

# Numerical Methods for Conservation Laws

# Computational Science & Engineering

The SIAM series on Computational Science and Engineering publishes research monographs, advanced undergraduate- or graduate-level textbooks, and other volumes of interest to an interdisciplinary CS&E community of computational mathematicians, computer scientists, scientists, and engineers. The series includes both introductory volumes aimed at a broad audience of mathematically motivated readers interested in understanding methods and applications within computational science and engineering and monographs reporting on the most recent developments in the field. The series also includes volumes addressed to specific groups of professionals whose work relies extensively on computational science and engineering.

SIAM created the CS&E series to support access to the rapid and far-ranging advances in computer modeling and simulation of complex problems in science and engineering, to promote the interdisciplinary culture required to meet these large-scale challenges, and to provide the means to the next generation of computational scientists and engineers.

---

## Editor-in-Chief

Donald Estep  
Colorado State University

Chen Greif  
University of British Columbia

J. Nathan Kutz  
University of Washington

## Editorial Board

Daniela Calvetti  
Case Western Reserve University

Jan S. Hesthaven  
Ecole Polytechnique Fédérale  
de Lausanne

Ralph C. Smith  
North Carolina State University

Paul Constantine  
Colorado School of Mines

Johan Hoffman  
KTH Royal Institute of Technology

Charles F. Van Loan  
Cornell University

Omar Ghattas  
University of Texas at Austin

David Keyes  
Columbia University

Karen Willcox  
Massachusetts Institute  
of Technology

## Series Volumes

Hesthaven, Jan S., *Numerical Methods for Conservation Laws: From Analysis to Algorithms*

Sidi, Avram, *Vector Extrapolation Methods with Applications*

Borzi, A., Ciaramella, G., and Sprengel, M., *Formulation and Numerical Solution of Quantum Control Problems*

Benner, Peter, Cohen, Albert, Ohlberger, Mario, and Willcox, Karen, editors, *Model Reduction and Approximation: Theory and Algorithms*

Kuzmin, Dmitri and Hämäläinen, Jari, *Finite Element Methods for Computational Fluid Dynamics: A Practical Guide*

Rostamian, Rouben, *Programming Projects in C for Students of Engineering, Science, and Mathematics*

Smith, Ralph C., *Uncertainty Quantification: Theory, Implementation, and Applications*

Dankowicz, Harry and Schilder, Frank, *Recipes for Continuation*

Mueller, Jennifer L. and Siltanen, Samuli, *Linear and Nonlinear Inverse Problems with Practical Applications*

Shapira, Yair, *Solving PDEs in C++: Numerical Methods in a Unified Object-Oriented Approach, Second Edition*

Borzi, Alfio and Schulz, Volker, *Computational Optimization of Systems Governed by Partial Differential Equations*

Ascher, Uri M. and Greif, Chen, *A First Course in Numerical Methods*

Layton, William, *Introduction to the Numerical Analysis of Incompressible Viscous Flows*

Ascher, Uri M., *Numerical Methods for Evolutionary Differential Equations*

Zohdi, T. I., *An Introduction to Modeling and Simulation of Particulate Flows*

Biegler, Lorenz T., Ghattas, Omar, Heinkenschloss, Matthias, Keyes, David, and van Bloemen Waanders, Bart, editors, *Real-Time PDE-Constrained Optimization*

Chen, Zhangxin, Huan, Guanren, and Ma, Yuanle, *Computational Methods for Multiphase Flows in Porous Media*

Shapira, Yair, *Solving PDEs in C++: Numerical Methods in a Unified Object-Oriented Approach*

JAN S. HESTHAVEN

EPFL

Lausanne, Switzerland

# Numerical Methods for Conservation Laws

## From Analysis to Algorithms



Society for Industrial and Applied Mathematics  
Philadelphia

Copyright © 2018 by the Society for Industrial and Applied Mathematics

10 9 8 7 6 5 4 3 2 1

All rights reserved. Printed in the United States of America. No part of this book may be reproduced, stored, or transmitted in any manner without the written permission of the publisher. For information, write to the Society for Industrial and Applied Mathematics, 3600 Market Street, 6th Floor, Philadelphia, PA 19104-2688 USA.

Trademarked names may be used in this book without the inclusion of a trademark symbol. These names are used in an editorial context only; no infringement of trademark is intended.

MATLAB is a registered trademark of The MathWorks, Inc. For MATLAB product information, please contact The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760-2098 USA, 508-647-7000, Fax: 508-647-7001, [info@mathworks.com](mailto:info@mathworks.com), [www.mathworks.com](http://www.mathworks.com).

<i>Publications Director</i>	Kivmars H. Bowling
<i>Executive Editor</i>	Elizabeth Greenspan
<i>Developmental Editor</i>	Gina Rinelli Harris
<i>Managing Editor</i>	Kelly Thomas
<i>Production Editor</i>	Lisa Briggeman
<i>Copy Editor</i>	Linda Jenkins
<i>Production Manager</i>	Donna Witzleben
<i>Production Coordinator</i>	Cally Shrader
<i>Compositor</i>	Cheryl Hufnagle
<i>Graphic Designer</i>	Lois Sellers

#### **Library of Congress Cataloging-in-Publication Data**

Please visit [www.siam.org/books/cs18](http://www.siam.org/books/cs18) to view the CIP data.

*To David*

2

# Contents

<b>Preface</b>	<b>xi</b>
<b>MATLAB Scripts</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Challenges ahead . . . . .	3
1.2 Conservation through history . . . . .	8
1.3 Some of the great equations of continuum mechanics . . . . .	10
1.3.1 Linear equations . . . . .	10
1.3.2 Nonlinear equations . . . . .	12
1.4 Test cases and software . . . . .	13
1.4.1 One-dimensional problems . . . . .	14
1.4.2 Two-dimensional problems . . . . .	18
1.5 What remains and what needs to be found elsewhere . . . . .	20
1.6 Audience and use . . . . .	22
References . . . . .	23
<b>I Conservation Laws</b>	<b>27</b>
<b>2 Scalar conservation laws</b>	<b>29</b>
2.1 Weak solutions . . . . .	32
2.2 Entropy conditions and solutions . . . . .	34
2.3 Entropy functions . . . . .	40
References . . . . .	47
<b>3 Systems of conservation laws</b>	<b>49</b>
3.1 Linear systems . . . . .	49
3.2 Riemann problems . . . . .	53
3.3 Entropy conditions and functions . . . . .	59
References . . . . .	64
<b>II Monotone Schemes</b>	<b>67</b>
<b>4 From continuous to discrete</b>	<b>69</b>
4.1 Conservation and conservation form . . . . .	71
4.2 Monotonicity and entropy conditions . . . . .	74
References . . . . .	87

---

<b>5</b>	<b>Finite difference methods</b>	<b>89</b>
5.1	Linear problems . . . . .	90
5.1.1	Consistency, stability, and convergence . . . . .	93
5.1.2	Nonsmooth problems . . . . .	99
5.2	Nonlinear problems . . . . .	102
5.3	Finite difference methods in action . . . . .	104
5.3.1	Boundary conditions . . . . .	104
5.3.2	Linear wave equation . . . . .	105
5.3.3	Burgers' equation . . . . .	107
5.3.4	Maxwell's equations . . . . .	110
5.3.5	Euler equations . . . . .	113
	References . . . . .	120
<b>6</b>	<b>Finite volume methods</b>	<b>123</b>
6.1	Godunov's method . . . . .	123
6.2	Approximate Riemann solvers . . . . .	128
6.2.1	Roe fluxes . . . . .	128
6.2.2	Engquist–Osher fluxes . . . . .	134
6.2.3	Harten–Lax–van Leer (HLL) fluxes . . . . .	135
6.3	Central schemes . . . . .	137
6.4	Finite volume methods in action . . . . .	138
6.4.1	The Euler equations . . . . .	139
	References . . . . .	144
<b>7</b>	<b>Beyond one dimension</b>	<b>147</b>
7.1	Two-dimensional monotone schemes in action . . . . .	149
7.1.1	Burgers' equation . . . . .	149
7.1.2	Nonconvex problem . . . . .	151
7.1.3	The Euler equations . . . . .	153
	References . . . . .	161
<b>III</b>	<b>High-Order Schemes</b>	<b>163</b>
<b>8</b>	<b>High-order accuracy and its challenges</b>	<b>165</b>
8.1	The good . . . . .	165
8.1.1	Phase error analysis . . . . .	170
8.2	The bad . . . . .	173
8.2.1	Total variation stability . . . . .	174
8.2.2	Entropy stability . . . . .	184
8.3	The ugly . . . . .	198
8.3.1	The Gibbs phenomenon . . . . .	201
8.3.2	Does it matter? . . . . .	206
8.3.3	What to do if it does matter . . . . .	209
	References . . . . .	216
<b>9</b>	<b>Strong stability preserving time integration</b>	<b>219</b>
9.1	Runge–Kutta methods . . . . .	222
9.1.1	Explicit strong stability preserving (SSP) Runge–Kutta schemes . . . . .	223

---

9.1.2	Implicit SSP Runge–Kutta schemes . . . . .	227
9.1.3	Order barriers . . . . .	228
9.2	Multistep methods . . . . .	232
	References . . . . .	234
<b>10</b>	<b>High-order accurate limiter-based methods</b>	<b>237</b>
10.1	Flux limited schemes . . . . .	238
10.1.1	Flux correction transport (FCT) schemes . . . . .	239
10.1.2	TVD stable high-order schemes . . . . .	240
10.1.3	Positive schemes . . . . .	247
10.1.4	Flux limited schemes in action . . . . .	250
10.2	Slope limited schemes . . . . .	260
10.2.1	Monotone upstream-centered scheme for conservation laws (MUSCL) . . . . .	260
10.2.2	Polynomial methods based on Lagrangian reconstruction . . . . .	265
10.2.3	Slope limited schemes in action . . . . .	268
10.3	Central schemes . . . . .	274
10.3.1	Central schemes in action . . . . .	281
10.4	Extension to multidimensional problems . . . . .	288
10.4.1	Burgers’ equation . . . . .	289
10.4.2	Nonconvex scalar equation . . . . .	294
10.4.3	Euler equations . . . . .	296
	References . . . . .	302
<b>11</b>	<b>Essentially nonoscillatory schemes</b>	<b>307</b>
11.1	Interpolation and reconstruction . . . . .	307
11.2	ENO methods . . . . .	314
11.2.1	ENO method for conservation laws . . . . .	316
11.2.2	A bit of theory . . . . .	318
11.2.3	ENO methods in action . . . . .	322
11.3	WENO methods . . . . .	337
11.3.1	WENO variants . . . . .	342
11.3.2	Well balanced schemes . . . . .	347
11.3.3	A little more theory . . . . .	350
11.3.4	WENO methods in action . . . . .	353
11.4	Dealing with nonuniform grids . . . . .	358
11.5	Beyond one dimension . . . . .	362
11.5.1	Interlude on non-Cartesian boundaries . . . . .	363
11.5.2	Scalar equations . . . . .	365
11.5.3	The Euler equations . . . . .	366
	References . . . . .	372
<b>12</b>	<b>Discontinuous Galerkin methods</b>	<b>377</b>
12.1	The basics . . . . .	383
12.1.1	The local approximation . . . . .	385
12.1.2	Key properties . . . . .	396
12.1.3	Error estimates . . . . .	402
12.1.4	Phase error analysis . . . . .	405
12.2	Nonsmooth problems . . . . .	408

---

12.2.1	A detour on hidden accuracy . . . . .	409
12.2.2	Filtering . . . . .	417
12.2.3	Nonlinear dissipation . . . . .	425
12.2.4	Slope limiters . . . . .	439
12.2.5	WENO-based limiters . . . . .	449
12.2.6	Extrema preserving limiters . . . . .	453
12.3	Related formulations . . . . .	457
12.3.1	Spectral penalty methods . . . . .	458
12.3.2	Spectral finite volume schemes . . . . .	461
12.3.3	Spectral difference schemes . . . . .	461
12.3.4	Flux reconstruction schemes . . . . .	464
12.4	Extension to multidimensional problems . . . . .	469
12.5	Discontinuous Galerkin methods in action . . . . .	472
12.5.1	Linear wave equation . . . . .	473
12.5.2	Burgers' equation . . . . .	476
12.5.3	Maxwell's equations . . . . .	481
12.5.4	The Euler equations . . . . .	484
	References . . . . .	492
13	<b>Spectral methods</b>	503
13.1	Fourier modes and nodes . . . . .	504
13.1.1	The continuous Fourier expansion . . . . .	504
13.1.2	The discrete Fourier expansion . . . . .	509
13.2	Fourier spectral methods . . . . .	514
13.2.1	Fourier–Galerkin methods . . . . .	514
13.2.2	Fourier collocation methods . . . . .	517
13.3	Nonlinear problems . . . . .	521
13.3.1	Skew-symmetric form . . . . .	522
13.3.2	Vanishing viscosity . . . . .	525
13.4	Postprocessing . . . . .	529
13.4.1	Filtering . . . . .	530
13.4.2	Fourier–Padé reconstruction . . . . .	535
13.4.3	Overcoming the Gibbs phenomenon . . . . .	541
13.5	Spectral methods in action . . . . .	550
13.5.1	Burgers' equation . . . . .	550
13.5.2	Maxwell's equations . . . . .	554
13.5.3	Euler equations . . . . .	556
	References . . . . .	559
	<b>Index</b>	565

# Preface

Emerging as the mathematical expression of principles of conservation, conservation laws have proven themselves to provide effective and accurate predictive models of our physical world. However, their solution, whether done mathematically or by computational techniques, remains a substantial challenge with many questions left open.

Driven by the importance of such models, extensive research during the past four decades has led to substantial advances in our understanding of such models and, in particular, the development of robust and accurate computational techniques to effectively solve such problems. The goal of this text is to offer a contemporary, in-depth introduction to central computational techniques for conservation laws. While it will be grounded in a discussion based on mathematical ideas and insight, we strive to maintain a focus on practical aspects of these methods, their implementation, and their application.

Computational methods only come to life when being expressed in software. Throughout the text, MATLAB is used to illustrate the many different techniques being discussed. To enable the reader to recreate the results in the text and gain additional experiences, all software, along with additional resources, is available at [www.siam.org/books/cs18](http://www.siam.org/books/cs18).

This text is dedicated to two people, both carrying the name of David. David Gottlieb, a professor of applied mathematics at Brown University, USA, until his untimely passing in 2008, was my mentor and colleague at Brown University during many of my formative years. However, he taught me so much more than computational mathematics. By being an almost impossible example to follow, he shared, through our many conversations and his friendship, his deep sense of humanism with me. These lessons, whether related to mathematics or to life itself, continue to inspire me in my personal and my professional life.

A second source of inspiration and curiosity continues to be my son, David. His being part of my life has opened my eyes to the playfulness and an unquestioned joy that one so easily forgets. Yet these are elements that are so essential in our short lives. He continues to bring a smile to my daily life, a life that would be poorer without him.

I would also like to thank a number of colleagues who have been instrumental for the completion of this text. A special thank you goes to Paolo Gatto, who provided detailed and unfiltered feedback on all parts of the text. This has undoubtedly helped improve the quality and readability of the text. Let me also express my gratitude to Sigal Gottlieb and Eitan Tadmor, who generously shared their views on earlier stages of the text. Their feedback helped shape the overall structure of the text in essential ways. The careful reading of earlier drafts by anonymous referees likewise helped to clarify numerous issues and reduced the number of misprints and inaccuracies. Needless to say, remaining mistakes and misprints are results originating solely from my shortcomings.

Thanks are also due to Tobin Driscoll and Bengt Fornberg for allowing me to reproduce two software scripts—`lagrangeweights.m` (section 11.3) and `SingularFourierPade.m` (section 13.4). Finally, I would also like to express my gratitude to Elizabeth Greenspan from SIAM for her help and encouragement during the process of completing this text.

Substantial parts of the material in this text have been developed while teaching a class on Numerical Methods for Conservations Laws at École Polytechnique Fédérale de Lausanne, and I am indebted to the thoughtful comments and many suggestions offered by the students of these classes. Other parts have been completed while being in residence at the Technical University of Denmark and the Chinese University of Hong Kong, and I am grateful for the hospitality during these stays.

*Jan S. Hesthaven*

École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland  
August 2017

# MATLAB Scripts

5.1	extend.m . . . . .	104
5.2	LinwaveMDriver1D.m . . . . .	105
5.3	LinwaveM1D.m . . . . .	106
5.4	LinwaveMrhs1D.m . . . . .	106
5.5	LinwaveLF.m . . . . .	106
5.6	LinwaveLW.m . . . . .	107
5.7	BurgersMDriver1D.m . . . . .	108
5.8	BurgersM1D.m . . . . .	108
5.9	BurgersMrhs1D.m . . . . .	109
5.10	BurgersLF.m . . . . .	109
5.11	BurgersLW.m . . . . .	109
5.12	BurgersRoe.m . . . . .	110
5.13	MaxwellMDriver1D.m . . . . .	110
5.14	CavityExact.m . . . . .	111
5.15	MaxwellM1D.m . . . . .	111
5.16	MaxwellMrhs1D.m . . . . .	112
5.17	MaxwellLF.m . . . . .	112
5.18	MaxwellLW.m . . . . .	112
5.19	EulerMDriver1D.m . . . . .	114
5.20	EulerM1D.m . . . . .	114
5.21	EulerMrhs1D.m . . . . .	115
5.22	EulerLF.m . . . . .	115
5.23	EulerLLF.m . . . . .	115
5.24	EulerLW.m . . . . .	116
6.1	Chebtau2p.m . . . . .	133
6.2	Chebeps.m . . . . .	133
6.3	EulerRoe.m . . . . .	139
6.4	EulerHLL.m . . . . .	140
6.5	EulerHLLC.m . . . . .	141
7.1	BurgersMDriver2D.m . . . . .	149
7.2	BurgersM2D.m . . . . .	150
7.3	BurgersMrhs2D.m . . . . .	150
7.4	KPPMDriver2D.m . . . . .	151
7.5	KPPM2D.m . . . . .	152
7.6	KPPMrhs2D.m . . . . .	152
7.7	KPPxLF.m . . . . .	153
7.8	KPPyLF.m . . . . .	153
7.9	EulerMDriver2D.m . . . . .	153

7.10	EulerM2D.m	155
7.11	EulerMrhs2D.m	155
7.12	EulerLF2Dx.m	156
7.13	EulerLF2Dy.m	156
7.14	EulerHLL2Dx.m	157
7.15	EulerHLL2Dy.m	158
7.16	IsentropicVortex2D.m	159
8.1	FilterFD.m	211
10.1	minmod.m	239
10.2	FluxLimit.m	251
10.3	LinwaveFLrhs1D.m	251
10.4	BurgersFLrhs1D.m	253
10.5	EulerFLrhs1D.m	255
10.6	EulerFLcharrhs1D.m	258
10.7	EulerChar.m	259
10.8	maxmod.m	264
10.9	minmodTVB.m	265
10.10	SlopeLimit.m	268
10.11	LinwaveSLrhs1D.m	269
10.12	BurgersSLrhs1D.m	272
10.13	EulerSL1D.m	273
10.14	EulerSLrhs1D.m	273
10.15	extendstag.m	281
10.16	LinwaveCrhs1D.m	282
10.17	BurgersFlux.m	284
10.18	BurgersJac.m	284
10.19	BurgersCrhs1D.m	284
10.20	EulerCrhs1D.m	286
10.21	EulerFlux.m	287
10.22	EulerJac.m	287
10.23	BurgersFLrhs2D.m	289
10.24	BurgersSLrhs2D.m	290
10.25	BurgersCrhs2D.m	291
10.26	BurgersFlux2Dx.m	293
10.27	BurgersFlux2Dy.m	293
10.28	BurgersJac2Dx.m	294
10.29	BurgersJac2Dy.m	294
10.30	KPPFlux2Dx.m	295
10.31	KPPFlux2Dy.m	295
10.32	KPPJac2Dx.m	295
10.33	KPPJac2Dy.m	295
10.34	EulerSL2D.m	297
10.35	EulerSLrhs2D.m	297
10.36	EulerFlux2Dx.m	299
10.37	EulerFlux2Dy.m	300
10.38	EulerJac2Dx.m	301
10.39	EulerJac2Dy.m	301
11.1	ReconstructWeights.m	312
11.2	LinearWeights.m	313
11.3	ddnewton.m	315

---

11.4	ENO.m	322
11.5	LegendreGQ.m	323
11.6	BurgersENODriver1D.m	324
11.7	BurgersENO1D.m	324
11.8	BurgersENOrhs1D.m	325
11.9	MaxwellENODriver1D.m	327
11.10	MaxwellENO1D.m	327
11.11	MaxwellENOrhs1D.m	328
11.12	MaxwellUpwind.m	330
11.13	EulerENODriver1D.m	331
11.14	EulerENO1D.m	332
11.15	EulerENOrhs1D.m	333
11.16	lagrangeweights.m	339
11.17	Qcalc.m	340
11.18	betarcalc.m	341
11.19	WENO.m	353
11.20	MaxwellWENO1D.m	354
11.21	EulerWENOcharrhs1D.m	356
11.22	KPPWENOrhs2D.m	365
11.23	EulerWENODriver2D.m	367
11.24	EulerWENO2D.m	368
11.25	EulerWENOrhs2D.m	369
12.1	LegendreP.m	387
12.2	LegendreGL.m	392
12.3	VandermondeDG.m	393
12.4	GradLegendreP.m	394
12.5	GradVandermondeDG.m	395
12.6	DmatrixDG.m	395
12.7	FilterDG.m	422
12.8	HeatDGrhs1D.m	432
12.9	HeatFlux.m	432
12.10	Nonvisc1.m	433
12.11	Nonvisc2.m	434
12.12	Entvisc.m	436
12.13	SlopeLimitCSDG.m	443
12.14	SlopeLimitBSBDG.m	445
12.15	MomentLimitDG.m	446
12.16	WENOlimitDG.m	450
12.17	WENODGWeights.m	451
12.18	extprelimitDG.m	456
12.19	extendDG.m	472
12.20	LinwaveDGDriver1D.m	474
12.21	LinwaveDG1D.m	474
12.22	LinwaveDGrhs1D.m	475
12.23	BurgersDGDriver1D.m	476
12.24	BurgersDG1D.m	479
12.25	BurgersDGrhs1D.m	480
12.26	MaxwellDGDriver1D.m	482
12.27	MaxwellDG1D.m	482
12.28	MaxwellDGrhs1D.m	483

12.29	EulerDGDriver1D.m	485
12.30	EulerDG1D.m	485
12.31	EulerDGrhs1D.m	487
12.32	EulerQtoRDG.m	488
12.33	EulerRtoQDG.m	489
13.1	Fourierdx.m	508
13.2	FourierD.m	511
13.3	FourierVanishHypVisc.m	528
13.4	FourierF.m	535
13.5	FourierPade.m	537
13.6	SingularFourierPade.m	538
13.7	GegenbauerP.m	544
13.8	GegenbauerRecon.m	546
13.9	GegenbauerGQ.m	548
13.10	GegenbauerPade.m	549
13.11	BurgersSpecDriver1D.m	551
13.12	BurgersSpec1D.m	551
13.13	BurgersSpecrhs1D.m	552
13.14	MaxwellSpecDriver1D.m	554
13.15	MaxwellSpec1D.m	555
13.16	MaxwellSpecrhs1D.m	555
13.17	EulerSpecDriver1D.m	557
13.18	EulerSpec1D.m	557
13.19	EulerSpecrhs1D.m	559

# Chapter 1

# Introduction

The central postulate, leading to the formulation of conservation laws, is one of balance in a physical system. The postulate states that the production of an entity, e.g., mass, energy, or charge, in a closed volume is exactly balanced by the flux of the entity across the boundary of the volume. In other words, an entity in a volume can only increase if more of it flows into the volume than what leaves.

Let us consider a simple physical system in which the mass per unit length, often referred to as the density,  $\rho(x, t)$ , of a material, e.g., a gas or a liquid, is distributed along  $x \in [x_1, x_2] = \Omega_x$ . The total mass is given as

$$M(t) = \int_{\Omega_x} \rho(x, t) dx.$$

If  $v(x, t)$  designates the local velocity of the gas at position  $x$  and time  $t$ , mass conservation is expressed as

$$\frac{d}{dt} \int_{\Omega_x} \rho(x, t) dx = \rho(x_1, t)v(x_1, t) - \rho(x_2, t)v(x_2, t).$$

This reflects a balance in which mass can only be gained by having a higher inward mass flux,  $\rho v$ , at  $x_1$  than an outward flux at  $x_2$ . Defining a temporal domain  $\Omega_t = [t_1, t_2]$  and integrating over this we recover

$$\int_{\Omega_x} \rho(x, t_2) dx - \int_{\Omega_x} \rho(x, t_1) dx = \int_{\Omega_t} \rho(x_1, t)v(x_1, t) - \rho(x_2, t)v(x_2, t) dt, \quad (1.1)$$

as a direct consequence of the basic physical principle that mass cannot be created or disappear by itself. This conservation principle is expressed in integral form which is, as we shall discuss in more details later, perhaps the most fundamental expression of a principle of conservation.

Taking the discussion one step further, we assume that the density and the velocity are both differentiable functions at  $(x, t)$ . The fundamental theorem of calculus immediately yields

$$\rho(x, t_2) - \rho(x, t_1) = \frac{d}{dt} \int_{\Omega_t} \rho(x, t) dt = \int_{\Omega_t} \frac{\partial}{\partial t} \rho(x, t) dt,$$

and, likewise,

$$\rho(x_2, t)v(x_2, t) - \rho(x_1, t)v(x_1, t) = \frac{d}{dx} \int_{\Omega_x} \rho(x, t)v(x, t) dx = \int_{\Omega_x} \frac{\partial}{\partial x} \rho(x, t)v(x, t) dx.$$

Combining this in (1.1), we recover

$$\int_{\Omega_x} \int_{\Omega_t} \left[ \frac{\partial \rho(x, t)}{\partial t} + \frac{\partial}{\partial x} (\rho(x, t)v(x, t)) \right] dt dx = 0.$$

Since this must hold for any volume  $\Omega_x \times \Omega_t$ , it must hold in a pointwise sense and we recover the conservation law on differential form

$$\frac{\partial \rho(x, t)}{\partial t} + \frac{\partial}{\partial x} (\rho(x, t)v(x, t)) = 0, \quad (x, t) \in \Omega_x \times \Omega_t.$$

If we take this just one step further and introduce the general conserved variable,  $u(x, t)$ , and the associated flux,  $f(u, t)$ , this yields the conservation law in integral form

$$\int_{\Omega_x} u(x, t_2) dx - \int_{\Omega_x} u(x, t_1) dx = \int_{\Omega_t} f(x_1, t) - f(x_2, t) dt, \quad (1.2)$$

and, provided  $u(x, t)$  and  $f(u, t)$  are both differentiable, the conservation law in differential form

$$\frac{\partial u(x, t)}{\partial t} + \frac{\partial f(u, t)}{\partial x} = 0, \quad (x, t) \in \Omega_x \times \Omega_t.$$

This prototype conservation law follows from the fundamental principle of conservation. The application of similar ideas leads to the formulation of systems of conservation laws in multiple dimensions. In this case, the statement of conservation is expressed as

$$\frac{d}{dt} \int_{\Omega_x} u(x, t) dV = - \oint_{\partial \Omega_x} \hat{n} \cdot f(u, x, t) dS, \quad x \in \Omega_x \subset \mathbb{R}^d, t > 0,$$

subject to appropriate initial conditions. Now  $u(x, t) : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  is the dependent variable,  $\hat{n}$  is an outward pointing unit vector along the boundary  $\partial \Omega_x$  of  $\Omega_x$ , and  $f(u, x, t) : \mathbb{R}^m \times \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  is the flux through the boundary. If we again assume that the solution and the flux are differentiable, we apply Gauss's theorem over a control volume to recover

$$\int_{\Omega_t} \int_{\Omega_x} \left( \frac{\partial}{\partial t} u(x, t) + \nabla \cdot f(u, x, t) \right) dx dt = 0.$$

Since this holds for any control volume, we recover the general conservation law in differential form

$$\frac{\partial u(x, t)}{\partial t} + \nabla \cdot f(u, x, t) = 0, \quad x \in \Omega_x, t > 0,$$

expressing a system of  $m$  conservation laws in  $d$ -dimensional space.

While perhaps simple looking the conservation principle—postulating conservation of mass, linear and angular momentum, energy of mechanical systems, electric charge,

etc.—is a tremendously powerful concept to model and understand the physical world around us.

Whereas the derivation of physical models based on conservation principles arises naturally, solving the resulting models, using either computational or mathematical means, remains a substantial challenge and an active area of research with numerous open questions. On the mathematical side, many fundamental questions of existence and uniqueness of the solutions remain open once the minimum of the spatial dimension and the size of the system exceeds one. In other words, scalar problems in multiple dimensions and nonlinear systems in one dimension are reasonably well understood, but many fundamental properties of systems in more than one spatial dimension remain poorly understood. Unfortunately, this latter case is exactly the most relevant for applications across science and engineering.

Without a rigorous mathematical understanding of the fundamental conservation laws, there are limits to the achievable rigor when one seeks to develop numerical solution techniques. Nevertheless, the last four decades have witnessed tremendous progress in the development and analysis of computational techniques for the solution of complex systems of conservation laws and the success of these models is unquestioned. A central goal of this text is to offer the reader an overview of many of these developments, supported by some analysis of the algorithms and accompanied with insights into the implementation of these algorithms.

## 1.1 • Challenges ahead

Conservation is a fundamental principle applied to model the physical world, stretching across quantum mechanics, continuum mechanics, and gravitational physics, with prominent examples including the Euler equations of gas dynamics, Navier's equations of elasticity, Maxwell's equations of electromagnetics, and Einstein's equations of gravitation.

It is therefore no surprise that the mathematical analysis of conservation laws is a deep research area with contributions stretching back to Leibniz. The development of efficient and accurate computational methods to solve such problems have occupied scientific pioneers such as von Neumann, Lax, and Lions. Unfortunately, as simple as the problem may appear, a closer look reveals some serious challenges when attempting to understand the basic conservation law and the nature of its solutions.

