that error.

But this book is really about numerical methods, not programming. I am more interested in hyperbolic conservation laws well after graduate school. I am indebted to several people for helping me to develop that interest. John

Gregory Shubin were particularly helpful when we worked together at Exxo Production Research. At Lawrence Livermore National Laboratory, I learnt much about Godunov methods from both John Bell and Phil Colella, and about object oriented programming from Bill Crutchfield and Mike Welcome. I want to thank all of them for their kind assistance during our years together.

Finally, emotional support throughout a project of this sort is essential. I want to thank my wife, Becky, for all her love and understanding throughout our years together. I could not have written this book without her.



# Contents

<i>Preface</i>	<i>page xv</i>
<b>1 Introduction to Partial Differential Equations</b>	
<b>2 Scalar Hyperbolic Conservation Laws</b>	
2.1 Linear Advection	
2.1.1 Conservation Law on an Unbounded Domain	
2.1.2 Integral Form of the Conservation Law	
2.1.3 Advection–Diffusion Equation	
2.1.4 Advection Equation on a Half-Line	
2.1.5 Advection Equation on a Finite Interval	
2.2 Linear Finite Difference Methods	
2.2.1 Basics of Discretization	
2.2.2 Explicit Upwind Differences	
2.2.3 Programs for Explicit Upwind Differences	
2.2.3.1 First Upwind Difference Program	
2.2.3.2 Second Upwind Difference Program	
2.2.3.3 Third Upwind Difference Program	
2.2.3.4 Fourth Upwind Difference Program	
2.2.3.5 Fifth Upwind Difference Program	
2.2.4 Explicit Downwind Differences	
2.2.5 Implicit Downwind Differences	
2.2.6 Implicit Upwind Differences	
2.2.7 Explicit Centered Differences	
2.3 Modified Equation Analysis	
2.3.1 Modified Equation Analysis for Explicit Upwind Differences	

- 2.3.2 Modified Equation Analysis for Explicit Downwind Differences
- 2.3.3 Modified Equation Analysis for Explicit Centered Differences
- 2.3.4 Modified Equation Analysis Literature
- 2.4 Consistency, Stability and Convergence
- 2.5 Fourier Analysis of Finite Difference Schemes
  - 2.5.1 Constant Coefficient Equations and Waves
  - 2.5.2 Dimensionless Groups
  - 2.5.3 Linear Finite Differences and Advection
  - 2.5.4 Fourier Analysis of Individual Schemes
- 2.6  $L^2$  Stability for Linear Schemes
- 2.7 Lax Equivalence Theorem
- 2.8 Measuring Accuracy and Efficiency
- 3 Nonlinear Scalar Laws**
  - 3.1 Nonlinear Hyperbolic Conservation Laws
    - 3.1.1 Nonlinear Equations on Unbounded Domains
    - 3.1.2 Characteristics
    - 3.1.3 Development of Singularities
    - 3.1.4 Propagation of Discontinuities
    - 3.1.5 Traveling Wave Profiles
    - 3.1.6 Entropy Functions
    - 3.1.7 Oleinik Chord Condition
    - 3.1.8 Riemann Problems
    - 3.1.9 Galilean Coordinate Transformations
  - 3.2 Case Studies
    - 3.2.1 Traffic Flow
    - 3.2.2 Miscible Displacement Model
    - 3.2.3 Buckley–Leverett Model
  - 3.3 First-Order Finite Difference Methods
    - 3.3.1 Explicit Upwind Differences
    - 3.3.2 Lax–Friedrichs Scheme
    - 3.3.3 Timestep Selection
    - 3.3.4 Rusanov’s Scheme
    - 3.3.5 Godunov’s Scheme
    - 3.3.6 Comparison of Lax–Friedrichs, Godunov and Rusanov
  - 3.4 Nonreflecting Boundary Conditions
  - 3.5 Lax–Wendroff Process
  - 3.6 Other Second Order Schemes