To appreciate some of the problems lying ahead, consider the simplest of all conservation laws,

$$\frac{\partial u}{\partial t} + \frac{\partial au}{\partial x} = 0, \quad x \in \mathbb{R},$$

where  $a \in \mathbb{R}$  is a finite constant. This equation is obtained with the flux  $f(u) = au$  and is recognized as the scalar one-way wave equation. With an initial condition  $u(x, 0) = g(x)$  one quickly verifies that the exact solution is

$$u(x, t) = g(x - at).$$

In other words, the sole action of the linear conservation law is to transport the initial condition at a constant speed,  $a$ . When  $a > 0$  the solution propagates to the right while for  $a < 0$  the solution propagates to the left.

By defining  $x_0 = x - at$ , we realize that the solution is constant along  $x_0$ , i.e.,

$$\frac{dx}{dt} = a, \quad x(0) = x_0.$$

We shall refer to these curves in the  $(x, t)$ -plane as characteristics. We observe that, in this particular case, all characteristics have the same slope and, thus, are parallel. Now consider the solution along such a characteristic as

$$\frac{d}{dt}u(x(t), t) = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} \frac{\partial x}{\partial t} = \frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

where the last step follows since  $u(x, t)$  is a solution to the conservation law. In other words, the solution is a constant along the characteristics. A slight generalization is obtained by considering

$$\frac{\partial u}{\partial t} + \frac{\partial a u}{\partial x} = -c u,$$

where  $c \in \mathbb{R}$  is a constant. In this case, the solution is easily found to be  $u(x, t) = e^{-ct} g(x - at)$ , i.e., the low-order term acts as a sink or a source depending on the sign of  $c$ .

Now consider the slightly more general problem

$$\frac{\partial u}{\partial t} + \frac{\partial a(x)u}{\partial x} = 0.$$

If  $a(x)$  is differentiable, we have

$$\frac{\partial u}{\partial t} + a(x) \frac{\partial u}{\partial x} = -\frac{da(x)}{dx} u,$$

and, thus, an equation for the characteristics on the form

$$\frac{dx}{dt} = a(x), \quad x(0) = x_0.$$

The solution along the characteristics satisfies

$$\frac{d}{dt}u(x(t), t) = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} \frac{\partial x}{\partial t} = \frac{\partial u}{\partial t} + a(x) \frac{\partial u}{\partial x} = -\frac{da(x)}{dx} u.$$

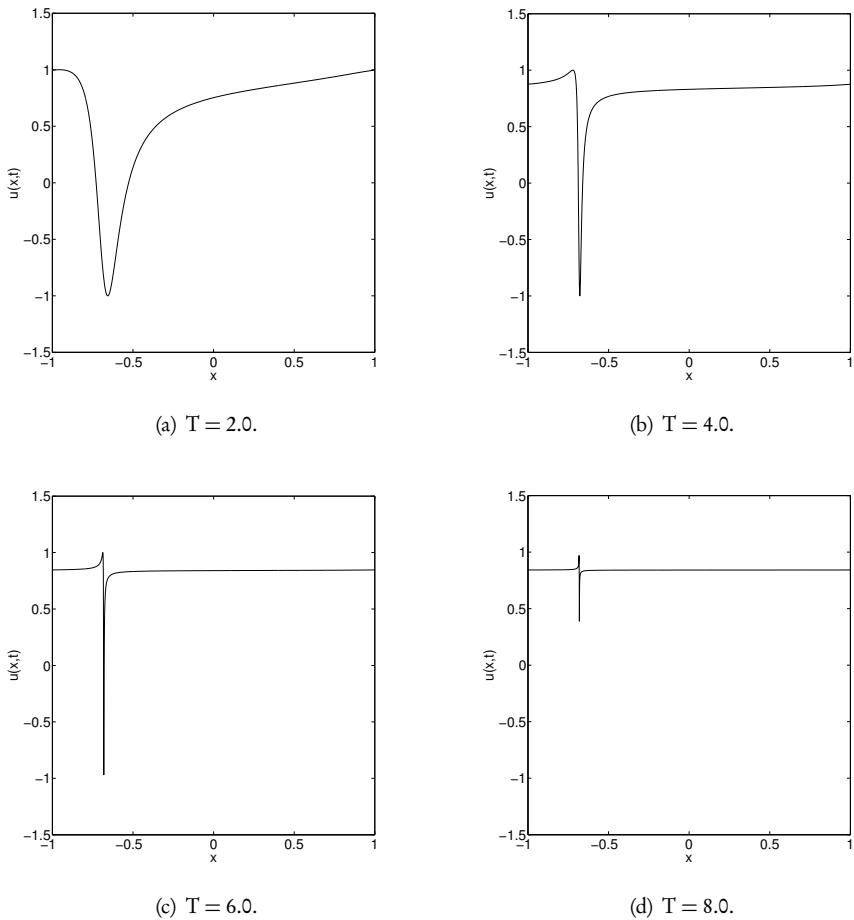
While the solution no longer remains constant along the characteristic lines, the overall picture of how the solution evolves remains the same.

This understanding of the solution is clear as long as  $a(x)$  does not change sign. However, if  $a(x)$  changes sign, the direction of propagation reverses and the evolution of the solution is less obvious. Let us consider an example to see what consequences this can have.

**Example 1.1.** We consider the problem

$$\frac{\partial u}{\partial t} + a(x) \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1], \quad (1.3)$$

subject to an initial condition  $u(x, 0) = \sin(\pi x)$  and periodic boundary conditions,  $u(-1, t) = u(1, t)$ .



**Figure 1.1.** The solution to the wave equation with a variable coefficient that changes sign at  $x = \pi^{-1} - 1 \simeq -0.682$ .

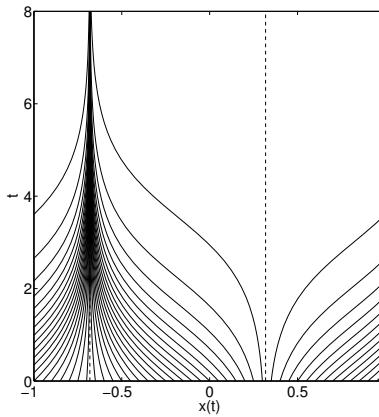
Let us assume that

$$a(x) = \frac{1}{\pi} \sin(\pi x - 1),$$

which changes sign in the domain of  $x$ . The exact solution to this problem is [5]

$$u(x, t) = \sin \left( 2 \arctan \left[ e^{-t} \tan \left( \frac{\pi x - 1}{2} \right) \right] + 1 \right).$$

We plot the solution at different times in Fig. 1.1 and observe the development of a very steep and highly localized solution around  $x = 1/\pi - 1 \simeq -0.682$ . This happens in spite of  $a(x)$  and the initial condition being analytic, i.e., the change in sign of  $a(x)$  renders the solution considerably more complex, even though it remains smooth and, asymptotically in time, relaxes to  $u(x, t) = \sin(1)$ .



**Figure 1.2.** The characteristics of the solution to the periodic wave equation (1.3) with a variable coefficient that changes sign at  $x = \pi^{-1} - 1$ . The dashed lines indicate the two critical points.

To understand this behavior, consider the characteristics of the problem, given by the initial value problem

$$\frac{dx}{dt} = \frac{1}{\pi} \sin(\pi x - 1), \quad x(0) = x_0.$$

For  $x \in [-1, 1]$ , this has two critical points, at  $x_1 = 1/\pi - 1$  and  $x_2 = 1/\pi$ . Away from these critical points, we recover the characteristics shown in Fig. 1.2. We observe that the formation of the strongly localized solution is a result of one critical point,  $x_1$ , being a stable fix point at which a substantial part of the mass of the solution concentrates, resulting in the very steep and localized solution. It should be noted that while the characteristics get very close, they never cross.

If we consider a problem in conservation form, i.e.,

$$\frac{\partial u}{\partial t} + \frac{\partial a(x)u}{\partial x} = 0, \quad x \in [-1, 1],$$

the integral of  $u$  would remain conserved and a singularity would have formed at  $x_1$ . However, for (1.3) we have

$$\frac{\partial u}{\partial t} + \frac{\partial a(x)u}{\partial x} = \frac{da(x)}{dx}u = \cos(\pi x - 1)u.$$

Recall that when  $\cos(\pi x - 1)$  takes on negative values, this term has a dissipative effect. Indeed we observe that  $\cos(\pi x - 1)$  takes on a maximum negative value exactly at  $x_1$ , which results in the asymptotic decay of the solution. ■

Taking this discussion one step further, let us consider a simple but nonlinear conservation law, known as Burgers' equation, given as

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

and the initial condition  $u(x, 0) = \sin(2\pi x)$ . For simplicity, we assume that the solution is periodic in space, i.e.,  $u(0, t) = u(1, t)$ . We immediately observe

$$\frac{d}{dt} \int_0^1 u(x, t) dx = 0.$$

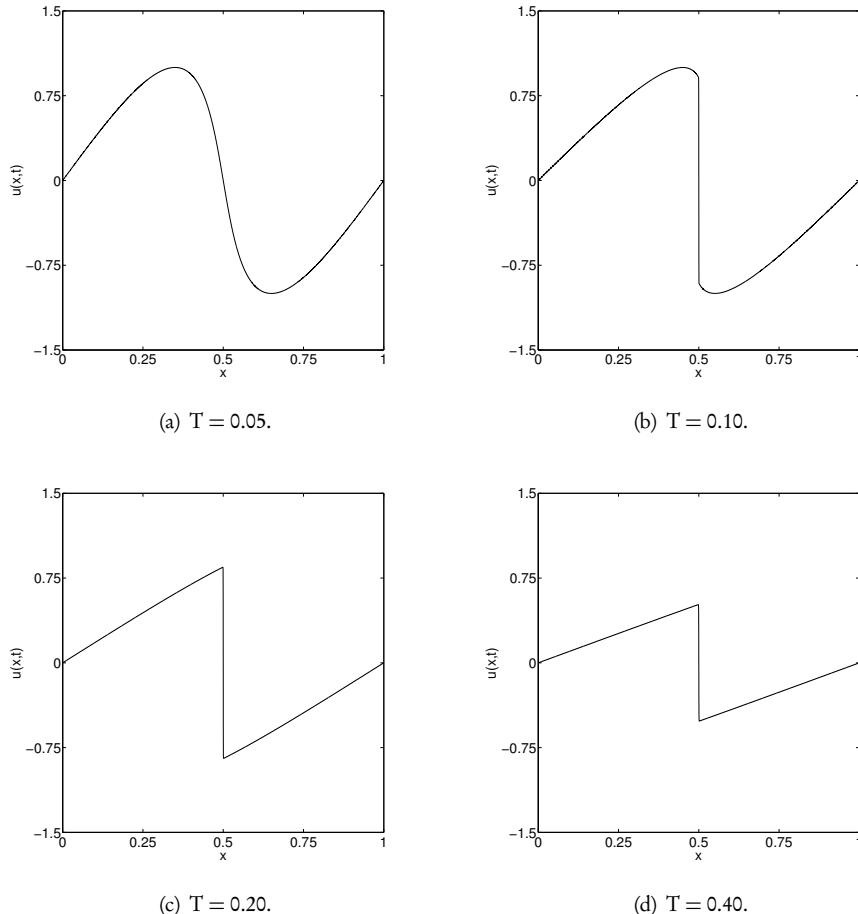
To understand the behavior of the solution, assume that  $u(x, t)$  is differentiable, at least locally in time. This allows us to write Burgers' equation as

$$\frac{\partial u}{\partial t} + 2u \frac{\partial u}{\partial x} = 0, \quad x \in [0, 1].$$

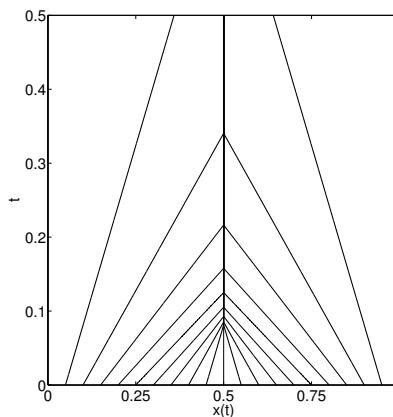
Clearly, this is similar to the variable coefficient case (1.3) and we recover the equation for the characteristics as

$$\frac{dx}{dt} = 2u(x, t), \quad x(0) = x_0.$$

In Fig. 1.3 we show the solution to Burgers' equation computed using a Lax-Friedrichs scheme (see Chapter 5) with high spatial and temporal resolution. We note the forma-



**Figure 1.3.** The solution to Burgers' equation at different times.



**Figure 1.4.** The characteristics of the solution to Burgers' equation.

tion of a discontinuous solution and the eventual dissipation of this. This discontinuous solution emerges from smooth initial conditions, i.e., it arises spontaneously as a solution to the nonlinear conservation law.

To shed some light on this behavior, we show in Fig. 1.4 the characteristics for the solution of Burgers' equation. Compared to the characteristics shown for the wave problem in Fig. 1.2, we observe a distinct difference—the characteristic lines cross.

The consequence of this crossing is that the history of the solution is destroyed, a discontinuity is formed, and many of the ideas discussed above, based on assumptions of minimal smoothness, become invalid.

This simple example helps us appreciate the central challenges associated with the understanding and solution of conservation laws:

- the possibility of spontaneous formation of discontinuous solutions, even from smooth initial conditions;
- loss of reversibility along the characteristics due to the intersection of the characteristics at the discontinuity;
- inadequacy of the differential form of the conservation law due to loss of smoothness and loss of classic solutions;
- preservation of conservation properties.

During the development and analysis of computational techniques for conservation laws, we must keep these simple observations in focus. Forgetting or, worse, ignoring these fundamental properties of the conservation law and the associated solution may result in methods of poor accuracy or, worse, methods that simply yield wrong solutions.

## 1.2 • Conservation through history

The concept of conservation as a fundamental property of nature was already familiar to the ancient Greeks, almost 2500 years ago. Thales of Miletus (624–546 BC) [32]

conjectured that certain elements of nature were conserved and that these substances made up the world around us. Empedocles (490–430 BC) [23] declared that these conserved substances, identified as water, air, fire, and earth, could neither be destroyed nor created. In a similar spirit, the geocentric universe, promoted by Aristotle (384–322 BC) [19] and Ptolemy (90–168) [34], in which planetary bodies, including the sun, orbit the earth in perfect circles, assumes conservation of the fixed distance between the earth and the other heavenly bodies.

This understanding of the heavenly world was not seriously challenged until the times of Copernicus (1473–1543) [33] and Galileo (1564–1642) [25], resulting in a model in which the planets orbit the sun in perfect circles. Kepler (1571–1630) [28] finally challenged the strong symmetry of this model, showing that the planetary trajectories were in fact elliptical, thereby destroying a last piece of a beautifully simple model of the cosmos. However, Kepler also formulated his second law of planetary motion as a conservation law—planets sweep out equal areas in equal time.

It appears to have been Leibniz (1646–1716) [26] who first attempted to formalize the concept of energy conservation by carefully studying mechanical systems, realizing that a motion-related energy—referred to as *vis viva* or “living force” by Leibniz—is conserved in many systems as long as the individual masses do not interact. However, it was soon realized that this concept of energy was inadequate, as many systems violate such a conservation principle due to conversion to other—at that time unknown—forms of energy. At the same time, it was held that the linear momentum of a mechanical system was conserved, although observations soon proved this understanding wrong.

Whereas there was no understanding of the conversion of energy and it remained unknown what happened to energy as it was lost, it was gradually being accepted that heat generated during friction accounted for another form of energy, initiating the discussion of mechanisms for conversion between different energy forms. The term *energy* appears to have been coined by Young (1773–1829) [36] to describe the totality of these different forms. Without understanding the details of the mechanisms of energy conversion, an understanding of the basic principle of conservation of energy began to take hold during the early parts of the 19th century.

At about the same time, Lavoisier (1743–1794) [18] formulated what we know today as the property of conservation of mass in a closed system. This was a crucial development as it also demonstrated that even when systems undergo change, i.e., from liquid to gas, mass is conserved as long as no external mass or energy is added. Soon thereafter, Joule (1818–1889) [27] demonstrated the equivalence between mechanical energy and heat, hence strengthening the insight already brought forward by Leibniz almost 200 years earlier.

In a different, and seemingly unrelated branch of physics, Franklin (1706–1790) [20], postulated what we know today as charge conservation, simply stating that charge within a system cannot change unless charge is added to the system from the outside.

While the idea that conservation of certain elements—energy, mass, linear momentum, charge—was taking hold during the 19th century, the connection between different conserved quantities remained poorly understood. With Einstein’s (1879–1955) [17] special theory of relativity in 1905, this fundamentally changed by demonstrating a direct connection between total energy and total mass through the famous law  $E = mc^2$ , connecting, for the first time, these two quantities.

The truly unifying development in the history of conservation principles came with the seminal theorem of Noether (1882–1935) [22], stating that any continuous symmetry of an action of a physical system is endowed with a corresponding conservation principle. While she did not cast this in its full generality for functions of space, as has

later been finalized, the original insight is clearly due to Noether. The implications are immediate. Conservation principles for linear and angular momentum are associated with translational and rotational symmetries in space, energy-mass conservations are driven by translational symmetries in time, and charge conservation is driven by what is known as gauge invariance.

With almost 2500 years of history, the importance of conservation principles as a tool to understand and model our world is unquestioned. These principles are applied when we formulate models to describe dynamical systems and their exchange of different types of energy, e.g., kinetic and internal energy. When these simple yet powerful principles are used to formulate balance laws for physical, chemical, or mechanical systems, the outcome is conservation laws of the form

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}) = 0.$$

When extended with thermodynamics, such models also include effects of heat generation and transfer.

Understanding such models is at the very core of our descriptive understanding of nature. It seems hard to imagine a better motivation for studying a subject like this, seeking to develop computational methods to solve problems in an effort to enhance our understanding of the world around us.

## 1.3 • Some of the great equations of continuum mechanics

With the application of conservation principles to formulate conservation laws, many of the great equations of continuum mechanics emerge naturally. In the following, let us revisit a few of the most prominent ones.

### 1.3.1 • Linear equations

A prominent linear example is Maxwell's equations [31] of electromagnetics

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E} + \mathbf{J}, \quad \frac{\partial \mathbf{D}}{\partial t} = \nabla \times \mathbf{H},$$

subject to the global constraints

$$\nabla \cdot \mathbf{D} = \rho, \quad \nabla \cdot \mathbf{B} = 0.$$

Here  $\mathbf{E} = \mathbf{E}(\mathbf{x}, t)$  and  $\mathbf{B} = \mathbf{B}(\mathbf{x}, t)$  represent the electric and magnetic vector fields, respectively, while  $\mathbf{D} = \mathbf{D}(\mathbf{x}, t)$  is the displacement vector field and  $\mathbf{H} = \mathbf{H}(\mathbf{x}, t)$  is the magnetizing vector field. Finally  $\mathbf{J} = \mathbf{J}(\mathbf{x}, t)$  represents the current density while  $\rho = \rho(\mathbf{x}, t)$  is the charge density. All vector fields are 3-vectors. If we restrict ourselves to materials with linear constitutive relations, the fields are related as

$$\epsilon \mathbf{E} = \mathbf{D}, \quad \mu \mathbf{H} = \mathbf{B},$$

where  $\epsilon$  and  $\mu$  are symmetric 3-tensors. Straightforward manipulations yield the charge conservation law

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J} = 0.$$

By defining

$$\mathbf{q} = \begin{bmatrix} \mathbf{E} \\ \mathbf{H} \end{bmatrix}, \quad \mathbf{f}_i = \begin{bmatrix} -\mathbf{e}_i \times \mathbf{H} \\ \mathbf{e}_i \times \mathbf{E} \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} \boldsymbol{\varepsilon} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mu} \end{bmatrix},$$

the flux  $\mathbf{f}(\mathbf{q}) = [f_1, f_2, f_3]$ , and with  $\mathbf{e}_i$  being the Cartesian unit vectors, we recover Maxwell's equations in conservation form as

$$\mathbf{Q} \frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{q}) = \begin{bmatrix} \mathbf{J} \\ \mathbf{0} \end{bmatrix}.$$

We can take this one step further and define the electromagnetic energy

$$E(t) = \frac{1}{2} [\mathbf{E} \cdot \mathbf{D} + \mathbf{H} \cdot \mathbf{B}],$$

and the Poynting vector,  $\mathbf{S} = \mathbf{E} \times \mathbf{H}$ , to recover

$$\frac{\partial E}{\partial t} + \nabla \cdot \mathbf{S} = -\mathbf{J} \cdot \mathbf{E}.$$

Hence, in the source free case where  $\mathbf{J} = \mathbf{0}$ , the electromagnetic energy satisfies a conservation law with the Poynting vector playing the role of the flux.

As a second example of equal importance, let us consider the equations of elastodynamics [29]

$$\rho \frac{\partial \mathbf{v}}{\partial t} = \nabla \cdot \mathcal{S} + \mathbf{b}, \quad (1.4)$$

where  $\rho = \rho(\mathbf{x}, t)$  is the density,  $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$  is the velocity 3-vector,  $\mathbf{b} = \mathbf{b}(\mathbf{x}, t)$  represents body forces, and  $\mathcal{S}$  is the 3-by-3 Cauchy stress tensor which, due to symmetry, only has 6 independent variables. For a linear material, we have the constitutive relation

$$\mathcal{S} = \mathbf{C} \mathbf{E},$$

connecting the stress with the strain,  $\mathbf{E} = \mathbf{E}(\mathbf{x}, t)$ , through the stiffness tensor  $\mathbf{C}$ . Here we have unfolded  $\mathcal{S}$  into the vector  $\mathcal{S} = [S_1, \dots, S_6]$  for the 6 independent variables and likewise for  $\mathbf{E}$ . With  $\mathcal{S}$  and  $\mathbf{E}$  being 6-vectors to account for anisotropic effects,  $\mathbf{C}$  has 36 elements but only 21 independent variables due to symmetries. The strain is directly related to the velocity as

$$\frac{\partial \mathbf{E}}{\partial t} = \frac{1}{2} (\nabla \mathbf{v} + (\nabla \mathbf{v})^T), \quad (1.5)$$

which needs to be understood in a matrix fashion for the symmetric  $3 \times 3$  strain matrix.

Let us introduce the operator ( $\mu = 1 \dots 3$ )

$$K_\mu = \begin{bmatrix} \delta_{\mu 1} & 0 & 0 & 0 & \delta_{\mu 3} & \delta_{\mu 2} \\ 0 & \delta_{\mu 2} & 0 & \delta_{\mu 3} & 0 & \delta_{\mu 1} \\ 0 & 0 & \delta_{\mu 3} & \delta_{\mu 2} & \delta_{\mu 1} & 0 \end{bmatrix},$$

with

$$\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$$

being the Kronecker delta. This allows (1.4) and (1.5) to be expressed as

$$\rho \frac{\partial \mathbf{v}}{\partial t} = \sum_{\mu=1}^3 K_{\mu} \frac{\partial S}{\partial x_{\mu}} + \mathbf{b}, \quad \frac{\partial E}{\partial t} = \sum_{\mu=1}^3 K_{\mu}^T \frac{\partial \mathbf{v}}{\partial x_{\mu}},$$

which, after some rewriting, is expressed as a conservation laws for  $\mathbf{v}$  and  $E$  in the velocity-stress formulation.

Combining (1.4) and (1.5), we recover

$$\rho \frac{\partial^2 \mathbf{v}}{\partial t^2} = \frac{1}{2} \nabla \cdot (\mathcal{C} (\nabla \mathbf{v} + (\nabla \mathbf{v})^T)) + \frac{\partial \mathbf{b}}{\partial t},$$

with  $\mathcal{C}$  encoding the stiffness tensor as a rank 4 tensor. We recognize this as the Navier-Cauchy equation [29] for elastic media with linear constitutive relations and, hidden in the notation, Newton's second law of motion.

With some additional algebraic manipulations we recover

$$\frac{d}{dt} \int_{\Omega_x} \left( \frac{\rho}{2} \mathbf{v} \cdot \mathbf{v} + S \cdot E \right) dx = \int_{\Omega_x} \nabla \cdot \mathbf{F} dx = \int_{\partial \Omega_x} \hat{\mathbf{n}} \cdot \mathbf{F} dx,$$

with  $\mathbf{F} = [F_1, F_2, F_3]$  and  $F_i = \mathbf{v} \cdot K_i S$ , recognized as a statement of conservation of the energy, comprising kinetic and elastic energy.

Both Maxwell's equations and the equations of elasticity are often expressed in their linear form as above. In this case the primary challenges in solving such problems are often found in the need to propagate waves over long distances and to model the interaction between waves and materials, in particular materials of great geometric and spatial variation. However, nonlinearities can enter into both models through the constitutive laws. In such cases, electric and magnetic properties of the materials depend on the fields in Maxwell's equations or the stress-strain relation becomes nonlinear. Understanding and modeling such nonlinear materials remains a research area of great activity and the need for accurate and efficient computational models is substantial.

### 1.3.2 • Nonlinear equations

Turning to nonlinear equations, the most studied conservation law is likely the Euler equations of gas dynamics. Assuming conservation of mass and linear momentum of a gas results in the equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0, \quad \frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{v}) = 0,$$

where we have the density,  $\rho = \rho(\mathbf{x}, t)$ , the linear momentum,  $\rho \mathbf{v} = \rho(\mathbf{x}, t) \mathbf{v}(\mathbf{x}, t)$ , the nonlinear flux,

$$\mathbf{f}(\mathbf{v}) = [f_1(\mathbf{v}), f_2(\mathbf{v}), f_3(\mathbf{v})],$$

$$f_i(\mathbf{v}) = [\rho v_1 v_i + \delta_{i1} p, \rho v_2 v_i + \delta_{i2} p, \rho v_3 v_i + \delta_{i3} p],$$

and the pressure,  $p = p(\mathbf{x}, t)$ .

If we also require conservation of energy, a third equation emerges:

$$\frac{\partial E}{\partial t} + \nabla \cdot [(E + p) \mathbf{v}] = 0.$$

Combining these five equations yields the famed Euler equations for gas dynamics [24],

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{q}) = 0,$$

with

$$\mathbf{q} = [\rho, \rho v_1, \rho v_2, \rho v_3, E]^T, \quad \mathbf{f}(\mathbf{q}) = [f_1(\mathbf{q}), f_2(\mathbf{q}), f_3(\mathbf{q})],$$

$$\mathbf{f}_i(\mathbf{q}) = [\rho v_i, \rho v_1 v_i + \delta_{i1} p, \rho v_2 v_i + \delta_{i2} p, \rho v_3 v_i + \delta_{i3} p, v_i(E + p)]^T.$$

The equations are closed by a thermodynamic relation, relating the density and the pressure, e.g., the law of ideal gases.

The inherently nonlinear nature of the Euler equations makes them very challenging to solve, even with modern computational techniques and substantial computational power at hand. This is a result of the nonlinearity and the formation of propagating waves as well as the spontaneous formation of a variety of discontinuous waves. We shall return to some of these challenges throughout the text, as the Euler equations remain the prototype nonlinear conservation law to consider when developing, understanding, and evaluating computational methods.

Unfortunately, the Euler equations mark just the beginning. If we consider problems with flow velocities close to the speed of light, strong relativistic effects must be accounted for, as done in relativistic fluid dynamics. Furthermore, under such circumstances, e.g., in stars and supernovae, the gases ionize and the dynamics of the fluid evolve under a larger set of equations, coupling gas dynamics with Maxwell's equations, leading to what is known as magneto hydrodynamics [30]. These more complex models are very challenging to solve even under simplified circumstances, yet they remain a simple and recognizable structure as a conservation law.

## 1.4 • Test cases and software

While the development and analysis of appropriate numerical techniques for conservation laws are core elements of this text, the subject only truly comes to life when being encoded into an algorithm and exercised through examples. To support this view, each major chapter, focusing on a specific family of methods, is concluded by a discussion of the computational aspects of the different methods. This is accompanied by embedded software, written in MATLAB [12], to illustrate the implementation of the different methods and support the results of the theoretical analysis. To offer some uniformity across the discussion of different methods, we illustrate the behavior of the methods through a few prototypical examples. To keep the discussion focused and the software manageable, we generally focus on one-dimensional examples. However, we demonstrate the performance and implementation also in two spatial dimensions for a selection of the methods being discussed.

While the examples and associated software are limited in scope and complexity, the goal is to provide the reader with sufficient background to explore and use more complex software libraries. We do not attempt to provide an overview of such options but refer to software libraries such as Clawpack [1], Deal.II [2], DUNE [4], and NUDG [13], where many of the algorithms discussed here are implemented with large scale applications in mind.

In the following we outline the basic examples and return to these as references when discussing the algorithmic aspects and the evaluation of the different computation methods.

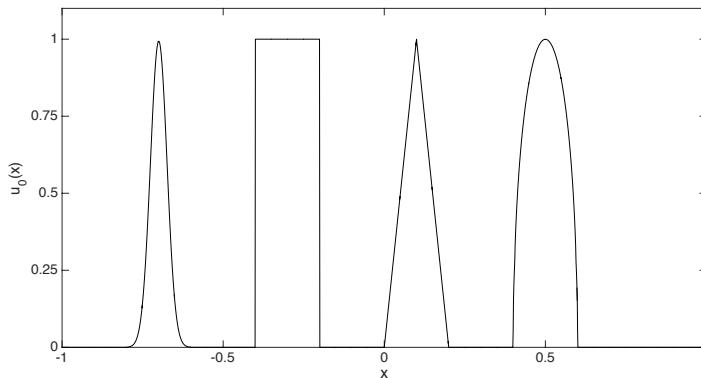


Figure 1.5. Initial conditions for the linear wave test case.

### 1.4.1 • One-dimensional problems

We introduce four different problems, linear and nonlinear, of increasing complexity. These are chosen to enable a comparative discussion of the different schemes.

#### Linear wave equation

As a first test case, we consider the linear advection problem

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions. The initial condition,  $u_0(x)$ , is given as [6]

$$u_0(x) = \begin{cases} \frac{1}{6}(G(x, \beta, z - \delta) + G(x, \beta, z + \delta) + 4G(x, \beta, z)), & -0.8 \leq x \leq -0.6, \\ 1, & -0.4 \leq x \leq -0.2, \\ 1 - |10(x - 0.1)|, & 0 \leq x \leq 0.2, \\ \frac{1}{6}(F(x, \alpha, a - \delta) + F(x, \alpha, a + \delta) + 4F(x, \alpha, z)), & 0.4 \leq x \leq 0.6, \\ 0 & \text{otherwise.} \end{cases} \quad (1.6)$$

Here we have defined

$$G(x, \beta, z) = \exp(-\beta(x - z)^2), \quad F(x, \alpha, a) = \sqrt{\max(1 - \alpha^2(x - a)^2, 0)},$$

and, following [6], we chose  $a = 0.5$ ,  $z = -0.7$ ,  $\delta = 0.005$ ,  $\alpha = 10$ ,  $\beta = \log 2 / (36\delta^2)$ .

As illustrated in Fig. 1.5, this initial condition comprises a combination of Gaussians, a square wave, a triangle wave, and half an ellipse.

#### Burgers' equation

As an example of a scalar nonlinear conservation law, we consider

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

with  $u = u(x, t)$ , subject to the initial condition  $u(x, 0)$ . This is recognized as Burgers' equation [21] and can be considered as a simple model of a one-dimensional fluid flow.

We shall primarily evaluate the properties of the different schemes for the simple case of a propagating shock

$$u(x, 0) = u_{x_0}(x) = \begin{cases} 2, & x \leq x_0, \\ 1, & x > x_0, \end{cases}$$

where the exact solution is  $u(x, t) = u_{x_0}(x - 3t)$ .

### Maxwell's equations

As an example of a linear system of equations with piecewise smooth solutions, consider the one-dimensional nondimensionalized Maxwell's equations [31]

$$\varepsilon(x) \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x}, \quad x \in [-1, 1],$$

where the relative permittivity,  $\varepsilon(x)$ , is assumed to be piecewise constant

$$\varepsilon(x) = \begin{cases} \varepsilon_1, & x \leq 0, \\ \varepsilon_2, & x > 0, \end{cases}$$

and  $\varepsilon_i \in \mathbb{R}^+$ .

If we assume  $E(-1, t) = E(1, t) = 0$ , the physical situation is that of a closed metallic cavity (partially) filled with a material specified by  $\varepsilon(x)$ . In such a case, the exact solution is given as [3]

$$\begin{aligned} E(x, t) &= [A_k e^{i n_k \omega x} - B_k e^{-i n_k \omega x}] e^{-i \omega t}, \\ H(x, t) &= n_k [A_k e^{i n_k \omega x} + B_k e^{-i n_k \omega x}] e^{-i \omega t}, \end{aligned} \quad (1.7)$$

where  $k = 1, 2$  represents the two materials and  $n_k = \sqrt{\varepsilon_k}$  is the index of refraction. The coefficients are given as

$$A_1 = \frac{n_2 \cos(n_2 \omega)}{n_1 \cos(n_1 \omega)}, \quad A_2 = e^{-i \omega(n_1 + n_2)},$$

and

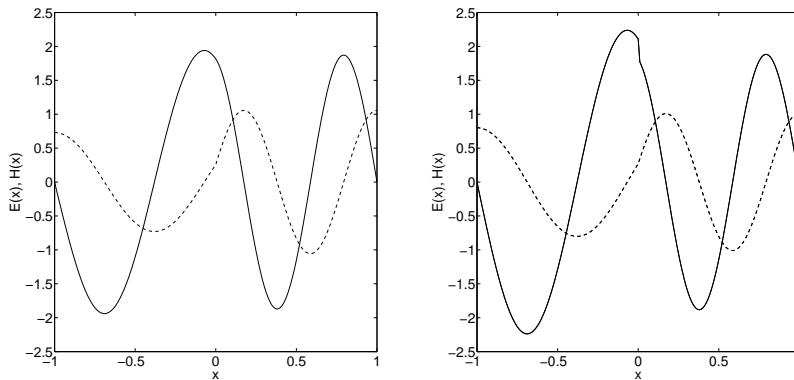
$$B_1 = A_1 e^{-i 2 n_1 \omega}, \quad B_2 = A_2 e^{i 2 n_2 \omega}.$$

In the special case of  $n_1 = n_2 = n$  and  $\omega = 2\pi/n$ , we recover a simple standing sinusoidal wave. For the more interesting case of  $n_1 \neq n_2$ , the resonance frequency  $\omega$  is found as the solution to the transcendental equation

$$-n_2 \tan(n_1 \omega) = n_1 \tan(n_2 \omega).$$

In this case, the solution is continuous across the interface at  $x = 0$  but discontinuous in the derivative. In Fig. 1.6 we show an example of the solution for  $\varepsilon_1 = 1, \varepsilon_2 = 2.25$ , which, roughly, corresponds to a cavity filled with air ( $\varepsilon = 1$ ) and glass ( $\varepsilon = 2.25$ ).

As an interesting modification to the test case, we assume that the interface, extended out of the plane, is inclined at an angle of  $\theta$  with the vertical interface. This results in



**Figure 1.6.** A snapshot of the solution to Maxwell's equations with discontinuous material coefficients. On the left, we show the fields for a vertical interface, displaying continuous solutions. The right shows a situation in which the interface is angled at 30 degrees, resulting in a discontinuous electric field. In both figures, the full line represents the electric field  $E$  while the dashed line represents the magnetic field  $H$ .

$E(x, t)$  being discontinuous at  $x = 0$ . If we take  $\varepsilon_1 = 1$  and  $\varepsilon_2 = \varepsilon \geq 1$ , the solution is given by (1.7) with [3]

$$\tan \omega = \frac{-\sqrt{\varepsilon} \tan(\sqrt{\varepsilon} \omega)}{1 - (\varepsilon - 1) \cos^2 \theta},$$

where  $\theta \in [0, \pi/2]$ . Taking  $\theta = 30$  degrees and  $\varepsilon = 2.25$ , Fig. 1.6 illustrates the discontinuous nature of the electric field.

Although the above model is cast in the language of electromagnetics, it can likewise be interpreted as a model for acoustic waves.

## Euler equations

As a final and considerably more complex one-dimensional example, we consider the one-dimensional Euler equations [24]

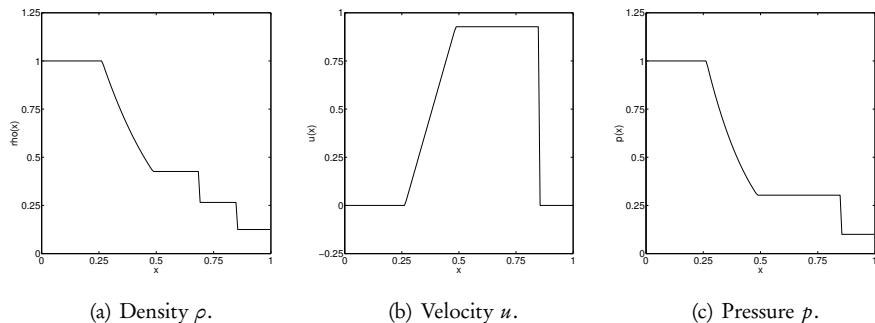
$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f} = 0,$$

where the conserved variables,  $\mathbf{q}$ , and the flux,  $\mathbf{f}$ , are given as

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (E + p)u \end{bmatrix}.$$

The equations are closed by the ideal gas law

$$p = (\gamma - 1) \left( E - \frac{1}{2} \rho u^2 \right), \quad c = \sqrt{\frac{\gamma p}{\rho}},$$



**Figure 1.7.** The solution to the Sod's shock tube problem at  $T = 0.2$ . The density  $\rho$ , the velocity  $u$ , and the pressure  $p$  are shown.

where  $c$  represents the local speed of sound and  $\gamma$  is a fluid-dependent constant. As is typical for atmospheric gases, we take  $\gamma = 7/5$ .

There are a number of classic test problems, known as shock tube problems, that allow the exact solution of the one-dimensional Euler problem. As one of these, we consider Sod's problem [35], defined by the initial conditions

$$\rho(x,0) = \begin{cases} 1.000, & x < 0.5, \\ 0.125, & x \geq 0.5, \end{cases} \quad \rho u(x,0) = 0, \quad E(x,0) = \frac{1}{\gamma-1} \begin{cases} 1.0, & x < 0.5, \\ 0.1, & x \geq 0.5. \end{cases}$$

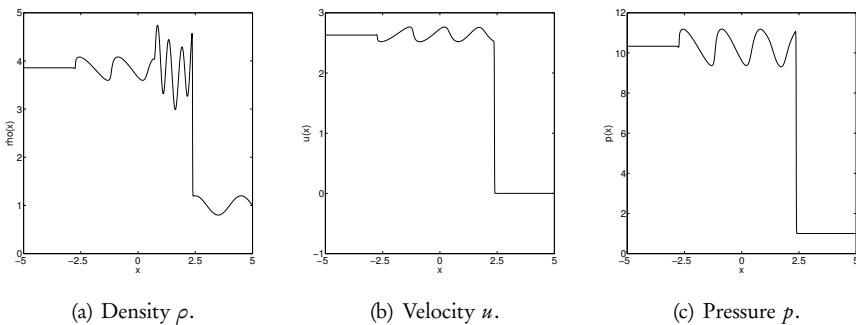
These also serve as boundary conditions, since any disturbance is assumed to stay away from the boundaries of the computational domain,  $x \in [0, 1]$ . The exact solution at  $T = 0.2$  is illustrated in Fig. 1.7. We recognize a shock at  $x \simeq 0.850$ , but also observe that other parts of the solution have limited smoothness. These will be identified as a contact discontinuity at  $x \simeq 0.685$  and a rarefaction wave between  $x \simeq 0.263$  and  $x \simeq 0.486$ .

As a second and considerably more complex test case, we consider the shock-entropy problem, with the initial conditions [14]

$$\begin{bmatrix} \rho \\ u \\ p \end{bmatrix} = \begin{cases} 3.857143, \\ 2.629369, \\ 10.33333, & x < -4.0, \\ 1 + 0.2 \sin(\pi x), \\ 0, \\ 1, & x \geq -4.0. \end{cases}$$

These also serve as boundary conditions, as disturbances are assumed not to reach the boundaries of the computational domain  $x \in [-5, 5]$ . As no exact solution is known for this problem, we compare our results with a highly resolved computational solution, illustrated in Fig. 1.8, where the solution is shown at time  $T = 1.80$ .

What makes this problem substantially more complex and challenging than Sod's problem is the mix of the discontinuity with high-frequency smooth features, all of which need to be carefully resolved. As we shall experience later, balancing these needs is a key challenge in the development of suitable methods for conservation laws.



**Figure 1.8.** The solution to the shock-entropy problem at  $T = 1.80$ . The density  $\rho$ , the momentum  $\rho u$ , and the pressure  $p$  are illustrated.

### 1.4.2 • Two-dimensional problems

Let us also consider a couple of two-dimensional test problems, focusing on nonlinear problems. Naturally, the one-dimensional linear problems discussed above can likewise be extended.

#### Nonconvex problem

We consider the two-dimensional nonconvex problem [8]

$$\frac{\partial u}{\partial t} + \nabla \cdot f(u) = 0, \quad (x, y) \in [-2, 2] \times [-2.5, 1.5],$$

with  $f(u) = (\sin(u), \cos(u))$ . The discontinuous initial condition is given as

$$u_0(x, y) = \frac{\pi}{4} \begin{cases} 14, & x^2 + y^2 < 1, \\ 1, & \text{otherwise.} \end{cases}$$

The problem has no exact solution and has been reported [8] to lead to wrong solutions when solved with particular schemes. In Fig. 1.9, a high resolution computational result illustrates the solution at  $T = 1$ .

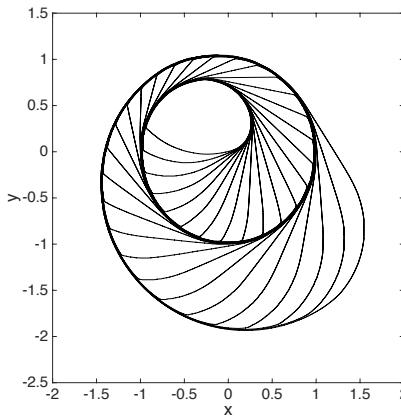
#### Euler equations

We also consider the two-dimensional version of the Euler equations

$$\frac{\partial q}{\partial t} + \nabla \cdot f = 0,$$

where the conserved variables,  $q$ , and the flux,  $f$ , are given as

$$q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \quad f_1 = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (E + p)u \end{bmatrix}, \quad f_2 = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (E + p)v \end{bmatrix},$$



**Figure 1.9.** High resolution reference solution for nonconvex scalar problem at  $T = 1$ .

with  $f = [f_1, f_2]$ . The equations are closed by the ideal gas law

$$p = (\gamma - 1) \left( E - \frac{1}{2} \rho (u^2 + v^2) \right), \quad c = \sqrt{\frac{\gamma p}{\rho}}.$$

Here  $c$  represents the local speed of sound and  $\gamma$  is a fluid-dependent constant. As is typical for atmospheric gases, we take  $\gamma = 7/5$ .

The maximum velocity, playing a central role in deciding the computational time step as we shall see later, is given by

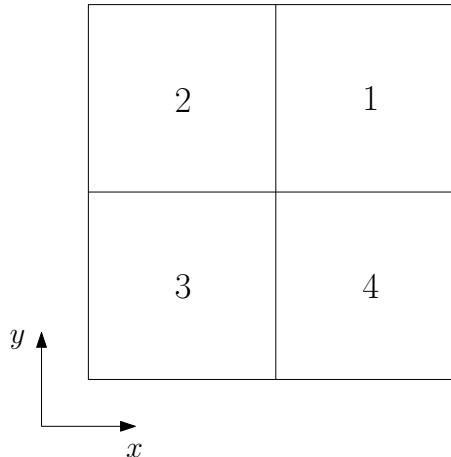
$$\alpha = \max_{\Omega_x} |c + \sqrt{u^2 + v^2}|.$$

As a smooth test case, we consider an isentropic vortex given as

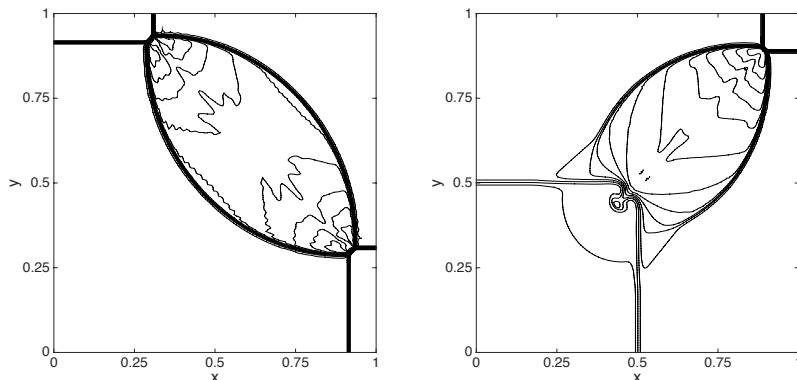
$$\begin{aligned} \rho &= \left( 1 - \left( \frac{\gamma - 1}{16\gamma\pi^2} \right) \beta^2 e^{2(1-r^2)} \right)^{\frac{1}{\gamma-1}}, \\ u &= u_0 - \beta e^{(1-r^2)} \frac{y - y_0}{2\pi}, \\ v &= v_0 + \beta e^{(1-r^2)} \frac{x - x_0}{2\pi}, \\ p &= \rho^\gamma, \end{aligned}$$

where  $r = \sqrt{(x - u_0 t - x_0)^2 + (y - v_0 t - y_0)^2}$ . This is an exact solution to the compressible Euler equations [37], comprising a vortex, initially centered at  $(x_0, y_0)$ , and convected at the constant velocity  $(u_0, v_0)$ .

To further evaluate the quality of the methods, we consider a two-dimensional version of the shock tube problems, comprising piecewise smooth initial conditions. The initial setup is sketched in Fig. 1.10, illustrating the four quadrants, each of which



**Figure 1.10.** Initial conditions for the two-dimensional Riemann problem are set constant in the four quadrants as sketched above. The piecewise constant values for 19 distinct cases can be found in [10, 9].



**Figure 1.11.** Solutions of Riemann problem number 4 (left) and Riemann problem number 12 (right) [10, 9].

has a constant state. As discussed in detail in [10, 9], there are 19 distinct cases, giving rise to a combination of shocks, rarefaction waves, and contacts.

These 19 problems have no exact solutions and we can only consider the convergence of the solutions with increasing resolution. In Fig. 1.11 we show well resolved solutions for Case 4, comprising four shocks, and Case 12, comprising 2 shocks and 2 contact waves. We refer to [10, 9] for computational results for all 19 cases.

## 1.5 • What remains and what needs to be found elsewhere

The text is split into three main parts. In Part I, we discuss some theoretical aspects of conservation laws and the solution of both linear and nonlinear problems. For linear

problems we address both scalar problems and systems, and introduce the Riemann problem. When discussing the nonlinear conservation law, we begin by considering the scalar problem and introduce key concepts such as weak solutions, uniqueness, and entropy conditions, all of which shall later play a key role in the analysis of the numerical schemes. We also scratch the surface of the discussion of solutions to systems of nonlinear conservation laws, but will not attempt to reach deep into this active and very technical research area.

This provides the necessary background and sets the stage for the remaining two central parts of the text, focusing on monotone schemes and high-order schemes, respectively, for solving conservation laws. In Part II we discuss in detail linear methods for solving general conservation laws, covering finite difference and finite volume methods. The discussion focuses on the analysis of these methods and their performance when applied to the test cases discussed above. We utilize the extensive theoretical foundation on which these methods stand to provide insight into the accuracy and stability for both linear and nonlinear problems and include a discussion of how these methods can be extended to solve problems in higher spatial dimensions.

The last and most extensive part, Part III, provides a comprehensive discussion of computational techniques of high-order accuracy for conservation laws. Due to Godunov's theorem, discussed in Part II, this is a delicate issue. The first chapter provides, along with the necessary tools, some background and motivation for attempting such developments.

Before continuing with the development of spatial schemes, we return to the temporal dimension and discuss in some detail the development of strong stability preserving schemes as they provide a central element in the analysis of high-order accurate methods and their nonlinear stability.

This lays the foundation for the discussion of high-order extensions of finite volume methods, ensuring high-order accuracy in smooth regions of the solution without introducing nonphysical oscillations in the neighborhood of discontinuities. The majority of these schemes are based on ideas of limiting to achieve high-order accuracy without introducing oscillations.

In the following chapter we introduce essentially nonoscillatory schemes, providing a novel and powerful approach to achieve high-order accuracy for complex problems with strongly discontinuous solutions. We development both essentially nonoscillatory (ENO) and weighted essentially nonoscillatory (WENO) methods in detail, including a discussion of their extension to multiple dimensions.

The development of high-order finite volume schemes and ENO methods illustrates well their potential but also highlights that their extension to more complex situations such as nonuniform grids or higher-dimensional versions utilizing unstructured grids is, at best, difficult. To address this issue, we turn the discussion toward a different approach based on an element based discretization of space. The resulting methods, known as discontinuous Galerkin methods, share many of the desirable properties of finite elements methods while retaining the robustness of the finite volume methods. We provide an overview of the theoretical aspects of discontinuous Galerkin methods and discuss the importance of special techniques to avoid oscillations near discontinuities.

In the final chapter, we introduce spectral methods for conservation laws, including the use of vanishing viscosity to control oscillations and reconstruction based techniques to enable the recovery of high accuracy solutions.

For a topic as mature and extensively developed as that of numerical methods for conservation laws, any author is faced with having to make choices. The emphasis of

this text is on providing a solid foundation in the analysis, algorithmic development, and basic application of modern computational methods for solving conservation laws. This focus has served as a guide in selecting and developing the material.

In particular, we restrict the discussion and analysis of fundamental properties of conservation laws and their solutions to what is required to enable the subsequent analysis of the computational methods. Furthermore, we will not discuss techniques that are developed specifically for steady-state problems.

The perhaps most restricting consequence of the emphasis on the fundamentals is the focus on one-dimensional problems or, if higher-dimensional, problems posed on rectangular domains. As a consequence, we will not discuss the important development of methods to problems on unstructured grids. Naturally, there is a substantial body of work that addresses this due to the central importance for realistic applications; see e.g., [7, 11, 16, 15].

In a similar spirit we also do not address efforts devoted to error estimation and grid adaptivity or problems associated with implicit time-stepping and related nonlinear solvers.

While these, and numerous other, aspects of solving conservation laws are not covered in detail here, we strive throughout the text to provide references to the research literature that points toward these more advanced topics. Equipped with the tools and insight provided by the text, it is our hope that such more advanced material is accessible.

## 1.6 • Audience and use

The main goal of this text is to offer a balanced but thorough introduction to the development of computational methods for conservation laws, focusing on a selected number of powerful techniques and their analysis. We strive to offer a solid foundation in the mathematical analysis of the conservation laws, while maintaining an ongoing attention to central computational and algorithmic aspects.

As mentioned, the development of each central computational methods will be complemented with embedded MATLAB software to illustrate the solution of a range of problems and highlight key elements of each algorithm. While not attempting to be fully reproducible, the software should allow the user to recreate all central computational results and figures in the text. All software is available at [www.siam.org/books/cs18](http://www.siam.org/books/cs18).

The level of the text is aimed at graduate level studies in computational science and related areas but is too comprehensive to be covered in detail during a semester class. A solid background in applied and numerical analysis is needed, and a first course in partial differential equations is certainly an advantage. Using the text in a course, one could combine Parts I and II to provide an introduction to conservation laws and their solution using more traditional methods. One could likewise combine Part I and parts of Part III in a course focusing on high-order accurate methods. Another approach could be to combine Part I with material on finite volume methods and discontinuous Galerkin methods as a comprehensive introduction to the latter family of methods for nonlinear conservation laws.

Regardless of which combination is being considered or whether the text is used for self study, the embedded software aims to enrich the effort of the reader to understand this fascinating, but at times complex and demanding material, to a level where it can be applied to solve problems across the sciences and engineering.

## References

- [1] Clawpack development team. *Clawpack Software*. <http://www.clawpack.org>, 2015.
- [2] Deal.II development team. *deal.II Software*. <http://www.dealii.org>.
- [3] Adi Ditkowski, Kim Dridi, and Jan S. Hesthaven. Convergent Cartesian grid methods for Maxwell's equations in complex geometries. *Journal of Computational Physics*, 170(1):39–80, 2001.
- [4] DUNE development team. *Distributed and Unified Numerics Environment (DUNE)*. <https://www.dune-project.org>, 2015.
- [5] Jan S. Hesthaven and Robert Kirby. Filtering in Legendre spectral methods. *Mathematics of Computation*, 77(263):1425–1452, 2008.
- [6] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 1(126):202–228, 1996.
- [7] Dietmar Kröner. *Numerical Schemes for Conservation Laws*. Wiley, 1997.
- [8] Alexander Kurganov, Guergana Petrova, and Bojan Popov. Adaptive semidiscrete central-upwind schemes for nonconvex hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 29(6):2381–2401, 2007.
- [9] Alexander Kurganov and Eitan Tadmor. Solution of two-dimensional Riemann problems for gas dynamics without Riemann problem solvers. *Numerical Methods for Partial Differential Equations*, 18(5):584–608, 2002.
- [10] Peter D. Lax and Xu-Dong Liu. Solution of two-dimensional Riemann problems of gas dynamics by positive schemes. *SIAM Journal on Scientific Computing*, 19(2):319–340, 1998.
- [11] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*, volume 31 of Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002.
- [12] The Mathworks. MATLAB Release 2016a, The MathWorks, Inc. <http://www.mathworks.com>, 2015.
- [13] NUDG development team. Nodal Unstructured Discontinuous Galerkin Methods (NUDG). <http://www.nudg.org>, 2015.
- [14] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 83(1):32–78, 1989.
- [15] John A. Trangenstein. *Numerical Solution of Hyperbolic Partial Differential Equations*. Cambridge University Press, 2009.
- [16] Henk K. Versteeg and Weeratunge Malalasekera. *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education, 2007.
- [17] Wikipedia contributors. *Albert Einstein*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Albert\\_Einstein](https://en.wikipedia.org/wiki/Albert_Einstein) (accessed August 2014).

- 
- [18] Wikipedia contributors. *Antoine Lavoisier*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Antoine\\_Lavoisier](https://en.wikipedia.org/wiki/Antoine_Lavoisier) (accessed August 2014).
  - [19] Wikipedia contributors. *Aristotle*. Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/wiki/Aristotle> (accessed August 2014).
  - [20] Wikipedia contributors. *Benjamin Franklin*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Benjamin\\_Franklin](https://en.wikipedia.org/wiki/Benjamin_Franklin) (accessed August 2014).
  - [21] Wikipedia contributors. *Burgers' equation*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Burgers'\\_equation](https://en.wikipedia.org/wiki/Burgers'_equation) (accessed August 2014).
  - [22] Wikipedia contributors. *Emmy Noether*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Emmy\\_Noether](https://en.wikipedia.org/wiki/Emmy_Noether) (accessed August 2014).
  - [23] Wikipedia contributors. *Empedocles*. Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/wiki/Empedocles> (accessed August 2014).
  - [24] Wikipedia contributors. *Euler equations (fluid dynamics)*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Euler\\_equations\\_\(fluid\\_dynamics\)](https://en.wikipedia.org/wiki/Euler_equations_(fluid_dynamics)) (accessed August 2014).
  - [25] Wikipedia contributors. *Galileo Galilei*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Galileo\\_Galilei](https://en.wikipedia.org/wiki/Galileo_Galilei) (accessed August 2014).
  - [26] Wikipedia contributors. *Gottfried Wilhelm Leibniz*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Gottfried\\_Wilhelm\\_Leibniz](https://en.wikipedia.org/wiki/Gottfried_Wilhelm_Leibniz) (accessed August 2014).
  - [27] Wikipedia contributors. *James Prescott Joule*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/James\\_Prescott\\_Joule](https://en.wikipedia.org/wiki/James_Prescott_Joule) (accessed August 2014).
  - [28] Wikipedia contributors. *Johannes Kepler*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Johannes\\_Kepler](https://en.wikipedia.org/wiki/Johannes_Kepler) (accessed August 2014).
  - [29] Wikipedia contributors. *Linear elasticity*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Linear\\_elasticity](https://en.wikipedia.org/wiki/Linear_elasticity) (accessed August 2014).
  - [30] Wikipedia contributors. *Magnetohydrodynamics*. Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/wiki/Magnetohydrodynamics> (accessed August 2014).
  - [31] Wikipedia contributors. *Maxwell's equations*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Maxwell's\\_equations](https://en.wikipedia.org/wiki/Maxwell's_equations) (accessed August 2014).
  - [32] Wikipedia contributors. *Miletus*. Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/wiki/Miletus> (accessed August 2014).

- 
- [33] Wikipedia contributors. *Nicolaus Copernicus*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Nicolaus\\_Copernicus](https://en.wikipedia.org/wiki/Nicolaus_Copernicus) (accessed August 2014).
  - [34] Wikipedia contributors. *Ptolemy*. Wikipedia, The Free Encyclopedia. <https://en.wikipedia.org/wiki/Ptolemy> (accessed August 2014).
  - [35] Wikipedia contributors. *Sod's problem*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Sod\\_shock\\_tube](https://en.wikipedia.org/wiki/Sod_shock_tube) (accessed August 2014).
  - [36] Wikipedia contributors. *Thomas Young*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Thomas\\_Young\\_\(scientist\)](https://en.wikipedia.org/wiki/Thomas_Young_(scientist)) (accessed August 2014).
  - [37] Helen C. Yee, Neil D. Sandham, and M. J. Djomehri. Low-dissipative high-order shock-capturing methods using characteristic-based filters. *Journal of Computational Physics*, 150(1):199–238, 1999.

## Chapter 2

# Scalar conservation laws

Before pursuing the development and analysis of computational methods for conservation laws, we need to take a step back and consider the properties of the conservation laws and the nature of the solutions in a continuous setting. As we already experienced in Chapter 1, the spontaneous emergence of discontinuous solutions poses a number of challenges and, in particular, puts into question the notion of a solution in a classic sense. As we shall see shortly, a careful examination leads to the introduction of weak solutions, raising questions of existence and uniqueness of the solution.

The theory and analysis of conservation laws is a mature yet active and very rich area of research and we shall not attempt to be comprehensive in our treatment. Rather, we aim to provide enough background to understand the analysis of the numerical methods that will be developed in subsequent chapters. For a more thorough introduction to the mathematical analysis of conservation laws, we refer to [2, 8, 3, 7, 1, 11].