<b>4 Nonlinear Hyperbolic Systems</b>	<b>1</b>
4.1 Theory of Hyperbolic Systems	1
4.1.1 Hyperbolicity and Characteristics	1
4.1.2 Linear Systems	1
4.1.3 Frames of Reference	1
4.1.3.1 Useful Identities	1
4.1.3.2 Change of Frame of Reference for Conservation Laws	1
4.1.3.3 Change of Frame of Reference for Propagating Discontinuities	1
4.1.4 Rankine–Hugoniot Jump Condition	1
4.1.5 Lax Admissibility Conditions	1
4.1.6 Asymptotic Behavior of Hugoniot Loci	1
4.1.7 Centered Rarefactions	1
4.1.8 Riemann Problems	1
4.1.9 Riemann Problem for Linear Systems	1
4.1.10 Riemann Problem for Shallow Water	1
4.1.11 Entropy Functions	1
4.2 Upwind Schemes	1
4.2.1 Lax–Friedrichs Scheme	1
4.2.2 Rusanov Scheme	1
4.2.3 Godunov Scheme	1
4.3 Case Study: Maxwell’s Equations	1
4.3.1 Conservation Laws	1
4.3.2 Characteristic Analysis	1
4.4 Case Study: Gas Dynamics	1
4.4.1 Conservation Laws	1
4.4.2 Thermodynamics	1
4.4.3 Characteristic Analysis	1
4.4.4 Entropy Function	1
4.4.5 Centered Rarefaction Curves	1
4.4.6 Jump Conditions	1
4.4.7 Riemann Problem	2
4.4.8 Reflecting Walls	2
4.5 Case Study: Magnetohydrodynamics (MHD)	2
4.5.1 Conservation Laws	2
4.5.2 Characteristic Analysis	2
4.5.3 Entropy Function	2
4.5.4 Centered Rarefaction Curves	2
4.5.5 Riemann Problem	2

- 4.6 Case Study: Finite Deformation in Elastic Solids
  - 4.6.1 Eulerian Formulation of Equations of Motion for Solids
  - 4.6.2 Lagrangian Formulation of Equations of Motion for Solids
  - 4.6.3 Constitutive Laws
  - 4.6.4 Conservation Form of the Equations of Motion for Solids
  - 4.6.5 Jump Conditions for Isothermal Solids
  - 4.6.6 Characteristic Analysis for Solids
- 4.7 Case Study: Linear Elasticity
- 4.8 Case Study: Vibrating String
  - 4.8.1 Conservation Laws
  - 4.8.2 Characteristic Analysis
  - 4.8.3 Jump Conditions
  - 4.8.4 Lax Admissibility Conditions
  - 4.8.5 Entropy Function
  - 4.8.6 Wave Families for Concave Tension
  - 4.8.7 Wave Family Intersections
  - 4.8.8 Riemann Problem Solution
- 4.9 Case Study: Plasticity
  - 4.9.1 Lagrangian Equations of Motion
  - 4.9.2 Constitutive Laws
  - 4.9.3 Centered Rarefactions
  - 4.9.4 Hugoniot Loci
  - 4.9.5 Entropy Function
  - 4.9.6 Riemann Problem
- 4.10 Case Study: Polymer Model
  - 4.10.1 Constitutive Laws
  - 4.10.2 Characteristic Analysis
  - 4.10.3 Jump Conditions
  - 4.10.4 Riemann Problem Solution
- 4.11 Case Study: Three-Phase Buckley–Leverett Flow
  - 4.11.1 Constitutive Models
  - 4.11.2 Characteristic Analysis
  - 4.11.3 Umbilic Point
  - 4.11.4 Elliptic Regions
- 4.12 Case Study: Schaeffer–Schechter–Shearer System
- 4.13 Approximate Riemann Solvers
  - 4.13.1 Design of Approximate Riemann Solvers
  - 4.13.2 Artificial Diffusion
  - 4.13.3 Rusanov Solver
  - 4.13.4 Weak Wave Riemann Solver



## *Contents*

4.13.5	Colella–Glaz Riemann Solver	2
4.13.6	Osher–Solomon Riemann Solver	2
4.13.7	Bell–Colella–Trangenstein Approximate Riemann Problem Solver	2
4.13.8	Roe Riemann Solver	3
4.13.9	Harten–Hyman Modification of the Roe Solver	3
4.13.10	Harten–Lax–van Leer Scheme	3
4.13.11	HLL Solvers with Two Intermediate States	3
4.13.12	Approximate Riemann Solver Recommendations	3
<b>5</b>	<b>Methods for Scalar Laws</b>	<b>3</b>
5.1	Convergence	3
5.1.1	Consistency and Order	3
5.1.2	Linear Methods and Stability	3
5.1.3	Convergence of Linear Methods	3
5.2	Entropy Conditions and Difference Approximations	3
5.2.1	Bounded Convergence	3
5.2.2	Monotone Schemes	3
5.3	Nonlinear Stability	3
5.3.1	Total Variation	3
5.3.2	Total Variation Stability	3
5.3.3	Other Stability Notions	3
5.4	Propagation of Numerical Discontinuities	3
5.5	Monotonic Schemes	3
5.5.1	Smoothness Monitor	3
5.5.2	Monotonizations	3
5.5.3	MUSCL Scheme	3
5.6	Discrete Entropy Conditions	3
5.7	E-Schemes	3
5.8	Total Variation Diminishing Schemes	3
5.8.1	Sufficient Conditions for Diminishing Total Variation	3
5.8.2	Higher-Order TVD Schemes for Linear Advection	3
5.8.3	Extension to Nonlinear Scalar Conservation Laws	3
5.9	Slope-Limiter Schemes	3
5.9.1	Exact Integration for Constant Velocity	3
5.9.2	Piecewise Linear Reconstruction	3
5.9.3	Temporal Quadrature for Flux Integrals	3
5.9.4	Characteristic Tracing	3
5.9.5	Flux Evaluation	3
5.9.6	Non-Reflecting Boundaries with the MUSCL Scheme	3