Let us consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad (x, t) \in \Omega_x \times \Omega_t, \quad (2.1)$$

where  $u = u(x, t)$ ,  $u(x, 0) = u_0(x)$ , and  $f(u) = f(u(x, t))$ . Provided  $u$  and  $f(u)$  are differentiable, we can express this as

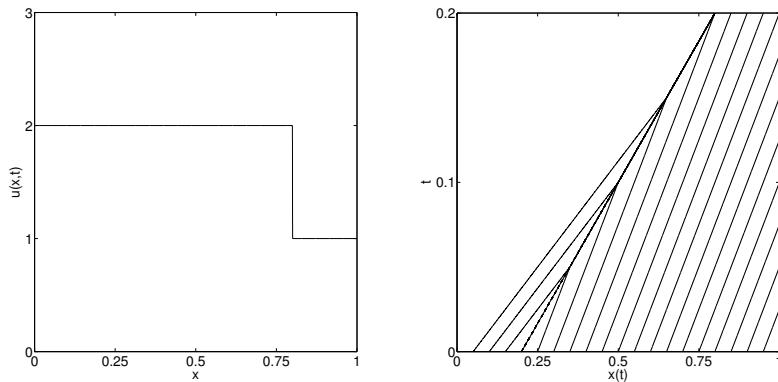
$$\frac{\partial u}{\partial t} + f'(u) \frac{\partial u}{\partial x} = 0, \quad (x, t) \in \Omega_x \times \Omega_t,$$

which we recognize as a wave problem with a wave speed of  $f'(u)$ . The characteristics  $x(t)$  are given by

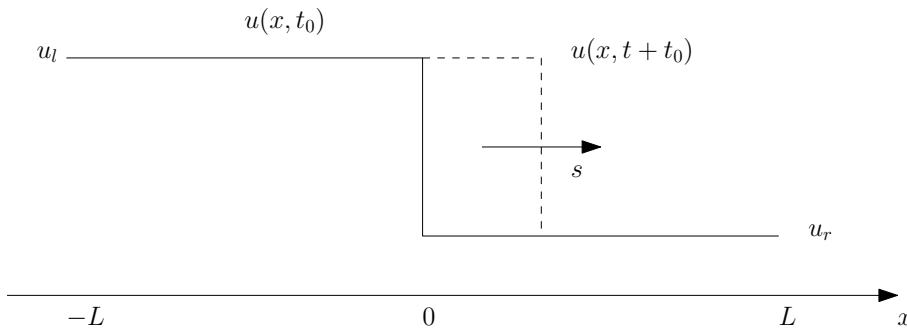
$$\frac{dx(t)}{dt} = f'(u),$$

with the initial conditions  $x(0) = x_0 \in \Omega_x$ . One easily finds that the solution is constant along these characteristics, i.e.,  $u(x, t) = u_0(x - f'(u)t)$ . Expressing this as  $u - u_0(x - f'(u)t) = 0$ , an application of the implicit function theorem yields [7]

$$\frac{\partial u}{\partial x} = \frac{-u'_0}{1 + u'_0 f''(u)t}, \quad \frac{\partial u}{\partial t} = \frac{u'_0 f'(u)}{1 + u'_0 f''(u)t}.$$



**Figure 2.1.** A propagating discontinuous solution to Burgers' equation (left) and the corresponding characteristics (right).



**Figure 2.2.** Sketch of the moving discontinuity, propagating at a constant speed  $s$ .

Hence, provided  $u'_0 f''(u) < 0$ , the derivative becomes unbounded as  $t$  increases. In particular if  $f(u)$  is convex, i.e.,  $f''(u) > 0$ , any initial condition with a negative gradient, i.e.,  $u'_0 < 0$ , will result in the formation of a discontinuity in finite time.

Once a discontinuity forms, the situation changes rather dramatically. The formation of the discontinuity results from the crossing of the characteristics, thereby eliminating our ability to fully understand the behavior of the solution by tracking the characteristics. However, once the discontinuity is formed and propagates, characteristics may again make sense and the discontinuity itself will propagate along a characteristic. To realize this, consider Burgers' equation with discontinuous initial conditions, as discussed in subsection 1.4.1. In Fig. 2.1 we show the solution at  $T = 0.2$  as well as the characteristics and observe that the characteristics run into the discontinuity which, however, propagates at a constant speed.

To recover the speed of the propagating discontinuity, consider the situation in Fig. 2.2. If we assume that the discontinuity is located at  $x = 0$ ,  $x \in [-L, L]$ , and we denote  $u_l$  and  $u_r$  as the left and right state of the solution, respectively, we have

$$\frac{d}{dt} \int_{-L}^L u(x, t) dx = f(u_l) - f(u_r),$$

from the conservation law itself. Furthermore, if we assume that the discontinuity propagates at the constant speed  $s$ , conservation of mass requires

$$\int_{-L}^L u(x, t) dx = (L + st)u_l + (L - st)u_r,$$

from which we recover

$$\frac{d}{dt} \int_{-L}^L u(x, t) dx = s(u_l - u_r) = f(u_l) - f(u_r).$$

An immediate consequence of this is

$$s[u] = [f], \quad [v] = v_l - v_r,$$

known as the Rankine–Hugoniot jump condition. We recover the speed of propagation for the discontinuity as

$$s = \frac{[f]}{[u]} = \frac{f(u_l) - f(u_r)}{u_l - u_r}. \quad (2.2)$$

This is a direct consequence of mass conservation and, thus, must be satisfied across any discontinuity. In a general case,  $u_l$  and  $u_r$  represent the left and right limiting solutions, respectively, as  $L$  approaches zero.

**Example 2.1.** Manipulating conservation laws with discontinuous solutions is a treacherous activity. To realize this, consider Burgers' equation [9]

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0. \quad (2.3)$$

Using the Rankine–Hugoniot condition, we recover the speed of propagation of a discontinuity

$$s = \frac{u_l^2 - u_r^2}{u_l - u_r} = u_l + u_r.$$

Provided the solution is smooth, we can multiply (2.3) with  $2u$  to recover the conservation law

$$\frac{\partial u^2}{\partial t} + \frac{4}{3} \frac{\partial u^3}{\partial x} = 0.$$

The Rankine–Hugoniot condition yields a speed of propagation as

$$s = \frac{4}{3} \frac{u_l^3 - u_r^3}{u_l^2 - u_r^2},$$

which, for a general discontinuous solution, predicts a different speed than for (2.3).

The lesson here is that manipulations that work well for problems with smooth solutions may fail once the solutions are discontinuous. ■

A more serious consequence of the formation of the discontinuity is the loss of the meaning of a classic derivative. To overcome this, let us first consider the modified Burgers' equation

$$\frac{\partial u_\varepsilon}{\partial t} + \frac{\partial u_\varepsilon^2}{\partial x} = \varepsilon \frac{\partial^2 u_\varepsilon}{\partial x^2},$$

where we assume that  $u_\varepsilon$  is smooth. It seems reasonable to conjecture that  $\|u - u_\varepsilon\|$  approaches zero in some sense as  $\varepsilon$  approaches zero. To obtain a qualitative understanding of this, consider

$$\frac{\partial u_\varepsilon}{\partial t} + a \frac{\partial u_\varepsilon}{\partial x} = \varepsilon \frac{\partial^2 u_\varepsilon}{\partial x^2}, \quad (2.4)$$

and let us seek traveling wave solutions on the form  $u_\varepsilon(x, t) = v_\varepsilon(x - at, t)$ . Inserting this into (2.4) yields

$$\frac{\partial v_\varepsilon}{\partial t} = \varepsilon \frac{\partial^2 v_\varepsilon}{\partial x^2},$$

which we recognize as the heat equation, for which it is well known that the solution is analytic for  $t > 0$  [4]. Hence,  $u_\varepsilon(x, t)$  is analytic and the classic solution exists. This approach, known as vanishing viscosity solutions, eliminates the challenges associated with the loss of the classic solution, but introduces the need to understand the convergence of  $\|u - u_\varepsilon\|$ .

## 2.1 • Weak solutions

Consider the original conservation law and let us recall that its native expression is the integral form

$$\int_{\Omega_x} u(x, t_2) dx - \int_{\Omega_x} u(x, t_1) dx = \int_{\Omega_t} f(x_1, t) - f(x_2, t) dt. \quad (2.5)$$

We now introduce a  $C^1$  test function  $\phi(x, t) : \Omega_x \times \Omega_t \rightarrow \mathbb{R}$  and assume that it is compactly supported, i.e.,  $\phi$  vanishes as  $x \rightarrow \partial \Omega_x$  and  $t \rightarrow T$ . Now consider

$$\int_{\Omega_x} \int_{\Omega_t} \left[ \frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} \right] \phi(x, t) dx dt = 0.$$

Integration by parts yields

$$\int_{\Omega_x} \int_{\Omega_t} \left[ u \frac{\partial \phi}{\partial t} + f(u) \frac{\partial \phi}{\partial x} \right] dx dt = - \int_{\Omega_x} u(x, 0) \phi(x, 0) dx. \quad (2.6)$$

Let us introduce the function

$$g(x; a, b, \varepsilon) = \begin{cases} 1, & a \leq x \leq b, \\ q_a(x), & a - \varepsilon \leq x \leq a, \\ q_b(x), & b \leq x \leq b + \varepsilon, \\ 0, & \text{otherwise,} \end{cases}$$

where  $0 \leq q_a(x) \leq 1$  connects zero and one in a smooth fashion over an interval of width  $\varepsilon$ , e.g.,  $q_a(x)$  could be a polynomial of sufficiently high order. The function  $q_b(x)$  is constructed similarly. If we now define the test function as

$$\phi(x, t) = g(x; x_1, x_2, \varepsilon) \times g(t; t_1, t_2, \varepsilon),$$

we see that  $\partial_x \phi = \partial_t \phi = 0$  for  $(x, t) = [x_1, x_2] \times [t_1, t_2] = \Omega_x \times \Omega_t$ . Furthermore, as  $\varepsilon$  approaches zero,  $\partial_x \phi(x_1)$  approaches  $\delta_{x_1}(x)$ , where  $\delta_{x_0}(x)$  is the Dirac delta function defined as

$$\delta_{x_0}(x) = \begin{cases} 1, & x = x_0, \\ 0, & x \neq x_0. \end{cases}$$

We obtain a similar result for the other intervals and recover

$$\int_{\Omega_x} u(x, t_2) dx - \int_{\Omega_x} u(x, t_1) dx + \int_{\Omega_t} f(x_2, t) - f(x_1, t) dt = - \int_{\Omega_x} u(x, 0) \phi(x, 0) dx, \quad (2.7)$$

which we recognize as the conservation law in integral form (2.5), with the exception of the term on the right-hand side. However, this term accounts for the initial condition which is not included in (2.5), which only measures change. Nevertheless, the connection between solutions to (2.5) and (2.6) is clear and we shall refer to  $u(x, t)$  as a weak solution to the conservation law (2.1) provided it satisfies (2.6) for all admissible test functions. It follows immediately that a weak solution is also a solution to the conservation law as it is recovered for a special test function.

With the introduction of the weak solution, we overcome the problem associated with the loss of the classic derivative once a discontinuity forms. Unfortunately, as illustrated in the following example, this approach introduces a new problem.

**Example 2.2.** Consider Burgers' equation,

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

with the initial condition

$$u_0(x) = \begin{cases} 1, & x < 0.2, \\ 2, & x \geq 0.2. \end{cases}$$

We recognize that a weak solution is given by  $u(x, t) = u_0(x - 3t)$  by the Rankine–Hugoniot condition (2.2).

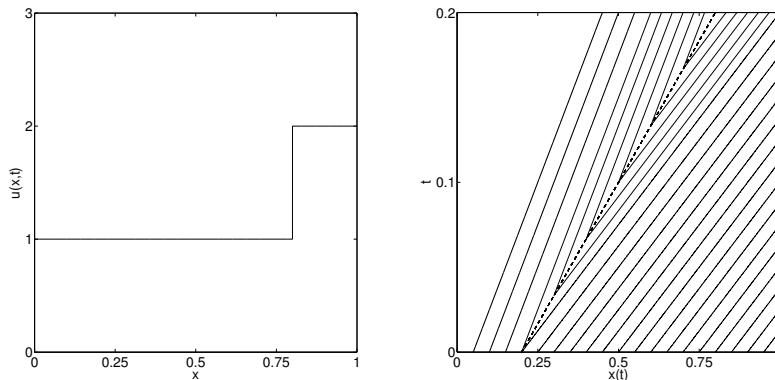
However, the problem also has a classic solution for  $t > 0$  of the form [9]

$$u(x, t) = \begin{cases} 1, & x < 0.2 + 2t, \\ (x - 0.2)/2t, & 0.2 + 2t < x < 0.2 + 4t, \\ 2, & x > 0.2 + 4t. \end{cases} \quad (2.8)$$

Hence, an unfortunate consequence of the introduction of the notion of weak solutions is the loss of uniqueness of the solution. The situation is, in fact, worse, since any solution of the type

$$u(x, t) = \begin{cases} 1, & x < 0.2 + 2st, \\ u_m, & 0.2 + 2st < x < 0.2 + 2tu_m, \\ (x - 0.2)/2t, & 0.2 + 2tu_m < x < 0.2 + 4t, \\ 2, & x > 0.2 + 4t, \end{cases}$$

is a weak solution to Burgers' equation. Here  $1 < u_m < 2$  and  $s$  is the speed of the discontinuity between 1 and  $u_m$ .



**Figure 2.3.** The solution of a propagating weak solution to Burgers' equation with  $u_r > u_l$  (left) and the corresponding characteristics (right).

This raises the natural questions of which solution, among this infinite family of candidates, is the right one and what criteria should we use to identify the unique solution. To gain some insight into this question, we show in Fig. 2.3 the traveling wave solution,  $u_0(x - 3t)$ , as well as the associated characteristics for a propagating discontinuity with  $u_r > u_l$ .

Comparing with Fig. 2.1, we observe essential differences in the nature of the characteristics. In particular, we see in Fig. 2.1, that all characteristics propagate into the discontinuity, whereas in Fig. 2.3 characteristics emerge from the discontinuity. One immediate consequence of this latter situation is that there is no direct relationship between the characteristics and the initial conditions. Furthermore, the emergence of the characteristics from the discontinuity suggests that even minor changes or perturbations of the propagating discontinuity will alter the dynamics and, ultimately, the solution. This is clearly an unstable solution and this observation suggests that this is an unlikely solution candidate.

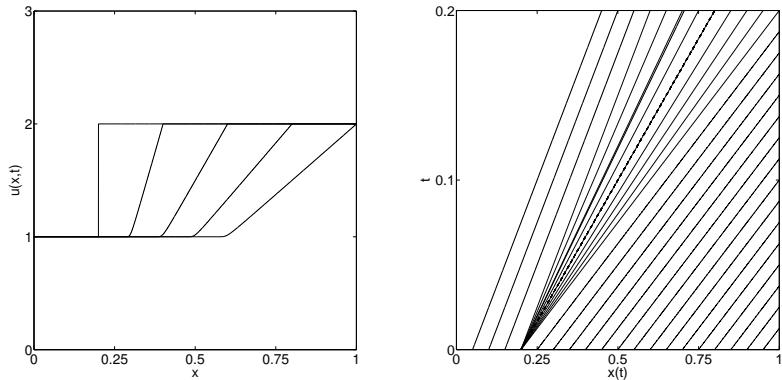
If we consider the classic solution in Fig. 2.4 and the associated characteristics for the solution (2.8), we observe that the characteristic lines create a continuous fan that connects the two states and leaves no ambiguity of where each characteristic originates.

Based on this qualitative insight, it is tempting to conclude that all weak solutions, containing a propagating discontinuity with  $u_l < u_r$ , should be eliminated and only the classic solution, (2.8), known as a rarefaction wave, should be retained as a admissible solution. As we shall see shortly, that is a correct interpretation and extends to more general cases. ■

## 2.2 • Entropy conditions and solutions

To overcome the issue of nonuniqueness of the weak solution, we need criteria to determine whether a solution is admissible or not and, furthermore, we need to show that such criteria are sufficient to identify a unique physically relevant solution. We refer to such a unique solution as the entropy solution.

If we return to Ex. 2.2 and recall the nature of the solution shown in Fig. 2.1, we recognized a distinct difference between the characteristics of the two discontinuous



**Figure 2.4.** The solution of a propagating classic solution to Burgers' equation (left) and the corresponding characteristics (right).

solutions. When  $u_l > u_r$ , a characteristic can be traced backward in time to the initial condition. In contrast to this, the result in Fig. 2.3 shows that we can no longer associate a characteristic to the initial condition. This suggests that this latter solution cannot be a physical solution.

Based on this insight, a natural condition of admissibility—an entropy condition—is [7]

$$\forall u \in (u_l, u_r) : \frac{f(u) - f(u_l)}{u - u_l} \geq s \geq \frac{f(u) - f(u_r)}{u - u_r}, \quad (2.9)$$

reflecting that the characteristics must run into the shock. This condition is often known as the Lax entropy condition. The notion of entropy is borrowed from thermodynamics, where this condition defines a direction of time by requiring that the entropy can only increase. While the convention for entropy in the context of conservation laws is that the entropy can only decrease, it likewise defines a direction of time. Comparing the characteristics of the physical solution in Fig. 2.1 with those in Fig. 2.3 of a nonphysical solution, we observe a similarity between the two if we reverse the direction of time in Fig. 2.3. We shall see shortly that the requirement of a decreasing entropy is a powerful criterion for selecting the unique solution.

If we, furthermore, simplify matters and assume that  $f(u)$  is convex, the shock speed is

$$\begin{aligned} s &\leq \frac{f(u) - f(u_l)}{u - u_l} = \frac{f(u_l) + f'(u_l)(u - u_l) + \frac{1}{2}f''(\xi)(u - u_l)^2 - f(u_l)}{u - u_l} \\ &= f'(u_l) + \frac{1}{2}f''(\xi)(u - u_l). \end{aligned}$$

Thus,  $s < f'(u_l)$  provided  $u_l > u_r$ . Similarly, for  $u_r$ , we recover  $s > f'(u_r)$ . For the solution in Fig. 2.3, the signs reverse since  $u_l < u_r$ , and (2.9) is violated. This results in the simplified Lax entropy condition

$$f'(u_l) > s > f'(u_r), \quad f''(u) > 0. \quad (2.10)$$

Going forward, we define a shock as a discontinuity that satisfies the Rankine–Hugoniot condition, (2.2), and an entropy condition.

A slightly different form of the entropy condition, known as the Oleinik E-condition [10] and limited to the case of a convex flux, requires

$$\forall \varepsilon > 0: \frac{u(x + \varepsilon, t) - u(x, t)}{\varepsilon} \leq \frac{E}{t}, \quad E = \frac{1}{\inf f''}, \quad t > 0, \quad (2.11)$$

where  $E > 0$  by convexity. Admissibility requires  $u_l > u_r$ , consistent with (2.10). Furthermore, this condition also contains information about the spreading of the rarefaction wave, shown in Fig. (2.4). The proof of (2.11) can be found in [11], where it is also shown that this condition is equivalent to (2.10).

If we define the total variation  $TV(u)$  of the solution

$$TV(u(t)) = \sup_{\varepsilon} \int_{\Omega_x} \left| \frac{u(x + \varepsilon, t) - u(x, t)}{\varepsilon} \right| dx,$$

(2.11) implies that

$$TV(u(t)) \leq TV(u(0)).$$

This states that the total variation is decaying for an entropy solution of a scalar conservation law with a convex flux. We shall refer to this property as total variation diminishing.

While this offers a path to identify admissible solutions, it leaves open the question of uniqueness, as there could be more than one admissible solution. To settle this, we consider the following result [7].

**Theorem 2.3.** *Let  $u$  and  $v$  be piecewise smooth weak solutions to (2.1) with a convex flux and assume that all discontinuities are shocks. Then*

$$\frac{d}{dt} \|u(t) - v(t)\|_1 \leq 0,$$

where  $\|\cdot\|_1$  signifies the  $L^1$ -norm in space.

**Proof.** We begin by writing the  $L^1(\Omega_x)$ -norm as

$$\|u(t) - v(t)\|_1 = \int_{\Omega_x} |u(x, t) - v(x, t)| dx = \sum_k (-1)^k \int_{x^k}^{x^{k+1}} (u(x, t) - v(x, t)) dx, \quad (2.12)$$

where we have introduced the nonoverlapping intervals  $[x^k(t), x^{k+1}(t)]$  such that  $u(x, t) - v(x, t)$  maintains a constant sign. We enumerate these intervals such that the sign of  $u(x, t) - v(x, t)$  in  $[x^k(t), x^{k+1}(t)]$  is  $(-1)^k$ . Since the solutions are time-dependent, so is the segmentation  $[x^k(t), x^{k+1}(t)]$ .

We differentiate (2.12) with respect to time, to recover

$$\begin{aligned} \frac{d}{dt} \|u(t) - v(t)\|_1 &= \sum_k (-1)^k \left[ \int_{x^k}^{x^{k+1}} \frac{\partial}{\partial t} (u(x, t) - v(x, t)) dx \right. \\ &\quad \left. + (u(x^{k+1}, t) - v(x^{k+1}, t)) \frac{dx^{k+1}}{dt} - (u(x^k, t) - v(x^k, t)) \frac{dx^k}{dt} \right]. \end{aligned}$$

Since both  $u$  and  $v$  satisfy (2.1) in a weak sense, we recover

$$\frac{d}{dt} \|u(t) - v(t)\|_1 = \sum_k (-1)^k \left[ (f(v) - f(u)) + (u - v) \frac{dx}{dt} \right]_{x^k}^{x^{k+1}}. \quad (2.13)$$

Let us now consider the contribution at an interface  $x^{k+1}(t)$ . If we first assume that  $u$  and  $v$  are continuous at  $x^{k+1}$ , the contribution vanishes. Now consider the case where  $u$  has a shock at  $x^{k+1}$ . Since the shock satisfies the entropy condition, we must have  $u_l > u_r$ . Suppose also that  $u_l > v > u_r$  so that  $u_l - v > 0$  and  $k$  must be even. Since the shock propagation satisfies the Rankine–Hugoniot condition, we recover

$$\frac{dx^{k+1}}{dt} = s = \frac{f(u_l) - f(u_r)}{u_l - u_r}.$$

Considering the contributions at  $x^{k+1}$  from (2.13) yields

$$f(v) - f(u_l) + (u_l - v) \frac{f(u_l) - f(u_r)}{u_l - u_r} = f(v) - \left[ \frac{v - u_r}{u_l - u_r} f(u_l) + \frac{u_l - v}{u_l - u_r} f(u_r) \right]. \quad (2.14)$$

Since  $f$  is convex, Jensen's inequality yields

$$f(v) = f\left(\frac{v - u_r}{u_l - u_r} u_l + \frac{u_l - v}{u_l - u_r} u_r\right) \leq \left[ \frac{v - u_r}{u_l - u_r} f(u_l) + \frac{u_l - v}{u_l - u_r} f(u_r) \right],$$

and the contribution (2.14) is negative.

The same approach can be used for a shock at  $x^k$ , for cases where  $v$  is outside the interval of  $[u_l, u_r]$ , or for the situation where the shock is assumed to be in  $v$ . This concludes the proof.  $\square$

The property that

$$\frac{d}{dt} \|u(t) - v(t)\|_1 \leq 0$$

is known as  $L^1$ -contraction and it has a number of immediate consequences.

**Corollary 2.4.** *If  $u$  is a weak solution to (2.1) with a convex flux and it satisfies an entropy condition, the solution is unique.*

*Proof.* It follows directly from the result by taking  $u(0) = v(0)$ .  $\square$

**Corollary 2.5.** *If a discontinuity violates the entropy condition, then there is a solution,  $v$ , such that*

$$\frac{d}{dt} \|u(t) - v(t)\|_1 \geq 0.$$

While the combination of the entropy condition and the Rankine–Hugoniot condition guarantees uniqueness of the solution, it remains unclear whether such solutions

may result from general initial conditions  $u_0(x)$ , i.e., is the entropy condition too strong? To settle this, recall the modified problem

$$\frac{\partial u_\varepsilon}{\partial t} + \frac{\partial f(u_\varepsilon)}{\partial x} = \varepsilon \frac{\partial^2 u_\varepsilon}{\partial x^2}, \quad u_\varepsilon(x, 0) = u_0(x), \quad (2.15)$$

where  $u_\varepsilon$  is the vanishing viscosity solution to the conservation law (2.1). The answer to the question is found in the following result [7].

**Theorem 2.6.** *Assume that the flux,  $f(u)$ , is convex. In the limit of  $\varepsilon \rightarrow 0$ , the solution to (2.15) is a weak solution to (2.1) and satisfies the entropy condition.*

**Proof.** Since (2.15) is a parabolic problem for  $\varepsilon > 0$ , standard results guarantee that a unique smooth solution exists for  $t > 0$ . We consider

$$\int_{\Omega_t} \int_{\Omega_x} \left[ \frac{\partial u_\varepsilon}{\partial t} + \frac{\partial f(u_\varepsilon)}{\partial x} \right] \phi(x, t) dx dt = \int_{\Omega_t} \int_{\Omega_x} \varepsilon \frac{\partial^2 u_\varepsilon}{\partial x^2} \phi(x, t) dx dt,$$

where  $\phi(x, t)$  is a smooth test function, compactly supported on  $\Omega_x \times \Omega_t$ . Integration by parts yields

$$\int_{\Omega_t} \int_{\Omega_x} \left[ \frac{\partial \phi}{\partial t} u_\varepsilon + f(u_\varepsilon) \frac{\partial \phi}{\partial x} \right] dx dt = -\varepsilon \int_{\Omega_t} \int_{\Omega_x} \frac{\partial^2 \phi}{\partial x^2} u_\varepsilon dx dt.$$

Since  $\phi$  is smooth, the right-hand side vanishes as  $\varepsilon$  approaches zero and we recover

$$\int_{\Omega_t} \int_{\Omega_x} \left[ \frac{\partial \phi}{\partial t} u + f(u) \frac{\partial \phi}{\partial x} \right] dx dt = 0,$$

confirming that the limiting solution is a weak solution to the conservation law.

Now consider two viscosity solutions,  $u_\varepsilon$  and  $v_\varepsilon$ , and write the  $L^1$ -norm as

$$\|u_\varepsilon(t) - v_\varepsilon(t)\|_1 = \int_{\Omega_x} |u_\varepsilon(x, t) - v_\varepsilon(x, t)| dx = \sum_k (-1)^k \int_{x^k}^{x^{k+1}} (u_\varepsilon(x, t) - v_\varepsilon(x, t)) dx,$$

with the notation borrowed from the proof of Theorem 2.3. Since  $u_\varepsilon$  are  $v_\varepsilon$  are viscosity solutions, they are continuous and  $u_\varepsilon = v_\varepsilon$  at all  $x^k$ .

Proceeding as in the proof of Theorem 2.3 we differentiate the above expression to obtain

$$\begin{aligned} \frac{d}{dt} \|u_\varepsilon(t) - v_\varepsilon(t)\|_1 \\ = \sum_k (-1)^k \left[ \int_{x^k}^{x^{k+1}} \frac{\partial}{\partial t} (u_\varepsilon(x, t) - v_\varepsilon(x, t)) dx + \left[ (u_\varepsilon(x, t) - v_\varepsilon(x, t)) \frac{dx(t)}{dt} \right]_{x^k}^{x^{k+1}} \right]. \end{aligned}$$

Using the equation itself yields

$$\begin{aligned} \frac{d}{dt} \|u_\varepsilon(t) - v_\varepsilon(t)\|_1 &= \sum_k (-1)^k \left[ \varepsilon \frac{\partial}{\partial x} (u_\varepsilon(x, t) - v_\varepsilon(x, t)) - (f(u_\varepsilon) - f(v_\varepsilon)) \right. \\ &\quad \left. + (u_\varepsilon(x, t) - v_\varepsilon(x, t)) \frac{dx(t)}{dt} \right] \Big|_{x^k}^{x^{k+1}} \\ &= \sum_k (-1)^k \left[ \varepsilon \frac{\partial}{\partial x} (u_\varepsilon(x, t) - v_\varepsilon(x, t)) \right] \Big|_{x^k}^{x^{k+1}}, \end{aligned}$$

where the last reduction follows by continuity of the viscosity solution. Now consider each term of the type

$$(-1)^k (u_\varepsilon(x, t) - v_\varepsilon(x, t)) \geq 0, \quad x \in [x^k, x^{k+1}],$$

and zero at  $x^k$  and  $x^{k+1}$  since  $u_\varepsilon = v_\varepsilon$  at the interfaces by construction. If we assume that  $k$  is even, then the gradient of  $u_\varepsilon - v_\varepsilon$  at  $x^k$  must be nonnegative and, likewise, nonpositive at  $x^{k+1}$ . Hence,

$$(-1)^k \left[ \varepsilon \frac{\partial}{\partial x} (u_\varepsilon(t) - v_\varepsilon(t)) \right] \Big|_{x^k}^{x^{k+1}} \leq 0$$

and, thus,

$$\frac{d}{dt} \|u(t) - v(t)\|_1 \leq 0.$$

Corollary 2.5 guarantees that  $u_\varepsilon$  satisfies the entropy condition, hence completing the proof.  $\square$

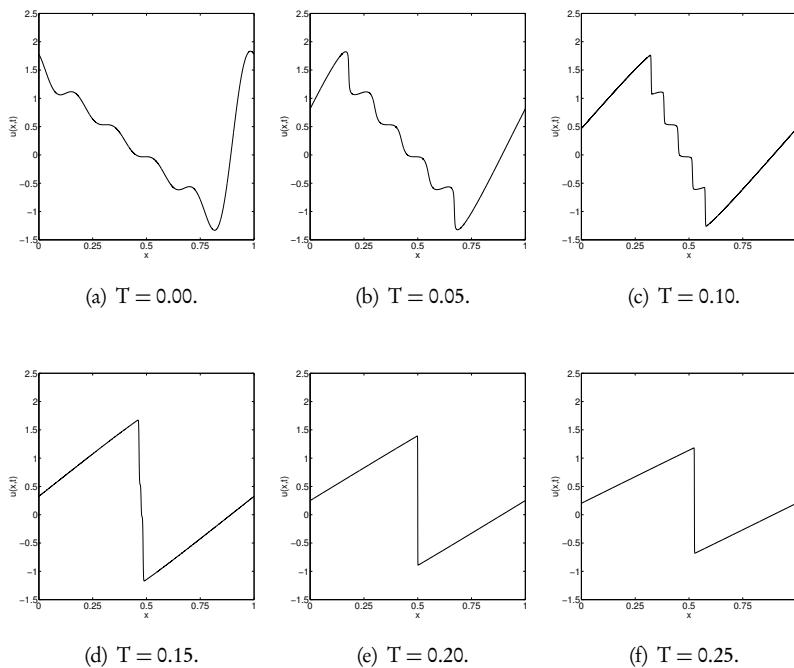
This result implies that if  $u$  is a vanishing viscosity solution in the  $L^1$ -sense and  $v$  is a genuine solution to (2.1), then  $\|u(t) - v(t)\|_1$  is decreasing in time and, thus,  $u$  satisfies the entropy condition. In other words, the vanishing viscosity solution is precisely the unique solution to the conservation law. Since the viscosity solution is smooth and unique for any initial condition  $u_0(x)$ , this settles the concern about whether the entropy conditions are too strong to allow genuine solutions.

**Example 2.7.** We consider Burgers' equation with the initial condition

$$u(x, 0) = 0.5 + \sum_{n=1}^5 \frac{(-1)^n}{n} \sin(2\pi n(x - 0.4)), \quad x \in [0, 1],$$

and periodic boundary conditions.

In Fig. 2.5 we show the solution during  $t \in [0, 0.25]$  at regular intervals. The initial condition breaks into entropy satisfying waves that propagate and finally combine into one rightward propagating shock wave. The general solution is called an  $N$ -wave [9]. It is clear that the requirement that all characteristics run into the shock eliminates any chance of being able to integrate backwards in time to recover the initial condition. Once a shock forms, information is lost and the evolution is irreversible or, with a more physical interpretation, the entropy has increased.  $\blacksquare$



**Figure 2.5.** The solution to Burgers' equation with a multimodal initial condition, shown in the upper left corner.

The key results in much of the above discussion rely on the assumption of a convex flux, thus limiting their validity to scalar conservation laws in one spatial dimension. Nevertheless, these same results also hold under the more general entropy condition (2.9), thereby extending it to scalar conservation laws in multiple dimensions [11].

## 2.3 • Entropy functions

While the uniqueness results of the previous section extend to scalar problems in multiple dimensions, the extension to systems of equations requires a different approach. To keep things simple, let us initiate this development with the scalar problem, as that offers some additional insight. We return to the case of systems in the next chapter.

Let us define an entropy function  $\eta(u)$  and the associated entropy flux  $\psi(u)$  and require that they satisfy

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} = 0, \quad (2.16)$$

if  $u$  is smooth. We refer to  $(\eta, \psi)$  satisfying (2.16) as an entropy pair. For reasons that will soon become clear, we assume that  $\eta(u)$  is convex. Let us recall the following result [9].

**Theorem 2.8.** For the entropy pair  $(\eta, \psi)$  with  $\psi' = f' \eta'$ ,  $\eta$  being convex and  $u$  being a weak entropy solution to (2.1), it holds that

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0$$

in a weak sense.

**Proof.** If we first restrict ourselves to the case where  $u$  is smooth, we recover from (2.16)

$$\eta' \frac{\partial u}{\partial t} + \psi' \frac{\partial u}{\partial x} = 0.$$

Since  $\psi'(u) = \eta'(u)f'(u)$ , it follows that  $\eta(u)$  is conserved for  $u$  satisfying (2.1).

For the case where  $u$  lacks smoothness, this same approach is not possible. Let us therefore consider the vanishing viscosity solution, satisfying (2.15). Multiply (2.15) with  $\eta'$  to recover

$$\frac{\partial \eta(u_\varepsilon)}{\partial t} + \frac{\partial \psi(u_\varepsilon)}{\partial x} = \varepsilon \eta'(u_\varepsilon) \frac{\partial^2 u_\varepsilon}{\partial x^2}.$$

Now observe that

$$\eta'(u_\varepsilon) \frac{\partial^2 u_\varepsilon}{\partial x^2} = \frac{\partial}{\partial x} \left( \eta'(u_\varepsilon) \frac{\partial u_\varepsilon}{\partial x} \right) - \eta''(u_\varepsilon) \left( \frac{\partial u_\varepsilon}{\partial x} \right)^2,$$

to obtain

$$\frac{\partial \eta(u_\varepsilon)}{\partial t} + \frac{\partial \psi(u_\varepsilon)}{\partial x} = \varepsilon \left[ \frac{\partial}{\partial x} \left( \eta'(u_\varepsilon) \frac{\partial u_\varepsilon}{\partial x} \right) - \eta''(u_\varepsilon) \left( \frac{\partial u_\varepsilon}{\partial x} \right)^2 \right].$$

Integrating over  $\Omega_x \times \Omega_t$ , we obtain

$$\begin{aligned} & \int_{\Omega_x} \int_{\Omega_t} \frac{\partial \eta(u_\varepsilon)}{\partial t} + \frac{\partial \psi(u_\varepsilon)}{\partial x} dx dt \\ &= \varepsilon \int_{\Omega_x} \int_{\Omega_t} \left[ \frac{\partial}{\partial x} \left( \eta'(u_\varepsilon) \frac{\partial u_\varepsilon}{\partial x} \right) - \eta''(u_\varepsilon) \left( \frac{\partial u_\varepsilon}{\partial x} \right)^2 \right] dx dt. \end{aligned}$$

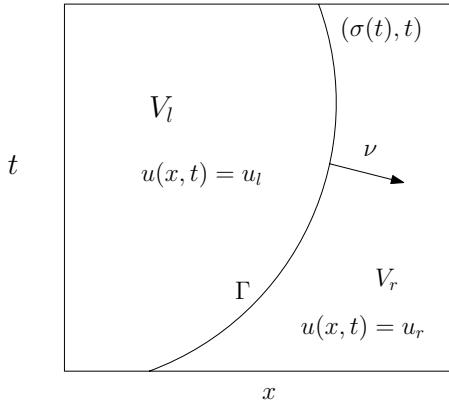
The first term on the right-hand side,

$$\varepsilon \int_{\Omega_t} \left( \eta'(u_\varepsilon) \frac{\partial u_\varepsilon}{\partial x} \right) \Big|_{x_1}^{x_2} dt,$$

vanishes as  $\varepsilon$  tends to zero. While we cannot bound the second term, the assumption that  $\eta(u)$  is convex implies

$$\int_{\Omega_x} \int_{\Omega_t} \frac{\partial \eta(u_\varepsilon)}{\partial t} + \frac{\partial \psi(u_\varepsilon)}{\partial x} dx dt \leq 0,$$

provided that  $u$  is the limit of  $u_\varepsilon$  in an  $L^1$ -sense. However, this was established in Theorem 2.6.  $\square$



**Figure 2.6.** Sketch of the piecewise smooth solution and the notation used in the proof of Theorem 2.9.

One can state a similar result in a weak sense as

$$\int_{\Omega_x} \int_{\Omega_t} -\eta(u) \frac{\partial \phi}{\partial t} - \psi(u) \frac{\partial \phi}{\partial x} dx dt \leq - \int_{\Omega_x} \eta(u(x, 0)) \phi(x, 0) dx, \quad (2.17)$$

as for the weak solution to the conservation law (2.7), with  $\phi$  being a smooth test function, compactly supported on  $\Omega_x \times \Omega_t$ .

Whereas this establishes Theorem 2.8 provided  $u$  is a weak entropy solution, the converse is equally important, i.e., if

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0, \quad (2.18)$$

then  $u$  is an entropy solution. Such a result would position (2.18) as an equivalent and more general entropy condition. This is established as follows [5].

**Theorem 2.9.** *Assume that  $u$  is a weak solution to (2.1) with a convex flux. Assume also that we have an entropy pair  $(\eta, \psi)$  with  $\psi' = f' \eta'$ , and  $\eta$  is convex such that it holds*

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0$$

*in a weak sense. Then the solution is a weak entropy solution satisfying (2.10).*

**Proof.** Consider (2.17)

$$\int_{\Omega_x} \int_{\Omega_t} -\eta(u) \frac{\partial \phi}{\partial t} - \psi(u) \frac{\partial \phi}{\partial x} dx dt \leq 0, \quad (2.19)$$

where we have neglected the initial conditions for simplicity and without loss of generality.

To assist in the proof, consider the sketch of the problem in Fig. 2.6. Assume that  $\Omega_x = [x_1, x_2]$  and consider a piecewise constant solution with a discontinuity located

along  $(\sigma(t), t)$  with  $x_1 < \sigma(t) < x_2$ . We rewrite (2.19) as

$$\int_{V_l} -\eta(u) \frac{\partial \phi}{\partial t} - \psi(u) \frac{\partial \phi}{\partial x} dx dt + \int_{V_r} -\eta(u) \frac{\partial \phi}{\partial t} - \psi(u) \frac{\partial \phi}{\partial x} dx dt \leq 0, \quad (2.20)$$

where  $V_l$  and  $V_r$  represent the two volumes with a constant solution, separated by  $\sigma(t)$ .

Since we assume  $u$  to be piecewise constant in  $V_l$  and  $V_r$ , respectively, we can define the constant vectors

$$\mathbf{e}_l = \begin{bmatrix} \psi(u_l) \\ \eta(u_l) \end{bmatrix}, \quad \mathbf{e}_r = \begin{bmatrix} \psi(u_r) \\ \eta(u_r) \end{bmatrix}.$$

Defining  $\nabla = \left( \frac{\partial}{\partial x}, \frac{\partial}{\partial t} \right)$ , we express (2.20) as

$$\int_{V_l} -\nabla \cdot (\phi \mathbf{e}_l) dx dt + \int_{V_r} -\nabla \cdot (\phi \mathbf{e}_r) dx dt \leq 0.$$

Since (2.19) is an equality when the solution is continuous, Gauss' theorem yields

$$\int_{\Gamma} -\mathbf{v} \cdot [\phi(\mathbf{e}_l - \mathbf{e}_r)] ds \leq 0, \quad (2.21)$$

where  $\Gamma$  is the trajectory  $(\sigma(t), t)$  of the discontinuity and  $\mathbf{v}$  is the normal vector pointing out of  $V_l$ . Since  $\Gamma$  is given by  $(\sigma(t), t)$ , its tangent vector is  $(\sigma'(t), 1)$ , and thus the normal vector is  $\mathbf{v} = C[1, -\sigma'(t)]$ , where  $C = \sqrt{1 + \sigma'(t)^2} > 0$ . However,  $\sigma'(t)$  is the speed of the discontinuity which, by the Rankine–Hugoniot condition, is equal to the speed  $s$ .

This allows us to write (2.21) as

$$\int_{\Gamma} C [(\eta(u_l) - \eta(u_r))s - (\psi(u_l) - \psi(u_r))] \phi(x, t) ds \leq 0. \quad (2.22)$$

Let us now define

$$s(u) = \frac{f(u) - f(u_r)}{u - u_r},$$

where we recognize  $s(u_l)$  as the shock speed, and

$$E(u) := (\eta(u) - \eta(u_r))s(u) - (\psi(u) - \psi(u_r)). \quad (2.23)$$

Clearly  $E(u_r) = 0$  since  $s(u_r) = f'(u_r)$  is bounded, and (2.22) requires that  $E(u_l) \leq 0$ . Consider  $E'(u)$  as

$$\begin{aligned} E'(u) &= s'(u)(\eta(u) - \eta(u_r)) + s(u)\eta'(u) - \psi'(u) \\ &= s'(u)(\eta(u) - \eta(u_r)) + \eta'(u)(s(u) - f'(u)). \end{aligned}$$

From the definition of  $s(u)$  we obtain

$$s'(u) = \frac{(u - u_r)f'(u) - (f(u) - f(u_r))}{(u - u_r)^2} = \frac{1}{u - u_r} (f'(u) - s(u)),$$

which, inserted in the previous expression to eliminate  $(s(u) - f'(u))$ , yields

$$E'(u) = s'(u)(\eta(u) - \eta(u_r)) - s'(u)\eta'(u)(u - u_r).$$

By the mean value theorem, we have  $s'(u) = \alpha f''(\xi)$  for  $\xi \in [u_l, u_r]$ . Since  $f$  is convex, this implies that  $s'(u)$  is bounded. It thus follows that  $E'(u_r) = 0$ .

Assuming  $\xi_1 \in [u, u_r]$ , the mean value theorem again yields

$$E'(u) = s'(u)\eta'(\xi_1)(u - u_r) - s'(u)\eta'(u)(u - u_r) = s'(u)(u - u_r)(\eta'(\xi_1) - \eta'(u)).$$

Repeating, we recover for  $\xi_2 \in [\xi_1, u]$

$$E'(u) = s'(u)(u - u_r)\eta''(\xi_2)(\xi_2 - u).$$

Since  $\eta(u)$  is assumed to be convex,  $\eta''(u) > 0$ .

Now assume  $u < u_r$  such that  $u \leq \xi_1 \leq u_r$ ,  $\xi_2 \geq u$ , which implies that  $E'(u) < 0$ . For the converse,  $u_r < u$ , the same holds. Hence  $E'(u) \leq 0$ .

We now have a total of four conditions:

$$E(u_r) = 0, \quad E'(u_r) = 0, \quad E(u_l) \leq 0, \quad E'(u) \leq 0.$$

It is easy to see that this is possible only when  $u_l \geq u_r$  which subsequently implies

$$f'(u_r) = s(u_r) < s(u) < f'(u_l),$$

since  $s'(u) > 0$ .  $\square$

An immediate consequence of this result is the following corollary.

**Corollary 2.10.** *If  $u$  is a weak entropy solution to a conservation law, endowed with an entropy pair  $(\eta, \psi)$  and a convex flux,  $f$ , then*

$$s(u) \leq \frac{\psi(u) - \psi(u_r)}{\eta(u) - \eta(u_r)},$$

where  $u \in [u_l, u_r]$  and  $u_l > u_r$ .

**Example 2.11.** Let us again consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

and define the entropy  $\eta(u) = u^2$ , which yields the entropy function,  $\psi = \frac{4}{3}u^3$ . For  $u$  being continuous, it is easy to see that

$$\frac{\partial \eta}{\partial t} + \frac{\partial \psi}{\partial x} = 0.$$

To establish the entropy inequality we need to show that  $E(u) \leq 0$  for  $u_l > u > u_r$ . Inserting into (2.23) and using the definition of  $s(u)$ , we recover

$$E(u) = -\frac{1}{3}(u - u_r)^3 < 0,$$

since  $u > u_r$ .  $\blacksquare$

We shall find it useful to introduce a special entropy pair, known as the Kružkov entropy pair, defined as

$$\eta(u) = |u - c|, \quad \psi(u) = \text{sign}(u - c)(f(u) - f(c)),$$

where  $c \in \mathbb{R}$  is an arbitrary real constant [6]. We note that  $\eta''(u) = 2\delta_0(u - c) \geq 0$ , hence satisfying the assumption that the entropy is convex. We have the following result [5].

**Theorem 2.12.** *Let  $u$  be a weak solution to the conservation law and assume that  $u$  satisfies Theorem 2.8 for all entropy pairs  $(\eta, \psi)$ . Then the Kružkov entropy pair satisfies*

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0$$

in a weak sense for all  $c \in \mathbb{R}$ .

**Proof.** Define a regularized entropy

$$\eta_\varepsilon(u) = \varepsilon \eta\left(\frac{u - c}{\varepsilon}\right),$$

where  $\eta(u)$  is a convex entropy such that  $\eta(u) = |u|$  for  $|u| \geq 1$ . The associated entropy flux is defined by  $\psi'_\varepsilon(u) = f'(u)\eta'_\varepsilon(u)$ , which we assume satisfies

$$\frac{\partial \eta_\varepsilon}{\partial t} + \frac{\partial \psi_\varepsilon}{\partial x} \leq 0.$$

Observe that we have the pointwise limits

$$\lim_{\varepsilon \rightarrow 0} \eta_\varepsilon(u) = |u - c|, \quad \lim_{\varepsilon \rightarrow 0} \psi'_\varepsilon(u) = \text{sign}(u - c).$$

We recover the entropy flux as

$$\lim_{\varepsilon \rightarrow 0} \psi_\varepsilon(u) = \lim_{\varepsilon \rightarrow 0} \int_c^u \eta'_\varepsilon(x) f'(x) dx = \text{sign}(u - c)(f(u) - f(c)) = \psi(u),$$

establishing that the Kružkov pair satisfies Theorem 2.8.  $\square$

This shows that if the entropy condition is satisfied for general entropy pairs, it is also satisfied for the Kružkov pair. The reverse, i.e., if the solution satisfies the entropy condition for the Kružkov pair it also satisfies the entropy condition for a general entropy pair, is established in the following [5].

**Theorem 2.13.** *Assume that Theorem 2.12 holds for the weak solution  $u$  for all  $c \in \mathbb{R}$ . Then Theorem 2.8 holds for any entropy pair  $(\eta, \psi)$ .*

**Proof.** Given the convex entropy,  $\eta(u) \in C^2([a, b])$ , we define  $k_i = a + h_m i$ ,  $i = 0, \dots, m + 1$ , and  $h_m = (b - a)/(m + 1)$ , and express the linear interpolation of  $\eta(u)$  as

$$\eta_m(u) = \beta_0(u - c) + \sum_{i=1}^m \beta_i |u - k_i|,$$

where  $c \in \mathbb{R}$ . If we consider  $u \in [k_j, k_{j+1}]$  and separate positive and negative contributions, we obtain

$$\eta_m(u) = \left( \sum_{i=0}^j \beta_i - \sum_{i=j+1}^m \beta_i \right) u - \left( \beta_0 c + \sum_{i=1}^j \beta_i k_i - \sum_{i=j+1}^m \beta_i k_i \right).$$

We now require that

$$\eta'(u) = \eta'_m(u) = \alpha_j, \quad u \in [k_j, k_{j+1}],$$

to recover

$$\forall j = 0, \dots, m : \sum_{i=0}^j \beta_i - \sum_{i=j+1}^m \beta_i = \alpha_j.$$

Considering this expression backwards, we have

$$\sum_{i=0}^m \beta_i = \alpha_m, \quad \sum_{i=0}^{m-1} \beta_i - \beta_m = \alpha_{m-1}$$

to recover

$$\beta_m = \frac{\alpha_m - \alpha_{m-1}}{2},$$

which generalizes as

$$\forall j = m, \dots, 1 : \beta_j = \frac{\alpha_j - \alpha_{j-1}}{2}, \quad \beta_0 = \alpha_0 + \sum_{i=1}^m \beta_i.$$

By convexity of  $\eta(u)$ , we have that

$$\forall j = m, \dots, 1 : \beta_j = \frac{\alpha_j - \alpha_{j-1}}{2} > 0.$$

Since  $\eta(u) \in C^2([a, b])$ , the linear interpolation exists and we have

$$\lim_{m \rightarrow \infty} \eta_m(u) = \eta(u), \quad \lim_{m \rightarrow \infty} \eta'_m(u) = \eta'(u).$$

This immediately implies that

$$\lim_{m \rightarrow \infty} \eta'_m(u) f'(u) = \eta'(u) f'(u) = \psi'(u).$$

Using the Kružkov pair, we define

$$\psi_m(u) = \beta_0 (f(u) - f(c)) + \sum_{i=1}^m \beta_i \operatorname{sign}(u - k_i) (f(u) - f(k_i)) + c_m,$$

where  $c_m$  is chosen such that  $\psi_m(0) = \psi(0)$ . Using the same argument as above, it is easily shown that

$$\lim_{m \rightarrow \infty} \psi_m(u) = \psi(u).$$

Now consider (2.18) in weak form:

$$\begin{aligned}
 & \int_{\Omega_x} \int_{\Omega_t} \eta(u) \frac{\partial \phi}{\partial t} + \psi(u) \frac{\partial \phi}{\partial x} dx dt \\
 &= \lim_{m \rightarrow \infty} \int_{\Omega_x} \int_{\Omega_t} \eta_m(u) \frac{\partial \phi}{\partial t} + \psi_m(u) \frac{\partial \phi}{\partial x} dx dt \\
 &= \lim_{m \rightarrow \infty} \int_{\Omega_x} \int_{\Omega_t} \beta_0 \left( (u - c) \frac{\partial \phi}{\partial t} + (f(u) - f(c)) \frac{\partial \phi}{\partial x} \right) + c_m \frac{\partial \phi}{\partial x} dx dt \\
 &+ \lim_{m \rightarrow \infty} \int_{\Omega_x} \int_{\Omega_t} \sum_{i=1}^m \beta_i \left( |u - k_i| \frac{\partial \phi}{\partial t} + \text{sign}(f(u) - f(k_i)) \frac{\partial \phi}{\partial x} \right) dx dt.
 \end{aligned}$$

Since  $u$  is a weak solution, the first term vanishes and, by Theorem 2.12 and  $\beta_i > 0$ , the latter sum is semipositive. Hence, it follows that

$$\int_{\Omega_x} \int_{\Omega_t} \eta(u) \frac{\partial \phi}{\partial t} + \psi(u) \frac{\partial \phi}{\partial x} dx dt \geq 0,$$

to complete the proof.  $\square$

This establishes the equivalence between the Kružkov entropy condition and the general entropy condition in Theorem 2.8. This result extends to multidimensional scalar equations.

## References

- [1] Alberto Bressan. *Hyperbolic Systems of Conservation Laws: The One-Dimensional Cauchy Problem*, volume 20 of Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2000.
- [2] Richard Courant and Kurt O. Friedrichs. *Supersonic Flow and Shock Waves*, volume 21. Springer Science+Business Media, 1976.
- [3] Constantine M. Dafermos. *Hyperbolic Conservation Laws in Continuum Physics*, volume 325 of Grundlehren der Mathematischen Wissenschaften. Springer-Verlag, 2010.
- [4] Lawrence C. Evans. *Partial Differential Equations*, 2nd ed., American Mathematical Society, 1998.
- [5] Dietmar Kröner. *Numerical Schemes for Conservation Laws*, Wiley, 1997.
- [6] Stanislav N. Kružkov. First order quasilinear equations in several independent variables. *Mathematics of the USSR-Sbornik*, 10(2):217, 1970.
- [7] Peter D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, volume 11 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1973.
- [8] Philippe G. LeFloch. *Hyperbolic Systems of Conservation Laws: The Theory of Classical and Nonclassical Shock Waves*, Lectures in Mathematics, ETH Zürich. Birkhäuser-Verlag, 2002.

- [9] Randall J. LeVeque. *Numerical Methods for Conservation Laws*, Lectures in Mathematics, ETH Zürich. Birkhäuser-Verlag, 1992.
- [10] Olga A. Oleinik. Discontinuous solutions of non-linear differential equations. *Uspekhi Matematicheskikh Nauk*, 12:3–73, 1957.
- [11] Joel Smoller. *Shock Waves and Reaction-Diffusion Equations*, volume 258 of Grundlehren der Mathematischen Wissenschaften. Springer Science+Business Media, 1994.

# Chapter 3

# Systems of conservation laws

Having gained an understanding of the mathematical properties of the scalar conservation law, let us now turn to the more interesting and, unfortunately, considerably more complex case of systems of conservation laws. In this chapter we discuss conservation laws of the form

$$\frac{\partial \mathbf{u}(x, t)}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u}, x, t)}{\partial x} = 0, \quad x \in \Omega_x \subset \mathbb{R}, \quad (3.1)$$

subject to initial conditions  $\mathbf{u}(x, 0) = \mathbf{u}_0(x)$ . Here  $\mathbf{u} : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  and  $\mathbf{f} : \mathbb{R}^m \times \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$ . Hence, we consider systems with  $m$  equations but restrict attention to problems posed in one spatial dimension. Concluding the chapter, we briefly highlight some of the challenges associated with the extension to multidimensional problems for which substantial parts of the fundamental theory remain incomplete.

## 3.1 • Linear systems

To begin this discussion, we consider the flux  $\mathbf{f} = \mathcal{A}(x, t)\mathbf{u}$ , where  $\mathcal{A}(x, t)$  is an  $m \times m$  matrix. This results in the linear system

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathcal{A}\mathbf{u}}{\partial x} = 0, \quad x \in \Omega_x, \quad (3.2)$$

subject to the initial condition,  $\mathbf{u}_0(x)$ . For simplicity, we assume periodic boundary conditions.

Let us also define the inner product and the associated norm

$$(\mathbf{u}, \mathbf{v})_{\Omega_x} = \int_{\Omega_x} \mathbf{u} \cdot \mathbf{v} \, dx, \quad \|\mathbf{u}\|_{\Omega_x}^2 = (\mathbf{u}, \mathbf{u})_{\Omega_x},$$

and recall the definition in [10].

**Definition 3.1.** *A problem is well posed if it admits a unique solution  $\mathbf{u}$  and there exist nonnegative numbers  $\alpha$  and  $K$  such that*

$$\|\mathbf{u}(t)\|_{\Omega_x} \leq K e^{\alpha(t-t_0)} \|\mathbf{u}(t_0)\|_{\Omega_x}.$$

Requiring wellposedness of a problem controls the sensitivity of the problem, i.e., if two different initial conditions are considered, wellposedness guarantees that the distance between the two solutions can grow no faster than exponential in time. This requirement is an essential property since any physical or computational model will be subject to noise.

A first understanding of the linear problem is given in the following result [10].

**Theorem 3.2.** *If we assume that  $\mathcal{A}(x, t)$  is symmetric and  $|\mathcal{A}_x(x, t)|$  is uniformly bounded for all  $(x, t)$ , then (3.2) is well posed.*

**Proof.** Uniqueness follows directly from linearity. Consider

$$\left( \mathbf{u}, \frac{\partial \mathbf{u}}{\partial t} \right)_{\Omega_x} + \left( \frac{\partial \mathbf{u}}{\partial t}, \mathbf{u} \right)_{\Omega_x} + \left( \mathbf{u}, \frac{\partial \mathcal{A}\mathbf{u}}{\partial x} \right)_{\Omega_x} + \left( \frac{\partial \mathcal{A}\mathbf{u}}{\partial x}, \mathbf{u} \right)_{\Omega_x} = 0.$$

Integration by parts and symmetry of  $\mathcal{A}$  implies

$$\left( \mathbf{u}, \frac{\partial \mathcal{A}\mathbf{u}}{\partial x} \right)_{\Omega_x} = - \left( \frac{\partial \mathcal{A}\mathbf{u}}{\partial x}, \mathbf{u} \right)_{\Omega_x} + \left( \frac{\partial \mathcal{A}}{\partial x} \mathbf{u}, \mathbf{u} \right)_{\Omega_x}.$$

This yields

$$\frac{1}{2} \frac{d}{dt} (\mathbf{u}, \mathbf{u})_{\Omega_x} \leq \max_{x \in \Omega_x} |\mathcal{A}_x(x, t)| \|\mathbf{u}\|_{\Omega_x}^2,$$

from which the result follows.  $\square$

To develop a better understanding of the nature of the solutions to (3.2), consider the frozen problem  $\mathcal{A}(x_0, t_0) = \mathbf{A}$ , for some  $(x_0, t_0) \in \Omega_x \times \Omega_t$ , as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{A} \frac{\partial \mathbf{u}}{\partial x} = 0, \quad x \in \Omega_x. \quad (3.3)$$

If  $\mathbf{A}$  is symmetric, wellposedness follows from Theorem 3.2.

Now assume that  $\mathbf{A}$  is uniformly diagonalizable, i.e., for all values of  $(x_0, t_0) \in \Omega_x \times \Omega_t$ , there exists a matrix  $\mathbf{S}$  such that

$$\mathbf{A} = \mathbf{S} \Lambda \mathbf{S}^{-1},$$

where  $\Lambda_{ii} = \lambda_i$  is a diagonal matrix and  $\lambda_i$  are the  $m$  eigenvalues of  $\mathbf{A}$ . We may then rewrite (3.3) as

$$\frac{\partial \mathbf{v}}{\partial t} + \Lambda \frac{\partial \mathbf{v}}{\partial x} = 0,$$

where  $\mathbf{v} = \mathbf{S}^{-1} \mathbf{u}$ . Since we have reduced the system to  $m$  decoupled scalar equations, discussed at length in Chapter 2, energy conservation requires that  $\Lambda = \Lambda^*$ . Here  $\mathbf{u}^*$  represents the complex conjugate of  $\mathbf{u}$ . Hence, all eigenvalues must be real.

If we explicitly express the solutions to (3.3) as

$$\mathbf{u}(x, t) = e^{\mathbf{A}(t-t_0)} \mathbf{u}(x, t_0),$$

where the matrix exponential is defined by the Taylor expansion

$$e^{At} = \sum_{k=0}^{\infty} \frac{1}{k!} (At)^k,$$

then the Cayley–Hamilton theorem yields

$$\|\mathbf{u}(t)\|_{\Omega_x} \leq \|\mathbf{S}\|_{\Omega_x} \left\| \mathbf{S}^{-1} \right\|_{\Omega_x} \left\| e^{\Lambda(t-t_0)} \right\|_{\Omega_x} \|\mathbf{u}(t_0)\|_{\Omega_x},$$

where the matrix norms are defined in the usual manner, based on the subordinate vector norm,  $\|\cdot\|_{\Omega_x}$ . Wellposedness is ensured provided

$$\|\mathbf{S}\|_{\Omega_x} \left\| \mathbf{S}^{-1} \right\|_{\Omega_x} \leq K, \quad \left\| e^{\Lambda(t-t_0)} \right\|_{\Omega_x} \leq e^{\alpha(t-t_0)}$$

uniformly in  $(x, t) \in \Omega_x \times \Omega_t$ .

We define (3.3) as hyperbolic provided all eigenvalues of  $\mathbf{A}$  are real. To ensure wellposedness, we must also require that  $\mathbf{A}$  is uniformly diagonalizable, leading to the characterization of the system as being symmetric hyperbolic if  $\mathbf{A}$  is symmetric, strictly hyperbolic if  $\mathbf{A}$  has distinct eigenvalues, and strongly hyperbolic if  $\mathbf{A}$  is uniformly diagonalizable. Otherwise we call the system weakly hyperbolic. Going forward we assume that the system is strongly hyperbolic unless stronger conditions are required.

We now express (3.3) as the decoupled problem

$$\frac{\partial \mathbf{v}}{\partial t} + \Lambda \frac{\partial \mathbf{v}}{\partial x} = 0.$$

Since each of the scalar problems has the solution

$$v_i(x, t) = v_i(x - \lambda_i t, 0),$$

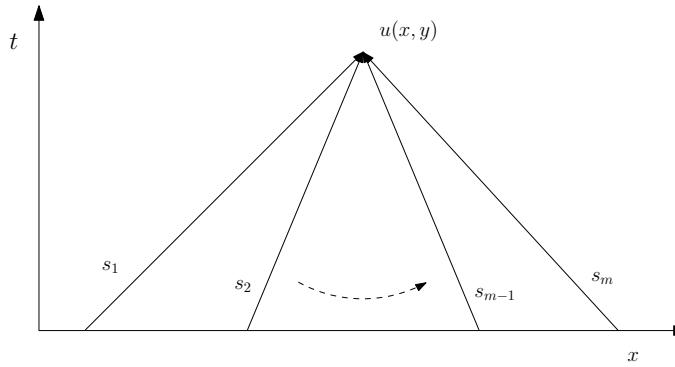
we recover the general solution to (3.3):

$$\mathbf{u}(x, t) = \sum_{i=1}^m v_i(x - \lambda_i t, 0) \mathbf{s}_i, \quad (3.4)$$

where  $\lambda_i$  is the real eigenvalue and  $\mathbf{s}_i$  the associated eigenvector such that  $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_m]$ . As illustrated in Fig. 3.1, the solution to (3.3) is composed of up to  $m$  linear waves, propagating at the speed given by the eigenvalues of  $\mathbf{A}$  and with the structure of the associated eigenvector. We generally assume that  $\lambda_i$  are ordered such that  $\lambda_1 \geq \lambda_2 \geq \dots \geq 0 \geq \dots \geq \lambda_m$ .