- 5.10 Wave Propagation Slope Limiter Schemes
  - 5.10.1 Cell-Centered Wave Propagation
  - 5.10.2 Side-Centered Wave Propagation
- 5.11 Higher-Order Extensions of the Lax–Friedrichs Scheme
- 5.12 Piecewise Parabolic Method
- 5.13 Essentially Non-Oscillatory Schemes
- 5.14 Discontinuous Galerkin Methods
  - 5.14.1 Weak Formulation
  - 5.14.2 Basis Functions
  - 5.14.3 Numerical Quadrature
  - 5.14.4 Initial Data
  - 5.14.5 Limiters
  - 5.14.6 Timestep Selection
- 5.15 Case Studies
  - 5.15.1 Case Study: Linear Advection
  - 5.15.2 Case Study: Burgers’ Equation
  - 5.15.3 Case Study: Traffic Flow
  - 5.15.4 Case Study: Buckley–Leverett Model

## **6 Methods for Hyperbolic Systems**

- 6.1 First-Order Schemes for Nonlinear Systems
  - 6.1.1 Lax–Friedrichs Method
  - 6.1.2 Random Choice Method
  - 6.1.3 Godunov’s Method
    - 6.1.3.1 Godunov’s Method with the Rusanov Flux
    - 6.1.3.2 Godunov’s Method with the Harten–Lax–vanLeer (HLL) Solver
    - 6.1.3.3 Godunov’s Method with the Harten–Hyman Fix for Roe’s Solver
- 6.2 Second-Order Schemes for Nonlinear Systems
  - 6.2.1 Lax–Wendroff Method
  - 6.2.2 MacCormack’s Method
  - 6.2.3 Higher-Order Lax–Friedrichs Schemes
  - 6.2.4 TVD Methods
  - 6.2.5 MUSCL
  - 6.2.6 Wave Propagation Methods
  - 6.2.7 PPM
  - 6.2.8 ENO
  - 6.2.9 Discontinuous Galerkin Method



6.3	Case Studies	4
6.3.1	Wave Equation	4
6.3.2	Shallow Water	4
6.3.3	Gas Dynamics	4
6.3.4	MHD	4
6.3.5	Nonlinear Elasticity	4
6.3.6	Cristescu's Vibrating String	4
6.3.7	Plasticity	4
6.3.8	Polymer Model	4
6.3.9	Schaeffer–Schechter–Shearer Model	4
<b>7</b>	<b>Methods in Multiple Dimensions</b>	<b>4</b>
7.1	Numerical Methods in Two Dimensions	4
7.1.1	Operator Splitting	4
7.1.2	Donor Cell Methods	4
7.1.2.1	Traditional Donor Cell Upwind Method	4
7.1.2.2	First-Order Corner Transport Upwind Method	4
7.1.2.3	Wave Propagation Form of First-Order Corner Transport Upwind	4
7.1.2.4	Second-Order Corner Transport Upwind Method	4
7.1.3	Wave Propagation	4
7.1.4	2D Lax–Friedrichs	4
7.1.4.1	First-Order Lax–Friedrichs	4
7.1.4.2	Second-Order Lax–Friedrichs	4
7.1.5	Multidimensional ENO	4
7.1.6	Discontinuous Galerkin Method on Rectangles	4
7.2	Riemann Problems in Two Dimensions	4
7.2.1	Burgers' Equation	4
7.2.2	Shallow Water	5
7.2.3	Gas Dynamics	5
7.3	Numerical Methods in Three Dimensions	5
7.3.1	Operator Splitting	5
7.3.2	Donor Cell Methods	5
7.3.3	Corner Transport Upwind Scheme	5
7.3.3.1	Linear Advection with Positive Velocity	5
7.3.3.2	Linear Advection with Arbitrary Velocity	5
7.3.3.3	General Nonlinear Problems	5
7.3.3.4	Second-Order Corner Transport Upwind	5
7.3.4	Wave Propagation	5