Apart from illustrating the construction of the solution, Fig. 3.1 also highlights another fundamental property of the hyperbolic problem—the finite speed of propagation. At any given point  $(x_0, t_0) \in \Omega_x \times \Omega_t$ , the solution depends only on past solutions,  $\mathbf{u}(x, t)$ , within the domain of dependence  $\mathcal{D} = [x_0 + \lambda_1(t - t_0), x_0 + \lambda_m(t - t_0)]$ . Recall that in this case  $t < t_0$ , so the domain of dependence represents a cone, expanding backward in time. The finite extent of the domain of dependence is a fundamental property of the hyperbolic problem, distinguishing it from the heat equation which has the entire  $\Omega_x$  as the domain of dependence [6].

Since  $\lambda_i$  can be either positive or negative, the general solution is composed of waves propagating both to the left and to the right. This is a central difference to the scalar problem, where only one wave exists at any given time.



**Figure 3.1.** Construction of the solution  $u(x, t)$  as the superposition of  $m$  linear waves, assuming an ordering of  $\lambda_1 \geq \lambda_2 \geq \dots \geq 0 \geq \dots \geq \lambda_m$ . The domain of dependence is contained within the cone defined by  $\lambda_1$  and  $\lambda_m$ .

Returning to the nonlinear case, a connection between (3.1) and (3.3) is obtained by rewriting the former in quasilinear form:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathcal{A}(\mathbf{u}) \frac{\partial \mathbf{u}}{\partial x} = 0,$$

where  $\mathcal{A}(\mathbf{u}) = \nabla_{\mathbf{u}} f$  is the Jacobian of the flux and we have introduced the gradient

$$\nabla_{\mathbf{u}} = \left( \frac{\partial}{\partial u_1}, \dots, \frac{\partial}{\partial u_m} \right).$$

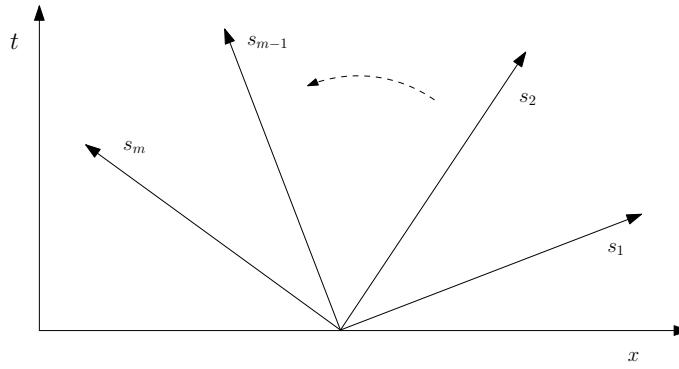
We must require that  $\mathcal{A}(\mathbf{u})$  is uniformly diagonalizable with real eigenvalues for the range of  $\mathbf{u}(x, t)$ . In this case, the eigenvalues  $\lambda_i$  now depend on  $\mathbf{u}$  and the solution can no longer be expressed as in (3.4). However, locally in time and space, the analysis of the frozen coefficient problem still holds and highlights that the solution in the  $(x, t)$ -plane is constructed from up to  $m$  waves, as illustrated in Fig. 3.1.

An alternative and perhaps complementary interpretation, gained from the analysis of the frozen coefficient problem but with a similar interpretation for the nonlinear problem, is illustrated in Fig. 3.2. This highlights that from every point in  $(x, t)$  information will propagate along  $m$  (or less) characteristic directions.

We can define the domain of influence as the subset of  $\Omega_x$  which can be impacted by the solution at  $(x_0, t_0) \in \Omega_x \times \Omega_t$ . For the linear problem with  $m$  waves, one finds this to be  $\mathcal{D} = [x_0 + \lambda_m(t - t_0), x_0 + \lambda_1(t - t_0)]$ . Solutions outside the domain of influence cannot be affected by the solution at  $(x_0, t_0)$ .

**Example 3.3.** Consider the one-dimensional Euler equations, introduced in subsection 1.4.1. We can express this in quasilinear form with

$$\mathcal{A}(\mathbf{q}) = \begin{bmatrix} 0 & 1 & 0 \\ -\frac{3-\gamma}{2}u^2 & (3-\gamma)u & \gamma-1 \\ -\frac{\gamma E u}{\rho} + (\gamma-1)u^3 & \frac{\gamma E}{\rho} - \frac{3(\gamma-1)u^2}{2} & \gamma u \end{bmatrix}.$$



**Figure 3.2.** Development of the solution  $u(x, t)$  for a system with  $m$  linear waves. We have assumed an ordering of  $\lambda_1 \geq \lambda_2 \geq \dots \geq 0 \geq \dots \geq \lambda_m$ . The domain of influence is contained within the cone defined by  $\lambda_m$  and  $\lambda_1$ .

This can be diagonalized by

$$S = \begin{bmatrix} \alpha & 1 & \alpha \\ \alpha(u+c) & u & \alpha(u-c) \\ \alpha(H+cu) & \frac{1}{2}u^2 & \alpha(H-cu) \end{bmatrix},$$

and

$$S^{-1} = \begin{bmatrix} 2\alpha(\frac{1}{2}(\gamma-1)u^2 - cu) & -2\alpha((\gamma-1)u - c) & 2\alpha(\gamma-1) \\ 1 - \frac{1}{2}(\gamma-1)\frac{u^2}{c^2} & \frac{\gamma-1}{c^2}u & -\frac{\gamma-1}{c^2} \\ 2\alpha(\frac{1}{2}(\gamma-1)u^2 + cu) & -2\alpha((\gamma-1)u + c) & 2\alpha(\gamma-1) \end{bmatrix}.$$

Here we have introduced

$$\alpha = \frac{1}{2c}, \quad c = \sqrt{\frac{\gamma p}{\rho}}, \quad H = \frac{E+p}{\rho}$$

as a scaling constant, the speed of sound, and the enthalpy, respectively.

As a result, we recover the diagonal matrix

$$S^{-1}AS = \Lambda = \begin{bmatrix} u+c & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & u-c \end{bmatrix},$$

which reflects the two sound waves, propagating at speeds  $u \pm c$ , and the entropy wave, propagating at the speed of the fluid flow. ■

## 3.2 • Riemann problems

Let us initially return to the linear constant coefficient case

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{A} \frac{\partial \mathbf{u}}{\partial x} = 0, \quad (3.5)$$

and assume, for simplicity, that the problem is strictly hyperbolic.

Consider a discontinuous initial condition of the form

$$\mathbf{u}_0(x) = \begin{cases} \mathbf{u}_l, & x < 0, \\ \mathbf{u}_r, & x > 0, \end{cases}$$

where  $\mathbf{u}_l \neq \mathbf{u}_r$ . The goal is to understand how this initial condition evolves, at least for a short period of time. This problem is known as the Riemann problem and its answer will play a central role in the remainder of the text.

Using (3.4), we express the two states as linear combinations of the characteristic waves

$$\mathbf{u}_l = S\boldsymbol{\alpha}, \quad \mathbf{u}_r = S\boldsymbol{\beta},$$

where  $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_m]^T$  and  $\boldsymbol{\beta} = [\beta_1, \dots, \beta_m]^T$ . Since  $\mathbf{v} = S^{-1}\mathbf{u}$ , we recover that

$$v_i(x, 0) = \begin{cases} \alpha_i, & x < 0, \\ \beta_i, & x > 0. \end{cases}$$

Utilizing our understanding of the solution to the scalar problem, we obtain

$$v_i(x, t) = \begin{cases} \alpha_i, & x - \lambda_i t < 0, \\ \beta_i, & x - \lambda_i t > 0. \end{cases}$$

If we now order the characteristic wave speeds  $\lambda_i$  as  $\lambda_1 > \lambda_2 > \dots > \lambda_m$  we can express the solution as

$$\mathbf{u}(x, t) = \sum_{i=1}^{q(x, t)} \alpha_i s_i + \sum_{i=q(x, t)+1}^m \beta_i s_i, \quad (3.6)$$

where  $q(x, t)$  is the maximal index  $p$  for which  $x - \lambda_p t < 0$ .

**Example 3.4.** Consider a linear system with the system matrix defined as

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix}.$$

One finds the eigenvalues of  $\mathbf{A}$  to be  $\lambda_1 = 3, \lambda_2 = 1, \lambda_3 = -1$ , i.e., there are three waves with one propagating to the left and two propagating to the right. Since  $\mathbf{A}$  is symmetric, the problem is symmetric hyperbolic and, thus, well posed.

To recover the exact solution, we need the eigenvectors of  $\mathbf{A}$ :

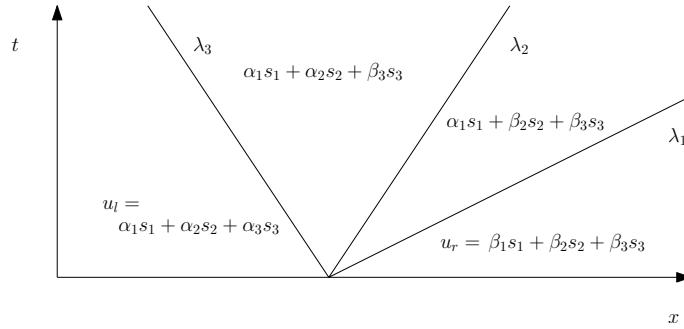
$$\mathbf{s}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{s}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{s}_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}.$$

Now, consider a Riemann problem with

$$\mathbf{u}_l = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{u}_r = \begin{bmatrix} 0 \\ 0 \\ -2 \end{bmatrix},$$

which yields the two vectors

$$\boldsymbol{\alpha} = [\sqrt{2}, 1, \sqrt{2}]^T, \quad \boldsymbol{\beta} = [\sqrt{2}, 1, -\sqrt{2}]^T.$$



**Figure 3.3.** Development of the solution  $u(x, t)$  for linear problem with  $m = 3$  waves. In this case  $\lambda_1 = 3, \lambda_2 = 1, \lambda_3 = -1$ .

If we focus on  $x = 0$ , we recover  $q(0, t) = 2$  and obtain the exact solution

$$u(0, t) = \alpha_1 s_1 + \alpha_2 s_2 + \beta_3 s_3.$$

The construction of the intermediate solutions is illustrated in Fig. 3.3. ■

Since we consider a linear constant coefficient problem, we know that the solution is constant along each of the characteristics and, thus, constant in the  $(x, t)$ -space between any two characteristic waves. Hence, the jump  $[u]$  of the solution across the  $p$ th characteristic line is

$$[u] = (\alpha_p - \beta_p) s_p.$$

To interpret this, consider

$$[f] = A[u] = (\alpha_p - \beta_p) A s_p = \lambda_p (\alpha_p - \beta_p) s_p = \lambda_p [u],$$

which we recognize as the Rankine–Hugoniot condition derived in Chapter 2, i.e., this condition holds across each of the characteristic waves.

We can directly connect the left and the right states by first expressing the exact solution, (3.6), as

$$u(x, t) = u_l - u_l + \sum_{i=1}^{q(x, t)} \alpha_i s_i + \sum_{i=q(x, t)+1}^m \beta_i s_i.$$

However, since  $u_l = S\alpha$ , we have

$$u(x, t) = u_l + \sum_{i=q(x, t)+1}^m (\beta_i - \alpha_i) s_i = u_l + \sum_{\lambda_i < x/t} (\beta_i - \alpha_i) s_i.$$

Similarly, we obtain

$$u(x, t) = u_r - \sum_{i=1}^{q(x, t)} (\beta_i - \alpha_i) s_i = u_r - \sum_{\lambda_i > x/t} (\beta_i - \alpha_i) s_i.$$

If we subtract one expression from the other, we recover

$$\mathbf{u}_l - \mathbf{u}_r = [\mathbf{u}] = \sum_{i=1}^m (\alpha_i - \beta_i) \mathbf{s}_i.$$

Thus, the solution of the Riemann problem can be seen as a decomposition of the initial jump into as many as  $m$  jumps, each of which is propagated forward at its characteristic wave speed,  $\lambda_i$ .

**Example 3.5.** Let us consider the one-dimensional Maxwell's equations (see subsection 1.4.1),

$$\varepsilon \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \mu \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x},$$

and assume that  $\varepsilon$  and  $\mu$  are constants. We rewrite this as

$$\mathbf{M} \frac{\partial \mathbf{q}}{\partial t} + \mathbf{A} \frac{\partial \mathbf{q}}{\partial x} = 0, \quad (3.7)$$

with

$$\mathbf{q} = \begin{bmatrix} E \\ H \end{bmatrix}, \quad \mathbf{M} = \begin{bmatrix} \varepsilon & 0 \\ 0 & \mu \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Although (3.7) is not exactly on the form discussed so far, such a form could be recovered by multiplying by  $\mathbf{M}^{-1}$ . Let us, however, pursue a slightly different approach and consider the matrix pencil

$$\mathbf{A} \mathbf{s}_i = \lambda_i \mathbf{M} \mathbf{s}_i,$$

from which we recover

$$\lambda_{1,2} = \mp \frac{1}{\sqrt{\varepsilon \mu}} = \mp c, \quad \mathbf{s}_{1,2} = \frac{1}{\sqrt{1+Z^2}} \begin{bmatrix} Z \\ \mp 1 \end{bmatrix}.$$

Here  $c$  is the speed of light and  $Z = \sqrt{\mu/\varepsilon}$  is the material impedance. Assuming that  $\mathbf{A} = \mathbf{S} \Lambda \mathbf{S}^T$ , we consider

$$\mathbf{S}^T \mathbf{M} \mathbf{S} \frac{\partial \mathbf{v}}{\partial t} + \mathbf{S}^T \mathbf{A} \mathbf{S} \frac{\partial \mathbf{v}}{\partial x} = \mathbf{S}^T \mathbf{M} \mathbf{S} \left( \frac{\partial \mathbf{v}}{\partial t} + \Lambda \frac{\partial \mathbf{v}}{\partial x} \right) = 0,$$

where  $\mathbf{V} = \mathbf{S}^T \mathbf{u}$ . As should be expected, we recover a general solution of the form

$$\mathbf{q}(\mathbf{x}, t) = v_1(\mathbf{x} + ct, 0) \mathbf{s}_1 + v_2(\mathbf{x} - ct, 0) \mathbf{s}_2,$$

which we recognize as two counter propagating light waves.

If we again consider the Riemann problem, we have that

$$\mathbf{q}_l = \mathbf{S} \boldsymbol{\alpha}, \quad \mathbf{q}_r = \mathbf{S} \boldsymbol{\beta},$$

and the jump across the  $i$ th wave is

$$[\mathbf{q}] = (\alpha_i - \beta_i) \mathbf{s}_i.$$

This yields

$$A[q] = (\alpha_i - \beta_i) A s_i = (\alpha_i - \beta_i) \lambda_i M s_i = \lambda_i M[u],$$

which is the Rankine–Hugoniot condition for (3.7).  $\blacksquare$

Let us now return to the nonlinear case. It is clear that finding a solution to the Riemann problem requires the construction of a path from  $u_l$  to  $u_r$ , through  $m$  or less waves. As we have already seen in Chapter 2, these waves can be shocks, rarefaction waves, or other types of waves.

To further explore this, let us first define the hyperbolic system as genuinely nonlinear if

$$\forall u : \nabla_u \lambda_i(u) \cdot s_i(u) \neq 0,$$

where  $(\lambda_i, s_i)$  are the eigenpairs of  $A(u) = \nabla_u f$ . Thus, this condition implies that  $\lambda_i$  must change monotonically along the curve  $s_i(u)$ . For  $m = 1$ , the condition requires that  $\frac{\partial \lambda_i}{\partial u} = f''(u) \neq 0$ , thus guaranteeing that  $f'(u)$  changes monotonically.

A wave for which

$$\forall u : \nabla_u \lambda_i(u) \cdot s_i(u) = 0 \quad (3.8)$$

is called a contact wave. The interpretation of (3.8) is that the eigenvalue along  $s_i(u)$  is constant and, hence, the characteristic curves for contacts are straight lines. For linear systems for which  $\lambda_i$  is constant, this is true for all waves, i.e., the contact wave can be expected to behave as a linear wave. For nonlinear systems, this condition implies that  $\lambda_i(u_l) = \lambda_i(u_r)$  if  $u_l$  and  $u_r$  connects through  $s_i(u)$ . We also conclude that contact waves are not possible in genuinely nonlinear systems.

Let us consider a simple characteristic:

$$\frac{d\gamma(t)}{dt} = \lambda(u(t), t),$$

where  $\lambda$  represents an eigenvalue of  $\nabla_u f$ . We define a  $C^1$ -function,  $w : \mathbb{R}^m \rightarrow \mathbb{R}$ , as the solution to the problem

$$(\nabla_u f)^T \nabla_u w = \lambda \nabla_u w. \quad (3.9)$$

The existence of a solution of this is guaranteed by the assumption that the problem is strongly hyperbolic. Consider

$$\begin{aligned} \frac{d}{dt} w(u(\gamma(t), t)) &= \nabla_u w \cdot \left( \frac{\partial}{\partial t} u + \gamma'(t) \nabla u \right) \\ &= \nabla_u w \cdot (-\nabla_u f \nabla u + \lambda \nabla u) = \nabla u \cdot (-(\nabla_u f)^T \nabla_u w + \lambda \nabla_u w) = 0, \end{aligned}$$

where the last step follows from (3.9). Hence  $w$  is constant along the characteristics  $\gamma(t)$ .

If we now consider  $s_k$  as an eigenvector to  $\nabla_u f$  we have

$$\lambda s_k^T \cdot \nabla_u w = s_k^T (\nabla_u f)^T \nabla_u w = \lambda_k s_k^T \cdot \nabla_u w.$$

Hence, if  $\lambda \neq \lambda_k$ , then

$$s_k^T \cdot \nabla_u w = 0.$$

This leads us to define Riemann invariants.

**Definition 3.6.** A  $C^1$ -function  $w : R^m \rightarrow R$  is called a  $k$ -Riemann invariant if

$$s_k \cdot \nabla_u w = 0,$$

where  $s_k$  is the  $k$ th eigenvector of  $\nabla_u f$ .

Furthermore, if

$$(\nabla_u f)^T \nabla_u w = \lambda \nabla_u w,$$

then  $w$  is called a Riemann invariant with respect to  $\lambda$ .

A further important property, proved in [13, 17], is the following.

**Theorem 3.7.** There exists  $m - 1$   $k$ -Riemann invariants such that  $\nabla_u w_i$  are all linearly independent.

We now consider the Riemann problem where the distance between  $\mathbf{u}_l$  and  $\mathbf{u}_r$  is small. In this case, a solution to this problem always exists [14, 17].

**Theorem 3.8.** Assume the two states,  $\mathbf{u}_l$  and  $\mathbf{u}_r$ , are sufficiently close. Then the Riemann problem for the genuinely nonlinear hyperbolic system has a solution comprising up to  $m + 1$  states, separated by shocks or rarefaction waves.

**Proof.** Let us first prove that if two states are sufficiently close, they can always be connected by a wave. For this, consider the function  $v(z)$  defined as the solution to

$$\frac{dv}{dz} = s_k(v(z)), \quad v(\lambda_k(\mathbf{u}_l)) = \mathbf{u}_l,$$

for  $z > \lambda_k(\mathbf{u}_l)$ . As previously,  $(\lambda_k, s_k)$  is an eigenpair of  $\nabla_u f$ . Clearly for  $\lambda_k(\mathbf{u}_l) \leq z \leq \lambda_k(\mathbf{u}_l) + \nu$ , and  $\nu$  sufficiently small,  $v(z)$  exists. Furthermore

$$\frac{d\lambda_k(v(z))}{dz} = \nabla_u \lambda_k \cdot \frac{dv}{dz} = \nabla_u \lambda_k \cdot s_k = 1$$

by normalization, since the problem is assumed to be genuinely nonlinear. This implies that  $\lambda_k(v(z)) = z$ , i.e., for  $\nu$  sufficiently small, the eigenvalue increases linearly.

Now consider the solution, expressed as a traveling wave

$$\mathbf{u}(x, t) = v\left(\frac{x}{t}\right) = v(z),$$

where

$$\lambda_k(\mathbf{u}_l) \leq z \leq \lambda_k(\mathbf{u}_l) + \nu.$$

If  $w$  is a  $k$ -Riemann invariant, then

$$\frac{dw}{dz} = \nabla_u w \cdot \frac{du}{dz} = \nabla_u w \cdot \frac{dv}{dz} = \nabla_u w \cdot s_k(v(z)) = 0.$$

Thus,  $w$  is constant and Theorem 3.7 guarantees that the wave is a simple wave. Furthermore,

$$\begin{aligned}\frac{\partial \mathbf{u}}{\partial t} + \nabla_{\mathbf{u}} f \frac{\partial \mathbf{u}}{\partial x} &= \left( -\frac{x}{t^2} + \nabla_{\mathbf{u}} f \frac{1}{t} \right) \frac{dv}{dz} = \left( -\frac{x}{t^2} + \nabla_{\mathbf{u}} f \frac{1}{t} \right) s_k \\ &= \left( -\frac{x}{t^2} + \lambda_k \frac{1}{t} \right) s_k = \left( -\frac{x}{t^2} + \frac{x}{t} \frac{1}{t} \right) s_k = 0.\end{aligned}$$

Hence,  $v$  is a traveling wave solution to the conservation law and there is a one-parameter family  $\mathbf{u}(z)$  through which the general state is connected to  $\mathbf{u}_l$  by a simple wave. Furthermore, we know that the  $k$ -Riemann invariant is constant, i.e.,  $w_i(\mathbf{u}) = w_i(\mathbf{u}_l)$  for  $i = 1, \dots, m-1$  and  $\lambda_k(\mathbf{u})$  increases linearly. If we therefore define a function  $F : \mathbb{R}^m \rightarrow \mathbb{R}^m$  as

$$F(w_1(\mathbf{u}) - w_1(\mathbf{u}_l), \dots, w_{m-1}(\mathbf{u}) - w_{m-1}(\mathbf{u}_l), \lambda_k(\mathbf{u}) - \lambda_k(\mathbf{u}_l) - v),$$

we can seek the solution between the two states as  $F(\mathbf{u}) = 0$ . Since  $\nabla_{\mathbf{u}} w_i$  are linearly independent and the problem is genuinely nonlinear, the Jacobian is nonsingular. By the inverse function theorem, it has a unique solution provided  $v$  is small enough.

For the general case of  $m$  waves, this result can be used repeatedly. Starting at  $\mathbf{u}_l$ , we know there is a one-parameter family of solutions,  $\mathbf{u}_1(z_1)$ , connecting to the next state. The path to the third state is then described by two parameters,  $\mathbf{u}_2(z_1, z_2)$ . This argument can be repeated across the  $m$  waves, reaching  $\mathbf{u}_m(z_1, \dots, z_m)$ . Since

$$\frac{d\mathbf{u}_m(z_k)}{dz_k} \propto s_k(\mathbf{u}_l),$$

and all  $s_k$  are orthogonal by strong hyperbolicity, the mapping is bijective and we can always find a set of parameters  $z_i$  such that  $\mathbf{u}_r = \mathbf{u}_m(z_1, \dots, z_m)$ . This provides a solution to the Riemann problem.  $\square$

Unfortunately, once  $\|\mathbf{u}_l - \mathbf{u}_r\| \leq \varepsilon$  is violated, this result on local existence of solutions to the Riemann problem does not, in general, hold. Examples of Riemann problems with no solution are known. We refer to [17] for a further discussion of this.

A stronger result than the above is given in the following seminal result.

**Theorem 3.9.** *The initial value problem for the system of conservation laws in one dimension has a global in time solution, provided the initial condition has sufficiently small oscillations and total variation.*

For a proof of this, we refer to [7]. This result can be extended to include contact discontinuities as required for the Euler equations. However, the result does not guarantee that the solution is an entropy solution and it does not extend beyond one spatial dimension.

### 3.3 • Entropy conditions and functions

As discussed in detail in Chapter 2, the nonlinear problem requires the introduction of weak solutions, which lead to the possible loss of uniqueness of the solution. In case of the system, we will have  $m$  waves and we will need to ensure that each of these waves is entropy satisfying.

To establish suitable entropy conditions for this scenario, first recall that for a scalar conservation law with a convex flux, we have

$$s[u] = [f], \quad f'(u_l) > s > f'(u_r),$$

which are, respectively, recognized as the Rankine–Hugoniot condition and the Lax entropy condition.

For the system and a discontinuity propagating at speed  $s$ , we need to require for some  $1 \leq k \leq m$  that

$$\lambda_1(u_l) > \lambda_2(u_l) > \dots > \lambda_k(u_l) > s > \lambda_{k+1}(u_l) > \dots > \lambda_m(u_l),$$

and, likewise,

$$\lambda_1(u_r) > \lambda_2(u_r) > \dots > \lambda_j(u_r) > s > \lambda_{j+1}(u_r) > \dots > \lambda_m(u_r).$$

To find the  $2m+1$  unknowns,  $u_l, u_r, s$ , we have the  $m$  Rankine–Hugoniot conditions as well as the  $k$  and  $m-j$  conditions from above. Solvability requires that  $j = k-1$ , to recover the Lax entropy condition

$$\lambda_k(u_l) > s > \lambda_{k+1}(u_l), \quad \lambda_{k-1}(u_r) > s > \lambda_k(u_r).$$

These conditions must hold across every shock, defined as a discontinuity that satisfies the entropy conditions and the Rankine–Hugoniot condition.

For the general nonlinear problem, computing the appropriate left and right states cannot be done explicitly. The notable exception to this is the linear hyperbolic problem, where all intermediate states can be derived explicitly as exemplified in the following problem.

**Example 3.10.** We continue Ex. 3.5 for Maxwell's equations

$$\varepsilon \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \mu \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x},$$

and rewrite this as

$$M \frac{\partial \mathbf{q}}{\partial t} + A \frac{\partial \mathbf{q}}{\partial x} = 0, \quad (3.10)$$

with

$$\mathbf{q} = \begin{bmatrix} E \\ H \end{bmatrix}, \quad M = \begin{bmatrix} \varepsilon & 0 \\ 0 & \mu \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

From Ex. 3.5 we know that

$$\lambda_{1,2} = \mp \frac{1}{\sqrt{\varepsilon \mu}} = \pm c,$$

with  $c$  being the speed of light.

We now write the two Rankine–Hugoniot conditions as

$$-cM(\mathbf{q}_l - \mathbf{q}^*) = A(\mathbf{q}_l - \mathbf{q}^*), \quad cM(\mathbf{q}^* - \mathbf{q}_r) = A(\mathbf{q}^* - \mathbf{q}_r).$$

Let us first assume that  $M$  is continuous. In this case we get the trivial connection that

$$\mathbf{q}_l = \mathbf{q}^* = \mathbf{q}_r,$$

since the solution is continuous. If we consider the more interesting case where  $M$  may change due to discontinuous material coefficients, we recover

$$-c_l(M_l \mathbf{q}_l - M^* \mathbf{q}^*) = A(\mathbf{q}_l - \mathbf{q}^*), \quad c_r(M^* \mathbf{q}^* - M_r \mathbf{q}_r) = A(\mathbf{q}^* - \mathbf{q}_r),$$

which yields the intermediate state

$$\mathbf{q}^* = \frac{1}{c_l + c_r} A(c_r Q_l \mathbf{q}_l - c_l Q_r \mathbf{q}_r), \quad M^* \mathbf{q}^* = \frac{1}{c_l + c_r} (Q_l \mathbf{q}_l + Q_r \mathbf{q}_r),$$

where

$$Q_l = c_l M_l + A, \quad Q_r = c_r M_r - A. \quad \blacksquare$$

As discussed for the scalar problem, an alternative approach is to seek an entropy pair  $(\eta, \psi)$  satisfying

$$\frac{\partial \eta(\mathbf{u})}{\partial t} + \frac{\partial \psi(\mathbf{u})}{\partial x} = 0,$$

where  $\mathbf{u}$  is a smooth solution to the system of conservation laws. Differentiation yields

$$\nabla_{\mathbf{u}} \eta \cdot \frac{\partial \mathbf{u}}{\partial t} + \nabla_{\mathbf{u}} \psi \cdot \frac{\partial \mathbf{u}}{\partial x} = 0,$$

and from the conservation law itself we have

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} = \frac{\partial \mathbf{u}}{\partial t} + \nabla_{\mathbf{u}} \mathbf{f} \frac{\partial \mathbf{u}}{\partial x}.$$

Hence, the entropy pair must satisfy the condition

$$\nabla_{\mathbf{u}} \eta \nabla_{\mathbf{u}} \mathbf{f} = (\nabla_{\mathbf{u}} \psi)^T. \quad (3.11)$$

For the scalar case, this reduces to  $\eta' f' = \psi'$ . The most immediate challenge that arises in this general case is that for  $m > 1$  one cannot guarantee that a solution to (3.11) exists since the system is severely overconstrained. Nevertheless, for many physical systems, a physically motivated definition of the entropy often allows the identification of families of entropy pairs. Prominent examples include the nonlinear equations of elastodynamics, the Euler and Navier–Stokes equations, and various formulations of the magnetohydrodynamic equations [12, 11].

Before discussing entropy conditions for general systems, let us first assume that an entropy pair exists, as this has profound implications. We seek to identify a new set of variables  $\mathbf{v}$ , such that the conservation law becomes symmetric, i.e.,

$$\frac{\partial \mathbf{u}(\mathbf{v})}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u}(\mathbf{v}))}{\partial x} = \nabla_{\mathbf{v}} \mathbf{u} \frac{\partial \mathbf{v}}{\partial t} + \nabla_{\mathbf{v}} \mathbf{f} \frac{\partial \mathbf{v}}{\partial x} = 0. \quad (3.12)$$

Let us now consider the two matrices  $\nabla_{\mathbf{v}} \mathbf{u}$  and  $\nabla_{\mathbf{v}} \mathbf{f}$ . If their symmetry is to be ensured, it must hold that  $\mathbf{u}$  and  $\mathbf{f}$  are gradients of a scalar function

$$\nabla_{\mathbf{v}} \mathbf{q}(\mathbf{v}) = \mathbf{u}^T, \quad \nabla_{\mathbf{v}} \mathbf{r}(\mathbf{v}) = \mathbf{f}^T. \quad (3.13)$$

By further requiring  $q$  to be convex,  $\nabla_v \mathbf{u}$  is ensured to be positive definite, thus enabling the solution of the transformed system (3.12).

The first of two surprising results is the following theorem [15].

**Theorem 3.11.** *Assume that  $\eta(\mathbf{u})$  is an entropy for (3.12). Then the new variables*

$$\mathbf{v}^T = \nabla_u \eta(\mathbf{u})$$

*symmetrize the conservation law.*

**Proof.** Since  $\eta$  is convex, the mapping between  $\mathbf{u}$  and  $\mathbf{v}$  is bijective. Now define

$$q(\mathbf{v}) = \mathbf{v}^T \mathbf{u} - \eta(\mathbf{u}),$$

and differentiate it with respect to  $\mathbf{v}$  to recover

$$\nabla_v q(\mathbf{v}) = \mathbf{u}^T + \mathbf{v}^T \nabla_v \mathbf{u} - \nabla_u \eta(\mathbf{u}) \nabla_v \mathbf{u} = \mathbf{u}^T,$$

by the definition of  $\mathbf{v}^T$ . Similarly, define

$$r(\mathbf{v}) = \mathbf{v}^T \mathbf{f}(\mathbf{u}) - \psi(\mathbf{u}),$$

and compute its gradient as

$$\nabla_v r(\mathbf{v}) = \mathbf{f}^T + \mathbf{v}^T (\nabla_u \mathbf{f} \nabla_v \mathbf{u}) - \nabla_u \psi \nabla_v \mathbf{u} = \mathbf{f}^T,$$

by (3.11). Convexity of  $q(\mathbf{v})$  follows from convexity of  $\eta(\mathbf{u})$ .  $\square$

This remarkable result shows that if an entropy pair exists, then there exists a set of variables which transforms the nonlinear conservation law into symmetric form. It is perhaps even more surprising that the converse statement also holds true [9].

**Theorem 3.12.** *Assume that (3.12) can be symmetrized by a new set of variables,  $\mathbf{v}$ . Then there exists an entropy pair in the form*

$$\eta(\mathbf{u}) = \mathbf{u}^T \mathbf{v} - q(\mathbf{v}), \quad \psi(\mathbf{u}) = \mathbf{f}^T \mathbf{v} - r(\mathbf{v}),$$

*where the scalar functions  $(q, r)$  are defined as (3.13).*

**Proof.** Consider

$$\nabla_u \eta(\mathbf{u}) = \mathbf{v}^T + \mathbf{u}^T \nabla_u \mathbf{v} - \nabla_v q(\mathbf{v}) \nabla_u \mathbf{v} = \mathbf{v}^T,$$

by the definition of  $q$ . Similarly,

$$\nabla_u \psi(\mathbf{u}) = \mathbf{v}^T \nabla_u \mathbf{f} + \mathbf{f}^T \nabla_u \mathbf{v} - \nabla_v r(\mathbf{v}) \nabla_u \mathbf{v} = \mathbf{v}^T \nabla_u \mathbf{f},$$

by (3.13). This implies that

$$q(\mathbf{v}) = \mathbf{v}^T \mathbf{u} - \eta(\mathbf{u}),$$

which, when differentiated with respect to  $\mathbf{v}$ , recovers (3.11). Convexity of  $\eta(\mathbf{u})$  follows from convexity of  $q(\mathbf{v})$ .  $\square$

We have thus established the equivalence between the existence of an entropy pair and the ability to symmetrize the conservation law. This result holds for general systems in multiple dimensions [11]. An immediate consequence is that any system with an entropy is symmetrizable and, thus, hyperbolic.

**Example 3.13.** If we consider the Euler equations (subsection 1.4.1), the search for entropy pairs is successful. In fact, there is an infinite family of entropy pairs of the form [11]

$$\eta(\mathbf{q}) = -\rho Q(t), \quad \psi(\mathbf{q}) = -\rho u Q(t),$$

where  $Q(t)$  is a function that satisfies

$$\frac{Q''}{Q'} \leq \gamma^{-1},$$

to ensure convexity of the entropy. The transformed entropy variables are given as

$$\mathbf{v} = \frac{Q'}{\iota} \begin{bmatrix} \iota(\gamma - Q/Q') - \frac{1}{2}u^2 \\ u \\ -1 \end{bmatrix},$$

where  $\iota$  is the internal energy, i.e.,  $\rho\iota = E - \frac{1}{2}\rho u^2$ .

If we choose [12]

$$Q(u) = \log\left(\frac{\gamma-1}{\rho\gamma} \rho\iota\right)$$

and define

$$\mathbf{A}_0 = \frac{\rho\iota}{(\gamma-1)v_3} \begin{bmatrix} -v_3^2 & e_1 & v_3(1-k_1) \\ e_1 & c_1 & v_2k_2 \\ v_3(1-k_1) & v_2k_2 & -k_3 \end{bmatrix},$$

$$\mathbf{A}_1 = \frac{\rho\iota}{(\gamma-1)v_3^2} \begin{bmatrix} e_1v_3 & c_1v_3 & k_2e_1 \\ c_1v_3 & -(c_1 + 2(\gamma-1)v_3)v_2 & c_1k_2 + (\gamma-1)v_2^2 \\ k_2e_1 & c_1k_2 + (\gamma-1)v_2^2 & k_4v_2 \end{bmatrix},$$

where

$$k_1 = \frac{v_2^2}{2v_3}, \quad k_2 = k_1 - \gamma, \quad k_3 = k_1^2 - 2\gamma k_1 + \gamma, \quad k_4 = k_2^2 - (\gamma-1)(k_1 + k_2),$$

$$c_1 = (\gamma-1)v_3 - v_2^2, \quad e_1 = v_2v_3,$$

we can write the Euler equations in the form

$$\mathbf{A}_0 \frac{\partial \mathbf{v}}{\partial t} + \mathbf{A}_1 \frac{\partial \mathbf{v}}{\partial x} = 0,$$

where  $\mathbf{v} = (v_1, v_2, v_3)$  are the entropy variables.

As intended, the nonlinear system has been symmetrized. The nonlinear flux is given as

$$f(v) = \frac{\rho \iota}{v_3} \begin{bmatrix} e_1 \\ c_1 \\ k_2 v_2 \end{bmatrix}.$$

It is a remarkable fact that this same construction symmetrizes the full three-dimensional Euler equations, as well as the much more complex compressible Navier-Stokes equations [12, 1, 5]. ■

As for the scalar case, we have the following result.

**Theorem 3.14.** *For a hyperbolic problem, endowed with an entropy pair  $(\eta, \psi)$  and with a weak entropy solution  $u$ , we have*

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0,$$

in a weak sense.

**Proof.** The proof, similar to that of Theorem 2.8, introduces a viscosity solution and exploits convexity of the entropy. Details can be found in [13]. □

Similarly, we have the following result.

**Theorem 3.15.** *Consider a hyperbolic problem, endowed with an entropy pair  $(\eta, \psi)$  and with a weak entropy solution  $u$  with a shock. If*

$$\frac{\partial \eta(u)}{\partial t} + \frac{\partial \psi(u)}{\partial x} \leq 0$$

holds in a weak sense, the Lax entropy conditions are satisfied, provided the shock is weak.

**Proof.** The proof is similar to that of Theorem 2.9. Details can be found in [17]. □

Chapters 1 and 2 have been devoted to the development of an adequate understanding of scalar conservation laws in multiple dimensions, systems of equations in one dimension, and a study of the nature of their solutions. It is tempting to continue to pursue an understanding of the full multidimensional system. Unfortunately, the theoretical underpinning of basic questions like existence and uniqueness of solutions to systems of conservation laws in multiple dimensions is incomplete and an area of active research. We refer to texts such as [4, 3, 2, 16, 8] for an introduction to recent work and open questions in this important, but technically very demanding, area of research.

## References

- [1] Saul Abarbanel and David Gottlieb. Optimal time splitting for two-and three-dimensional Navier-Stokes equations with mixed derivatives. *Journal of Computational Physics*, 41(1):1–33, 1981.

- 
- [2] Sylvie Benzoni-Gavage and Denis Serre. *Multi-dimensional Hyperbolic Partial Differential Equations: First-Order Systems and Applications*. Oxford University Press, 2007.
  - [3] Gui-Qiang Chen. Euler equations and related hyperbolic conservation laws. In: *Evolutionary Equations, vol. II*, Handbook of Differential Equations, C. M. Dafermos and E. Feireisl, eds., Elsevier-North Holland, 2005, pp. 1–104.
  - [4] Constantine M. Dafermos. *Hyperbolic Conservation Laws in Continuum Physics*, volume 325 of Grundlehren der Mathematischen Wissenschaften. Springer-Verlag, Berlin, 2010.
  - [5] Pravir Dutt. Stable boundary conditions and difference schemes for Navier-Stokes equations. *SIAM Journal on Numerical Analysis*, 25(2):245–267, 1988.
  - [6] Lawrence C. Evans. *Partial Differential Equations*, 2nd ed., American Mathematical Society, 1998.
  - [7] James Glimm. Solutions in the large for nonlinear hyperbolic systems of equations. *Communications on Pure and Applied Mathematics*, 18(4):697–715, 1965.
  - [8] Edwige Godlewski and Pierre-Arnaud Raviart. Hyperbolic Systems of Conservation Laws, Mathématiques & Applications, vol. 3/4, *Ellipses, Paris*, 1991.
  - [9] Sergei K. Godunov. An interesting class of quasilinear systems. *Doklady Akademii Nauk SSSR*, 139:521–523, 1961.
  - [10] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. *Time Dependent Problems and Difference Methods*, volume 24 of Pure and Applied Mathematics. John Wiley & Sons, 1995.
  - [11] Amiram Harten. On the symmetric form of systems of conservation laws with entropy. *Journal of Computational Physics*, 49(1):151–164, 1983.
  - [12] Thomas Hughes, Leopold Franca, and Michel Mallet. A new finite element formulation for computational fluid dynamics. I. Symmetric forms of the compressible Euler and Navier–Stokes equations and the second law of thermodynamics. *Computer Methods in Applied Mechanics and Engineering*, 54(2):223–234, 1986.
  - [13] Dietmar Kröner. *Numerical Schemes for Conservation Laws*. Wiley, 1997.
  - [14] Peter D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, volume 11 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1973.
  - [15] Michael S. Mock. Systems of conservation laws of mixed type. *Journal of Differential equations*, 37(1):70–88, 1980.
  - [16] Denis Serre. *Systems of Conservation Laws, vols. 1 and 2*. Cambridge University Press, 1999.
  - [17] Joel Smoller. *Shock Waves and Reaction-Diffusion Equations*, volume 258 of Grundlehren der Mathematischen Wissenschaften. Springer Science+Business Media, 1994.

# Chapter 4

# From continuous to discrete

Having gained a good understanding of conservation laws and the nature of their solutions, we can now begin the development of numerical methods to solve such problems. We must strive to ensure that the basic properties of the conservation laws carry over to the numerical schemes. As we shall soon experience, many of the central challenges of the subsequent analysis originate from this requirement.

We seek to develop methods with broad applicability but for the sake of simplicity, we focus most of the discussion on methods for the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad (x, t) \in \Omega_x \times \Omega_t,$$

with a suitable initial condition,  $u_0(x)$ , and periodic boundary conditions.

Without being specific about the details of the scheme, we generally focus on grid based methods and seek a solution  $U_j^n$  as an approximation to  $u(x_j, t^n)$ , where we introduce the grid  $(x_j, t^n)$  as

$$x_j = h j = \frac{L}{N} j, \quad t^n = n k = \frac{T}{K} k,$$

with  $j = 0 \dots N$ ,  $k = 0 \dots K$ , and  $\Omega_x = [0, L]$ ,  $\Omega_t = [0, T]$ .

To illustrate some of the challenges lying ahead, let us consider a simple example.

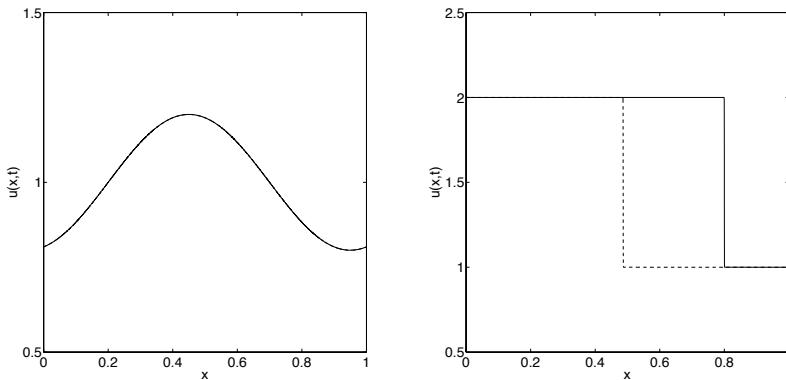
**Example 4.1.** We consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = f(x, t), \quad x \in [0, 1],$$

subject to the smooth initial condition

$$u_0(x) = 1 + 0.2 \sin(2\pi x).$$

We define  $f(x, t)$  such that  $u(x, t) = 1 + 0.2 \sin(2\pi(x-t))$  is the exact solution. Shortly, we also consider the case of a nonsmooth initial condition, discussed in subsection 1.4.1.



**Figure 4.1.** The solution at  $T = 0.2$  of Burgers' equation with a smooth (left) and nonsmooth (right) solution, computed using a nonconservative (dashed line) and a conservative scheme (solid line).

To seek an approximate solution, we consider two different grid based schemes, defined as

$$u_j^{n+1} = u_j^n - 2\frac{k}{h}u_j^n(u_j^n - u_{j-1}^n) + kf_j^n$$

and

$$u_j^{n+1} = u_j^n - \frac{k}{h}[(u_j^n)^2 - (u_{j-1}^n)^2] + kf_j^n.$$

The main difference between these two schemes is in the treatment of the nonlinear part of the equation. We refer to the first scheme as being in nonconservative form while the second scheme is said to be in conservative form.

The computed solutions, obtained by using the two schemes with the smooth and the nonsmooth initial condition, exhibit distinct differences. As shown in Fig. 4.1, the two schemes both appear to accurately compute the smooth solution. However, the nonconservative scheme fails completely for the nonsmooth case for which we recall that the correct solution is a moving shock that propagates at a speed of 3, i.e., at final time  $T = 0.2$ , the shock should be located at  $x = 0.8$ .

The result obtained with the nonconservative scheme illustrates a very dangerous type of failure since the scheme computes a solution comprising the expected moving shock, yet this shock moves at the wrong speed. Hence, it violates the Rankine-Hugoniot condition and, therefore, is not a weak solution to the conservation law.

For this simple problem, we can easily identify the wrong solution. However, for a complex application this may be much harder to ascertain, leaving open the risk of considering a physically wrong solution. ■

The above example emphasizes the importance of developing schemes that mimic the properties of the continuous problem to ensure that the computed solution is a weak entropy solution to the conservation law.

We shall now revisit many of the same questions that arose in the analysis of the continuous problem to identify central properties that we must require of the numerical scheme to ensure convergence to the correct solution to the conservation law.

## 4.1 • Conservation and conservation form

We define a scheme as being in conservation form if it can be written as

$$U_j^{n+1} = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n], \quad (4.1)$$

where we have introduced the numerical flux

$$F_{j+1/2}^n = F(U_{j-p}^n, \dots, U_{j+q}^n), \quad F_{j-1/2}^n = F(U_{j-p-1}^n, \dots, U_{j+q-1}^n).$$

The numerical flux can also be expressed as

$$F_{j+1/2}^n = F(E_l U^n), \quad F_{j-1/2}^n = F(U^n),$$

where  $U^n = (U_{j-p-1}^n, \dots, U_{j+q-1}^n)$  and  $E_l$  is the shift operator,  $E_l U_j = U_{j+l}$ . Hence, the update of  $U_j^n$  depends on  $(U_{j-p-1}^n, \dots, U_{j+q}^n)$ , which we refer to as the stencil of the scheme. Although  $U_j^n$  may refer to a grid point value, as in the discussion so far, it may also, as we shall explore later, be useful to consider different definitions.

The numerical flux  $F(U_{-p}, \dots, U_q)$  in (4.1) should clearly be related to  $f(u)$  of the conservation law. We call the numerical flux consistent if  $F(u, \dots, u) = f(u)$ . In general, we also assume that the numerical flux is Lipschitz continuous, i.e., there exists a constant  $M > 0$  such that

$$|F(U_{-p}, \dots, U_q) - f(u)| \leq M \max(|U_{-p} - u|, \dots, |U_q - u|),$$

provided that  $|U_i - u|$  is sufficiently small at every point  $u$ .

We often restrict the discussion to the simplest case of  $p + q = 1$ , which reflects a numerical flux that only depends on neighboring cells. As we shall soon discover, whether  $p$  or  $q$  should take the value of one is a problem-dependent issue.

The importance of the conservation form is highlighted by a few key results, the first one of which establishes conservation of the numerical scheme, also known as discrete conservation.

**Theorem 4.2.** *A scheme in conservation form with a consistent flux is conservative at the discrete level.*

**Proof.** For simplicity and without loss of generality, we consider the local scheme with  $p = 0, q = 1$ ,

$$U_j^{n+1} = U_j^n - \frac{k}{h} [F(U_j^n, U_{j+1}^n) - F(U_{j-1}^n, U_j^n)] = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n].$$

Summing over  $j$  we recover that

$$b \sum_j U_j^{n+1} - b \sum_j U_j^n = -k \sum_j [F_{j+1/2}^n - F_{j-1/2}^n].$$

As the sum on the right-hand side is telescopic, we obtain

$$b \sum_j U_j^{n+1} - b \sum_j U_j^n = -k \left[ F_{N+1/2}^n - F_{-1/2}^n \right].$$

If we assume that the solution is constant outside  $[0, L]$ , consistency of the numerical flux implies that

$$F_{N+1/2}^n = f(u(L, t^n)), \quad F_{-1/2}^n = f(u(0, t^n)),$$

leading to

$$b \sum_j U_j^{n+1} - b \sum_j U_j^n = -k [f(u(L, t^n)) - f(u(0, t^n))]. \quad (4.2)$$

In the periodic case, this immediately yields

$$b \sum_j U_j^{n+1} = b \sum_j U_j^n = b \sum_j U_j^{n-1} = \dots = b \sum_j U_j^0.$$

This reflects conservation at the discrete level, provided  $U_j^0$  is defined such that

$$b \sum_j U_j^0 = \int_{\Omega_x} u_0(x) dx.$$

If we instead assume that  $u$  is constant outside  $\Omega_x$ , we sum (4.2) over  $n$  to obtain

$$b \sum_j U_j^K - b \sum_j U_j^0 = -T [f(u(L)) - f(u(0))].$$

Defining  $U_j^0$  as above, we recover

$$\int_{\Omega_x} u(x, T) dx - \int_{\Omega_x} u(x, 0) dx = -T [f(u(L)) - f(u(0))],$$

which is recognized as the integral form of the conservation law (2.5), for which conservation is an inherent property.  $\square$

An immediate consequence of this result is that if a scheme in conservation form is used, a discontinuity cannot propagate at the wrong speed as this would violate the Rankine–Hugoniot condition and, thus, conservation.

However, we can take this one step further and recover the Lax–Wendroff theorem [4, 5].

**Theorem 4.3.** *Assume that a numerical scheme is given in conservation form with a consistent and Lipschitz continuous numerical flux. If the scheme converges to a total variation bounded solution in  $L^1$ , the solution is a weak solution to the conservation law.*

**Proof.** Let us first of all recall that a solution is called a weak solution if, for all smooth and compactly supported test functions,  $\phi(x, t)$ , it holds that

$$\int_{\Omega_t} \int_{\Omega_x} \left[ u \frac{\partial \phi}{\partial t} + f(u) \frac{\partial \phi}{\partial x} \right] dx dt = - \int_{\Omega_x} u_0(x) \phi(x, 0) dx.$$

We consider, for simplicity and without loss of generality, the scheme with  $p = 0, q = 1$ ,

$$U_j^{n+1} = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n].$$

Multiply by  $\phi(x_j, t^n) = \phi_j^n$  and sum over  $j$  and  $n$  to obtain

$$\sum_{j,n} \phi_j^n \left( \frac{U_j^{n+1} - U_j^n}{k} \right) = - \sum_{j,n} \phi_j^n \frac{F_{j+1/2}^n - F_{j-1/2}^n}{h}.$$

Let us now recall the telescoping property

$$\begin{aligned} \sum_{k=0}^p a^k (b^{k+1} - b^k) &= -a^0 b^0 + a^0 b^1 + a^1 b^2 - a^1 b^1 + \cdots + a^p b^{p+1} - a^p b^p \\ &= -a^0 b^0 - \sum_{k=1}^p b^k (a^k - a^{k-1}) + a^p b^{p+1}. \end{aligned}$$

When combined with the compactness of  $\phi$  ( $\phi \rightarrow 0$ , for  $(x, t) \rightarrow (\partial\Omega_x, T)$ ), this yields

$$- \sum_{j,n} U_j^n \left( \frac{\phi_j^n - \phi_j^{n-1}}{k} \right) - \frac{1}{k} \sum_j \phi_j^0 U_j^0 = \sum_{j,n} F_{j+1/2}^n \frac{\phi_{j+1}^n - \phi_j^n}{h}. \quad (4.3)$$

We now multiply by  $hk$  and consider each of the three terms for  $(h_l, k_l) \rightarrow 0$  as  $l \rightarrow \infty$ .

Since we assume  $L^1$ -convergence of  $U_j^n$  to  $u(x_j, t^n)$ , i.e.,

$$\lim_{l \rightarrow \infty} \sum_{j,n} |U_j^n - u(x_j, t^n)| h_l k_l = 0,$$

and  $\phi(x, t)$  is smooth, we have

$$\lim_{l \rightarrow \infty} h_l k_l \sum_{j,n} U_j^n \left( \frac{\phi_j^n - \phi_j^{n-1}}{k_l} \right) = \int_{\Omega_t} \int_{\Omega_x} u \frac{\partial \phi}{\partial t} dx dt.$$

In a similar fashion, the second term converges as

$$\lim_{l \rightarrow \infty} h_l \sum_j \phi_j^0 U_j^0 = \int_{\Omega_x} u_0(x) \phi(x, 0) dx.$$

The last term of (4.3) is slightly more complicated. We assume that the TV-norm of  $U^n$  is bounded:

$$TV(U^n) = \sup_b \sum_j |U_{j+1}^n - U_j^n| \leq R.$$

Combined with the assumption that the numerical flux is Lipschitz continuous, this yields

$$\lim_{l \rightarrow \infty} h_l k_l \sum_{j,n} F(U_j^n, U_{j+1}^n) \frac{\phi_{j+1}^n - \phi_j^n}{h_l} = \int_{\Omega_t} \int_{\Omega_x} f(u) \frac{\partial \phi}{\partial x} dx dt.$$

Hence, if  $U_j^n$  converges as  $l \rightarrow \infty$ , it converges to a solution  $u$  that satisfies

$$\int_{\Omega_t} \int_{\Omega_x} \left[ u \frac{\partial \phi}{\partial t} + f(u) \frac{\partial \phi}{\partial x} \right] dx dt = - \int_{\Omega_x} u_0(x) \phi(x, 0) dx.$$

Since  $\phi(x, t)$  is an arbitrarily smooth compactly supported test function,  $u$  is a weak solution to the conservation law.  $\square$

This key result establishes that, under very reasonable conditions, the solution computed using the conservation form is a weak solution to the conservation law, i.e., problems such as those illustrated in Fig. 4.1 are guaranteed to be absent.

## 4.2 • Monotonicity and entropy conditions

Having established the importance of the conservation form, one wonders if this suffices to guarantee the computation of the entropy solution to the conservation law. To illustrate that some additional conditions are needed, let us consider an example in some detail.

**Example 4.4.** Consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [-1, 1],$$

and let us first consider a problem with a discontinuous initial condition as

$$u_0(x) = \begin{cases} 2, & x < 0, \\ 1, & x \geq 0. \end{cases}$$

From section 2.1, we recall that this is an entropy satisfying solution and a solution of the form  $u(x, t) = u_0(x - 3t)$  is expected based on the Rankine–Hugoniot condition.

We solve the problem using a scheme in conservation form

$$u_j^{n+1} = u_j^n - \frac{k}{b} [F_{j+1/2}^n - F_{j-1/2}^n],$$

and we take  $u_j^n$  to represent a grid point value. This leaves open the question of how to choose the numerical flux  $F(u, v)$ .

We consider two different options for the numerical flux. The first one is

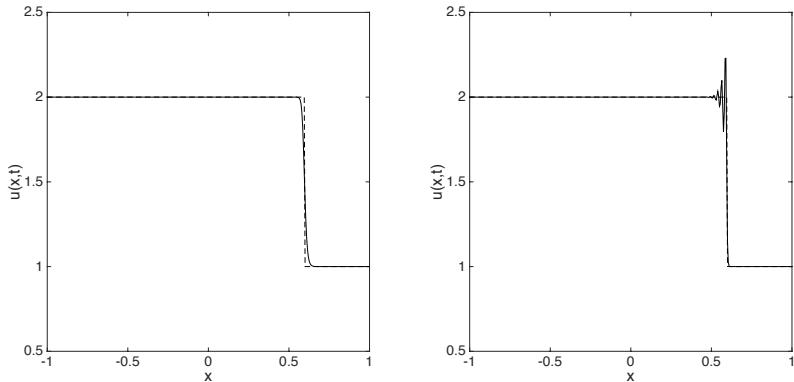
$$F_{LF}(u, v) = \frac{f(u) + f(v)}{2} - \frac{\alpha}{2}(v - u),$$

where  $\alpha = \max_u |f'(u)|$ . This is known as the Lax–Friedrichs flux or the Rusanov flux [6].

An alternative is the Lax–Wendroff flux, given as

$$F_{LW}(u, v) = \frac{f(u) + f(v)}{2} - \frac{k}{2b} f' \left( \frac{u + v}{2} \right) (f(v) - f(u)).$$

We shall derive these fluxes and discuss them in more detail in section 5.2. For now it suffices to note that both fluxes are consistent and, for small jumps, they appear very similar.



**Figure 4.2.** The solution at  $T = 0.2$  of Burgers' equation, solved using a scheme in conservation form with a Lax-Friedrichs numerical flux (left) and a Lax-Wendroff numerical flux (right). In both cases, the computed solution (solid line) and the exact solution (dashed line) is shown.

In Fig. 4.2 we show the results obtained with the two schemes. As both schemes are expressed in conservation form, the shock propagates at the correct speed. We observe that the Lax-Friedrichs scheme leads to a substantial smearing of the shock, while the Lax-Wendroff scheme captures the steepness of the shock better. Unfortunately, this appears to come at the cost of introducing strong oscillations upstream of the shock. As stated in (2.11), the total variation of the entropy solution must be decaying, i.e., the oscillations in the Lax-Wendroff scheme are clearly a numerical artifact.

Let us consider a different aspect of the challenges associated with computing accurate entropy solutions to conservation laws. We change the initial condition to

$$u_0(x) = \begin{cases} -1, & x < 0, \\ 1, & x \geq 0, \end{cases}$$

and recall from the discussion in Ex. 2.2 that although an infinite family of weak solutions exists, this is an initial condition for which the rarefaction wave is the correct entropy solution,

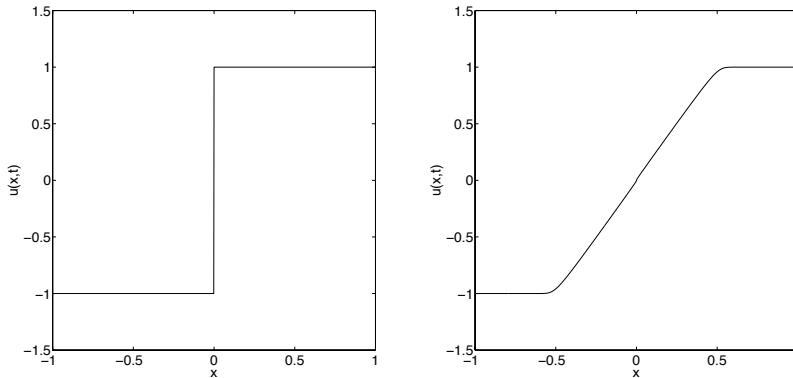
We first note that the Rankine–Hugoniot condition indicates that the speed of the discontinuity at  $x = 0$  is zero. As a numerical flux, we now choose

$$F(u, v) = \begin{cases} f(u), & s \geq 0, \\ f(v), & s < 0, \end{cases} \quad s = \frac{f(u) - f(v)}{u - v}, \quad (4.4)$$

known as the Roe flux. Here  $s$  is the shock speed based on  $(u, v)$ . The simple interpretation of this flux is that whenever  $s$  is positive, the shock moves from left to right and the upstream flux is used. When  $s$  is negative, the reverse approach is taken and the downstream flux is applied. In other words, it is simply an upwind approach for a nonlinear scheme.

Figure 4.3 shows the computed solution, a stationary discontinuity, obtained with (4.4). However, since the correct entropy solution is a rarefaction wave, we have computed an incorrect weak solution.

It is easy to realize how this happens, as the numerical flux, when combined with the initial condition, ensures that  $u_j^0 = u_j^1$ , thus maintaining the initial condition. This



**Figure 4.3.** The solution at  $T = 0.25$  of Burgers' equation with a nonsmooth initial condition, showing an entropy violating solution (left) and the correct entropy solution (right) computed with the same scheme, subject only to different initial conditions.

also suggests that we should be able to break this situation by changing either the initial condition or the numerical flux.

While it is not yet clear how to modify the latter, we can make a reasonable change of the initial condition as

$$u_0(x) = \begin{cases} -1, & x < 0, \\ 0, & x = 0, \\ 1, & x > 0, \end{cases}$$

i.e., we simply define the intermediate value of the discontinuity as the average value across the jump. As shown in Fig. 4.3, this breaks the initial shock and the correct rarefaction wave emerges.

This example illustrates the care needed in the construction of the numerical flux, as seemingly natural choices may result in wrong solutions. While this situation could be changed by slightly modifying the initial condition, this is not a practical solution in more complex cases. ■

It is clear that some properties of the numerical flux are key to ensure recovery of the entropy solution.

Let us recall the general scheme in conservation form

$$U_j^{n+1} = U_j^n - \lambda \left[ F_{j+1/2}^n - F_{j-1/2}^n \right] = G(U_{j-p-1}^n, \dots, U_{j+q}^n) := G(U^n)_j,$$

where we set  $\lambda = \frac{k}{\beta}$  to simplify the notation.

We call the scheme monotone if  $G(\cdot)$  is monotonically nondecreasing in all arguments, typically expressed using the notation  $G(\uparrow, \uparrow, \dots, \uparrow)$ . To understand the implications of requiring  $G(\cdot)$  to be monotone, let us restrict attention to the case of  $p = 0, q = 1$ , i.e.,

$$G(U_{j-1}^n, U_j^n, U_{j+1}^n) = U_j^n - \lambda \left[ F(U_j^n, U_{j+1}^n) - F(U_{j-1}^n, U_j^n) \right],$$

where, for the sake of clarity, we have written out  $F_{j\pm 1/2}^n$ . Computing the gradient of  $G(\cdot)$ , we recover

$$\begin{aligned}\frac{\partial G}{\partial U_{j-1}^n} &= \lambda \partial_1 F(U_{j-1}^n, U_j^n), \\ \frac{\partial G}{\partial U_j^n} &= 1 - \lambda \left[ \partial_1 F(U_j^n, U_{j+1}^n) - \partial_2 F(U_{j-1}^n, U_j^n) \right], \\ \frac{\partial G}{\partial U_{j+1}^n} &= -\lambda \partial_2 F(U_j^n, U_{j+1}^n),\end{aligned}$$

where  $\partial_i F$  indicates the derivative of the numerical flux  $F$  with respect to the  $i$ th argument.

Consequently, a necessary condition for  $G(\cdot)$  to be monotone is that the numerical flux is increasing in the first argument and decreasing in the second argument, i.e.,  $F(\uparrow, \downarrow)$ . By the same token, we must choose  $\lambda$  to be sufficiently small.

To require a scheme to be monotone has significant implications, as detailed in the following theorem.

**Theorem 4.5.** *Consider a monotone scheme. Then it holds*

1. *If  $U_j^n \leq V_j^n$ , then  $G(U^n)_j \leq G(V^n)_j$  for all  $j$ .*
2. *The numerical solution satisfies a local maximum principle*

$$\min_{i \in S_j} U_i^n \leq G(U^n)_j \leq \max_{i \in S_j} U_i^n,$$

where  $S_j$  represents the local stencil around  $j$ .

3. *The scheme is  $L^1$ -contractive:*

$$\|G(U^n)_j - G(V^n)_j\|_1 \leq \|U_j^n - V_j^n\|_1.$$

4. *The scheme is total variation diminishing (TVD).*

**Proof.** We shall prove each of the four statements separately.

Statement 1 is a direct consequence of  $G(\cdot)$  being monotone.

To establish Statement 2, let us choose a particular grid point,  $x_j$ , and define

$$V_i^n = \begin{cases} \max_{k \in S_j} U_k^n, & \text{if } i \in S_j, \\ U_i^n, & \text{otherwise.} \end{cases}$$

We have that  $U_j^n \leq V_j^n$  for all  $j$ . By consistency,  $G(V^n)_j = G(V_j^n, V_j^n, V_j^n) = V_j^n$  and, by Statement 1, it follows that

$$U_j^{n+1} = G(U^n)_j \leq G(V^n)_j = V_j^n \leq \max_{i \in S_j} U_i^n.$$

The reverse result can be proven in the same way.

Statement 3 is a central result, the proof of which is a bit more involved. Let us begin by introducing the notation

$$a \vee b = \max(a, b), \quad a \wedge b = \min(a, b), \quad a^+ = a \vee 0, \quad a^- = a \wedge 0.$$

We also define

$$W_j^n = U_j^n \vee V_j^n = V_j^n + (U_j^n - V_j^n)^+,$$

and observe that

$$G(W^n)_j \geq G(U^n)_j, \quad G(W^n)_j \geq G(V^n)_j,$$

by Statement 1. We also have

$$G(W^n)_j - G(V^n)_j \geq (G(U^n)_j - G(V^n)_j)^+.$$

Hence,

$$\begin{aligned} \sum_j (G(U^n)_j - G(V^n)_j)^+ &\leq \sum_j G(W^n)_j - G(V^n)_j = \sum_j W_j^{n+1} - V_j^{n+1} \\ &= \sum_j W_j^n - V_j^n = \sum_j (U_j^n - V_j^n)^+, \end{aligned}$$

where we have used conservation

$$\sum_j U_j^{n+1} = \sum_j U_j^n.$$

Observing that  $|a| = a^+ + (-a)^+$  we recover

$$\begin{aligned} \sum_j |G(U^n)_j - G(V^n)_j| &= \sum_j (G(U^n)_j - G(V^n)_j)^+ + \sum_j (G(V^n)_j - G(U^n)_j)^+ \\ &\leq \sum_j (U_j^n - V_j^n)^+ + \sum_j (V_j^n - U_j^n)^+ \\ &= \sum_j |U_j^n - V_j^n|, \end{aligned}$$

thus establishing the result.

Statement 4 follows directly by taking  $V_j^n = U_{j+1}^n$  in Statement 3.  $\square$

The importance of monotone schemes is now clear. Monotonicity ensures stability of the scheme, guarantees that no oscillations can be created, and implies that the scheme mimics essential properties of the conservation law.

**Example 4.6.** Let us analyze some of the fluxes discussed in detail in Ex. 4.4, beginning with the Lax–Friedrichs flux:

$$F_{LF}(u, v) = \frac{f(u) + f(v)}{2} - \frac{\alpha}{2}(v - u).$$

Consistency of the flux is immediate. To establish monotonicity, it suffices to show that  $F_{LF}(\uparrow, \downarrow)$ , i.e.,

$$\frac{\partial F_{LF}}{\partial u} = \frac{1}{2}(f'(u) + \alpha) \geq 0, \quad \frac{\partial F_{LF}}{\partial v} = \frac{1}{2}(f'(v) - \alpha) \leq 0,$$

where monotonicity follows if we define  $\alpha = \max_u |f'(u)|$ .

Consider now the Lax–Wendroff flux:

$$F_{LW}(u, v) = \frac{f(u) + f(v)}{2} - \frac{\lambda}{2} f'\left(\frac{u+v}{2}\right)(f(v) - f(u)).$$

Consistency of the flux is again immediate. To keep the discussion simple, we restrict ourselves to the Burgers' flux,  $f(u) = u^2$ , for which we have

$$F_{LW}(u, v) = \frac{u^2 + v^2}{2} - \frac{\lambda}{2}(u+v)(v^2 - u^2),$$

which implies

$$\frac{\partial F_{LW}}{\partial u} = u + \frac{\lambda}{2}[2(u+v)u + u^2 - v^2].$$

For a general value of  $v$ , positivity cannot be guaranteed, implying that the Lax–Wendroff flux is not monotone. This is consistent with the results in Fig. 4.2, where we observe the generation of oscillations in violation of the local maximum principle of Theorem 4.5.

Finally, let us consider the Roe flux introduced in (4.4), given as

$$F(u, v) = \begin{cases} f(u), & s \geq 0, \\ f(v), & s < 0. \end{cases}$$

Consider

$$\frac{\partial F}{\partial u} = \begin{cases} f'(u), & s \geq 0, \\ 0, & s < 0. \end{cases}$$

If we restrict attention to the Burgers' flux we obtain

$$\frac{\partial F}{\partial u} = \begin{cases} 2u, & (u+v) \geq 0, \\ 0, & (u+v) < 0. \end{cases}$$

It is easy to see that with a solution  $u < 0 < v$  positivity cannot be guaranteed. ■

We still have not fully realized the importance of the monotonicity requirement since we have not considered its connection to the entropy solution. This is established in the following result [3].

**Theorem 4.7.** *The solution obtained by a monotone scheme satisfies all entropy conditions.*

*Proof.* First recall the Kružkov entropy pair introduced in section 2.3

$$\eta(u) = |u - c| = u \vee c - u \wedge c, \quad \psi(u) = \text{sign}(u - c)(f(u) - f(c)) = f(u \vee c) - f(u \wedge c),$$

where  $c$  is a real constant. Also recall the equivalence of the entropy condition with

$$\frac{\partial \eta}{\partial t} + \frac{\partial \psi}{\partial x} \leq 0, \quad (4.5)$$

understood in a weak sense.

Now consider the cell entropy condition

$$\frac{\eta(U_j^{n+1}) - \eta(U_j^n)}{k} + \frac{\Psi_{j+1/2}^n - \Psi_{j-1/2}^n}{h} \leq 0.$$

If this holds in a cellwise sense, (4.5) clearly holds globally. We have introduced the numerical entropy flux

$$\Psi_{j+1/2}^n = F(U^n \vee c)_{j+1/2} - F(U^n \wedge c)_{j+1/2},$$

where, as usual,

$$F_{j+1/2}^n = F(U^n)_{j+1/2} = F(U_{j-p}^n, \dots, U_{j+q}^n).$$

Consider

$$\begin{aligned} G(U^n \vee c)_j &= (U^n \vee c)_j - \lambda [F(U^n \vee c)_{j+1/2} - F(U^n \vee c)_{j-1/2}], \\ G(U^n \wedge c)_j &= (U^n \wedge c)_j - \lambda [F(U^n \wedge c)_{j+1/2} - F(U^n \wedge c)_{j-1/2}] \end{aligned}$$

to obtain

$$G(U^n \vee c)_j - G(U^n \wedge c)_j = |U_j^n - c| - \lambda [\Psi_{j+1/2}^n - \Psi_{j-1/2}^n].$$

By consistency and monotonicity we have

$$c = G(c) \leq G(U_j^n \vee c), \quad U_j^{n+1} = G(U^n)_j \leq G(U^n \vee c)_j,$$

which implies that

$$c \vee U_j^{n+1} \leq G(U^n \vee c)_j.$$

Considering

$$\begin{aligned} \eta(U_j^{n+1}) &= |U_j^{n+1} - c| \leq G(U^n \vee c)_j - G(U^n \wedge c)_j \\ &= |U_j^n - c| - \lambda [\Psi_{j+1/2}^n - \Psi_{j-1/2}^n], \end{aligned}$$

this establishes the cell entropy condition.  $\square$

This is clearly an encouraging result, confirming the importance of monotone schemes. Let us now also consider a different observation of Ex. 4.4, i.e., the introduction of the artificial oscillations in the Lax–Wendroff scheme. Another expression of

the desired suppression of artificial oscillations is

$$\forall j : (U_{j+1}^n \geq U_j^n \Rightarrow U_{j+1}^{n+1} \geq U_j^{n+1}).$$

A scheme obeying this condition, prohibiting the generation of oscillations, is called a monotonicity-preserving scheme.

**Theorem 4.8.** *A TVD scheme is monotonicity preserving.*

**Proof.** Let us establish this by contradiction and begin by assuming that  $U_{j+1}^n \geq U_j^n$  for all  $j$ . Assume also that there is an index  $k$  where the condition for monotonicity preservation is violated, i.e.,  $U_{k+1}^{n+1} < U_k^{n+1}$ . If we now modify  $U_j^n$  to be constant outside the stencil used to compute  $U_{k+1}^{n+1}$  and  $U_k^{n+1}$ , then violating monotonicity preservation violates the TVD property.  $\square$

To summarize the results obtained so far, we have established the following relationships

$$\text{monotone} \Rightarrow L^1\text{-contractive} \Rightarrow \text{TVD} \Rightarrow \text{monotonicity preserving}.$$

So far, however, we have not experienced any substantial adverse impact of the requirement of monotonicity. To realize a very unfortunate consequence of this requirement, let us focus on the simplest of conservation laws, the linear wave equation, given as

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0,$$

subject to appropriate initial and boundary conditions. We express a general numerical scheme for its solution

$$U_j^{n+1} = \sum_{l=-p-1}^q c_l(\lambda) U_{j+l}^n. \quad (4.6)$$

For the linear problem, the schemes we have discussed so far can all be expressed in this form. We refer to this as a linear scheme where  $c_l(\lambda)$  may depend on  $\lambda$  and other constants, but not on  $U^n$ .

If we consider

$$U_j^{n+1} = G(U^n)_j = \sum_{l=-p-1}^q c_l(\lambda) U_{j+l}^n,$$

it is immediate that

$$\forall l : \partial_l G(U^n)_j = c_l(\lambda) \geq 0$$

if the scheme is monotone. Let us briefly consider previous schemes to see if this holds.

**Example 4.9.** We consider the linear problem

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad (4.7)$$

for which the monotone Lax–Friedrichs scheme becomes

$$U_j^{n+1} = \frac{U_{j+1}^n + U_{j-1}^n}{2} - \frac{\lambda}{2}(U_{j+1}^n - U_{j-1}^n).$$

A slight rewriting yields

$$U_j^{n+1} = G(U^n)_j = \frac{1}{2}(1 + \lambda)U_{j-1}^n + \frac{1}{2}(1 - \lambda)U_{j+1}^n = c_{-1}U_{j-1}^n + c_1U_{j+1}^n.$$

Provided  $|\lambda| \leq 1$ , then  $c_l \geq 0$  as required. As we shall see shortly,  $|\lambda| \leq 1$  is a natural condition.

If we instead consider the Lax–Wendroff scheme, we recover

$$U_j^{n+1} = U_j^n - \frac{\lambda}{2}(U_{j+1}^n - U_{j-1}^n) + \frac{\lambda^2}{2}(U_{j+1}^n - 2U_j^n + U_{j-1}^n).$$

A rewriting yields

$$U_j^{n+1} = \frac{\lambda(1 + \lambda)}{2}U_{j-1}^n + (1 - \lambda^2)U_j^n + \frac{\lambda(-1 + \lambda)}{2}U_{j+1}^n,$$

and we quickly realize that it is not possible to choose  $\lambda$  such that  $c_l \geq 0$ . ■

We have the following property for linear schemes.

**Theorem 4.10.** *A linear monotonicity-preserving scheme is monotone.*

*Proof.* Consider the monotone function

$$U_i = \begin{cases} 0, & i \leq m, \\ 1, & i > m. \end{cases} \quad (4.8)$$

Then

$$U_{j+1}^{n+1} = \sum_{l=-p-1}^q c_l(\lambda)U_{j+1+l}^n, \quad U_j^{n+1} = \sum_{l=-p-1}^q c_l(\lambda)U_{j+l}^n,$$

and thus

$$U_{j+1}^{n+1} - U_j^{n+1} = \sum_{l=-p-1}^q c_l(\lambda)(U_{j+l+1}^n - U_{j+l}^n).$$

For (4.8) we see that the sum vanishes unless  $j \in [m - p - 1, m + q]$ . In such cases have ( $j = m$ ), in which case

$$U_{m+1+l}^{n+1} - U_{m+l}^{n+1} = c_{-l}(\lambda) \geq 0,$$

for  $l \in [-p - 1, q]$  since the scheme is monotonicity preserving. □

The first indication of the restrictions imposed by the requirement of monotonicity is the following result, due to Godunov [1].

**Theorem 4.11.** *A linear monotone scheme is, at most, first order accurate.*

**Proof.** Let us assume there exists an analytic solution  $u(x, t)$  to the linear conservation law (4.7). Its Taylor expansion yields

$$u(x_{j+l}, t^n) = \sum_{r=0}^{\infty} \frac{(lh)^r}{r!} \frac{\partial^r}{\partial x^r} u(x_j, t^n).$$

Using the linear scheme (4.6), we express the computed solution as

$$U_j^{n+1} = \sum_{l=-p-1}^q c_l(\lambda) \sum_{r=0}^{\infty} \frac{(lh)^r}{r!} \frac{\partial^r}{\partial x^r} u(x_j, t^n).$$

On the other hand, the exact solution can likewise be expressed as

$$u(x_j, t^{n+1}) = \sum_{r=0}^{\infty} \frac{(lk)^r}{r!} \frac{\partial^r}{\partial t^r} u(x_j, t^n).$$

By using (4.7) to express temporal derivatives as spatial derivatives,

$$\frac{\partial^r u}{\partial t^r} = (-1)^r \frac{\partial^r u}{\partial x^r},$$

we can write the local error

$$u(x_j, t^{n+1}) - U_j^{n+1} = \sum_{r=0}^{\infty} \left( (-\lambda)^r - \sum_{l=-p}^q l^r c_l \right) \frac{h^r}{r!} \frac{\partial^r}{\partial x^r} u(x_j, t^n).$$

As discussed in more detail in Chapter 5, we must require that all terms for  $r \leq m$  vanish to recover a global error  $\mathcal{O}(h^m)$ , i.e.,

$$\forall r = 0 \dots m : \sum_{l=-p-1}^q l^r c_l(\lambda) = (-\lambda)^r.$$

Let us consider the three first conditions,

$$\sum_{l=-p-1}^q c_l = 1, \quad \sum_{l=-p-1}^q l c_l = -\lambda, \quad \sum_{l=-p-1}^q l^2 c_l = \lambda^2, \quad (4.9)$$

which must be satisfied to ensure a scheme of  $\mathcal{O}(h^2)$  accuracy. We express these as

$$\left( \sum_{l=-p-1}^q c_l \right) \left( \sum_{l=-p-1}^q l^2 c_l \right) = \left( \sum_{l=-p-1}^q l c_l \right)^2.$$

Since  $c_l > 0$  by monotonicity, we can write  $c_l = e_l^2$  to obtain

$$\left( \sum_{l=-p-1}^q e_l^2 \right) \left( \sum_{l=-p-1}^q l^2 e_l^2 \right) = \left( \sum_{l=-p-1}^q l e_l^2 \right)^2. \quad (4.10)$$

Defining the two vectors

$$\mathbf{a} = [e_{-p-1}, \dots, e_q]^T, \quad \mathbf{b} = [-(p+1)e_{-p-1}, \dots, qe_q]^T,$$

we express (4.10) as

$$(\mathbf{a} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{b}) = (\mathbf{a} \cdot \mathbf{b})^2.$$

Equality requires that  $\mathbf{a} = c\mathbf{b}$ , which is clearly not the case. Hence it is not possible to match the first three terms in the Taylor expansion, and therefore not possible to construct a second order accurate linear scheme.  $\square$

This is a disappointing result, since a scheme being monotone is at odds with the maximum attainable accuracy of the scheme, at least for the linear problem. Unfortunately, the result carries over to the general case [2].

**Theorem 4.12.** *A monotone scheme is, at most, first order accurate.*

*Proof.* We consider the general scheme

$$U_j^{n+1} = G(U^n)_j = U_j^n - \lambda [F_{j+1/2}^n - F_{j-1/2}^n].$$

For simplicity we restrict the proof to the case of  $p+q=1$ , which yields the numerical flux

$$F_{j+1/2}^n = F(U_j^n, U_{j+1}^n) = F(E_1 U^n), \quad F_{j-1/2}^n = F(U_{j-1}^n, U_j^n) = F(U^n),$$

where  $\mathbf{U}^n = (U_{j-1}^n, U_j^n)$  and  $E_1$  is the shift operator. We now express the scheme as

$$G(U^n)_j = G(U_{j-1}^n, U_j^n, U_{j+1}^n) = U_j^n - \lambda(F(E_1 U^n) - F(U^n)).$$

By consistency, we have

$$G(U, U, U) = U, \quad \sum_l \partial_l F(U, U) = f'(U).$$

Let us number the arguments of  $G(\cdot)$  as  $l = -1, 0, 1$  and, correspondingly, the two arguments of  $F(\cdot)$  as  $l = -1, 0$ . Then

$$\partial_l G(U^n) = \delta_{l,0} - \lambda [\partial_{l-1} F(E_1 U^n) - \partial_l F(U^n)],$$

where we define  $\partial_l F = 0$  if  $|l| > 1$ . In a similar fashion, we define

$$\partial_{l,m} G(U^n) = -\lambda [\partial_{l-1,m-1} F(E_1 U^n) - \partial_{l,m} F(U^n)],$$

where  $\partial_{l,m} F$  refers to elements of the Hessian of  $F$ .

Directly evaluating the sum of the elements, we realize

$$\sum_l \partial_l G(U, U, U) = 1, \quad \sum_l l \partial_l G(U, U, U) = -\lambda f'(U). \quad (4.11)$$

Provided  $F(\cdot)$  is consistent, we recognize the expression in (4.9). Similarly, we obtain

$$\sum_{l,m} (l-m)^2 \partial_{l,m} G(U, U, U) = 0. \quad (4.12)$$

Now consider the multivariate Taylor expansion:

$$\begin{aligned} G(U_{j-1}^n, U_j^n, U_{j+1}^n) &= G(U^n)_j + \sum_{l=-1}^1 (U_{j+l}^n - U_j^n) \partial_l G(U_j^n) \\ &\quad + \frac{1}{2} \sum_{l,m=-1}^1 (U_{j+l}^n - U_j^n)(U_{j+m}^n - U_j^n) \partial_{l,m} G(U_j^n) + \dots, \end{aligned} \quad (4.13)$$

where  $\mathbf{U}_j^n = [U_j^n, U_j^n, U_j^n]$ .

We consider each of the terms. Clearly, we have  $G(U^n)_j = U_j^n$  by consistency. Furthermore, if we assume that the conservation law has a smooth solution,  $u(x, t)$ , we have

$$\sum_{l=-1}^1 (U_{j+l}^n - U_j^n) \partial_l G(U_j^n) = bu_x \sum_{l=-1}^1 l \partial_l G(\mathbf{u}) + \frac{b^2}{2} u_{xx} \sum_{l=-1}^1 l^2 \partial_l G(\mathbf{u}) + \mathcal{O}(b^3).$$

For the last term in (4.13) we have

$$\frac{1}{2} \sum_{l,m=-1}^1 (U_{j+l}^n - U_j^n)(U_{j+m}^n - U_j^n) \partial_{l,m} G(U_j^n) = \frac{b^2}{2} (u_x)^2 \sum_{l,m=-1}^1 l m \partial_{l,m} G(\mathbf{u}) + \mathcal{O}(b^3).$$

In both cases,  $u_x = u_x(x_j, t^n)$  and likewise for  $u_{xx}$ , while  $\mathbf{u} = [u(x_j, t^n), \dots, u(x_j, t^n)]$ . Now realize that

$$u_{xx} \sum_{l=-1}^1 l^2 \partial_l G(\mathbf{u}) = \sum_{l=-1}^1 l^2 [\partial_l G(\mathbf{u}) u_x]_x - \sum_{l,m=-1}^1 l^2 u_x^2 \partial_{l,m} G(\mathbf{u}),$$

and, further,

$$\sum_{l,m=-1}^1 (l m - l^2) \partial_{l,m} G(\mathbf{u}) = -\frac{1}{2} \sum_{l,m=-1}^1 (l - m)^2 \partial_{l,m} G(\mathbf{u}) = 0,$$

by (4.12), and symmetry of  $\partial_{l,m} G(\mathbf{u}) = \partial_{m,l} G(\mathbf{u})$  by smoothness.

Using (4.11) we have

$$bu_x \sum_{l=-1}^1 l \partial_l G(\mathbf{u}) = bu_x (-\lambda f'(\mathbf{u})) = -k \partial_x f(\mathbf{u}) \quad (4.14)$$

to recover

$$\begin{aligned} G(u(x_{j-1}, t^n), u(x_j, t^n), u(x_{j+1}, t^n)) &= G(\mathbf{u}) - k \partial_x f(\mathbf{u}) \\ &\quad + \frac{b^2}{2} \left[ \sum_{l=-1}^1 l^2 \partial_l G(\mathbf{u}) u_x \right]_x + \mathcal{O}(b^3). \end{aligned}$$

The exact solution satisfies

$$\begin{aligned} u(x_j, t^{n+1}) &= u + k u_t + \frac{k^2}{2} u_{tt} + \mathcal{O}(k^3) \\ &= u - k \partial_x f(u) + \frac{k^2}{2} (f'(u)^2 u_x)_x + \mathcal{O}(k^3), \end{aligned}$$

where the expansion is centered at  $(x_j, t^n)$  and we use

$$\begin{aligned} u_{tt} &= -\partial_x \partial_t f(u) = -\partial_x (\partial_t f(u)) = -\partial_x (f'(u) \partial_t u) \\ &= \partial_x (f'(u) \partial_x f(u)) = \partial_x (f'(u)^2 \partial_x u). \end{aligned}$$

If we now compute the truncation error, i.e., the error by which the exact solution fails to satisfy the numerical scheme, we recover

$$u(x_j, t^{n+1}) - G(u(x_{j-1}, t^n), u(x_j, t^n), u(x_{j+1}, t^n)) = -k^2 \partial_x [\alpha(\lambda, u) \partial_x u] + \mathcal{O}(k^3),$$

where

$$\alpha(\lambda, u) = \frac{1}{2\lambda^2} \left[ \sum_{l=-1}^1 l^2 \partial_l G(u) - (\lambda f'(u))^2 \right].$$

Since the scheme is monotone,  $\partial_l G(u)$  is positive and we can express it as  $(\sqrt{\partial_l G(u)})^2$ . As for the linear problem discussed previously, we use (4.14) and consider

$$\begin{aligned} (\lambda f'(u))^2 &= \left( \sum_l l \partial_l G(u) \right)^2 = \left( \sum_l l \sqrt{\partial_l G(u)} \sqrt{\partial_l G(u)} \right)^2 \\ &\leq \sum_l l^2 \partial_l G(u) \cdot \sum_l \partial_l G(u) = \sum_l l^2 \partial_l G(u), \end{aligned}$$

by Cauchy-Schwarz and (4.11), guaranteeing that  $\alpha(\lambda, u) \geq 0$ .

An instructive interpretation of this is that when a monotone scheme is used, one solves to  $\mathcal{O}(h^3)$  the modified equation

$$\frac{\partial w}{\partial t} + \frac{\partial f(w)}{\partial x} = k \partial_x [\alpha(\lambda, w) \partial_x w],$$

which is an  $\mathcal{O}(k)$  approximation to the original conservation law. Hence, the approximation can be at best first order accurate. This completes the proof.  $\square$

With this negative result, we are at the end of this development. It is clear that the notion of monotone schemes is attractive but induces severe restrictions on the attainable accuracy of the scheme.

Nevertheless, we remark that monotonicity is the strongest of our constraints. Later, we shall use this observation and show that, by focusing on TVD schemes, some of the accuracy constraints can be lifted, paving the way for the development of higher-order accurate schemes.

## References

- [1] Sergei K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 89(3):271–306, 1959.
- [2] Amiram Harten, James M. Hyman, Peter D. Lax, and Barbara Keyfitz. On finite-difference approximations and entropy conditions for shocks. *Communications on Pure and Applied Mathematics*, 29(3):297–322, 1976.
- [3] Dietmar Kröner. *Numerical Schemes for Conservation Laws*. Wiley, 1997.
- [4] Peter D. Lax and Burton Wendroff. Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237, 1960.
- [5] Randall J. LeVeque. *Numerical Methods for Conservation Laws*, volume 132. Birkhäuser-Verlag, 1992.
- [6] Viktor V. Rusanov. The calculation of the interaction of non-stationary shock waves with barriers. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 1(2):267–279, 1961.

# Chapter 5

# Finite difference methods

With a general understanding of which properties we must require of numerical methods for conservation laws, we now begin a more detailed discussion of a specific class of methods, known as finite difference methods. Such methods are arguably the first family of methods to have been proposed to solve differential equations, reaching back to von Neumann (1903–1957) [22] and even predating the introduction of the digital computer. Indeed, the basic ideas behind finite difference approximations were known to Euler (1707–1783) [21], and the first analysis of finite difference methods for initial value problems was presented in the seminal paper by Courant, Friedrichs, and Lewy (1928) [1] in which the CFL condition was introduced.

The basis of a finite difference scheme is the use of finite difference approximations of derivatives in space and time. The development and analysis of such schemes is a classic area of numerical analysis and we shall not attempt to be exhaustive. For in-depth discussions and analysis we refer to some of the excellent existing texts [5, 15, 11, 10].

Let us define the grid operators

$$\Delta^+ = E_1 - E_0, \quad \Delta^- = E_0 - E_{-1},$$

reflecting down- and upwind differences. As usual  $E_l U_j = U_{j+l}$  is the shift operator. We further define the difference operators

$$D_b^+ = \frac{\Delta^+}{h}, \quad D_b^- = \frac{\Delta^-}{h}, \quad D_b^0 = \frac{\Delta^+ + \Delta^-}{2h},$$

based on the spatial grid  $x_j = x_0 + jh$  defined for  $x \in [x_0, x_0 + L]$  with  $h = L/N$ , where  $N + 1$  represent the number of grid points. Similarly, we have a temporal grid  $t^n = nk$  for  $t \in [0, T]$  and  $k = T/K$  with  $K + 1$  being the number of time steps. The difference operators can be applied in both space and time to represent spatial and temporal derivatives, respectively. In the operators, subscript  $b$  refers to differencing in the spatial direction, while subscript  $k$  will be used to denote temporal differencing.

If we apply these simple differencing formulas to a grid function  $u(x_j)$ , obtained from a smooth function  $u(x)$ , the Taylor expansion yields

$$D_b^\pm u(x_j) = u'(x_j) \pm \frac{h}{2} u''(x_j) + \mathcal{O}(h^2), \quad D_b^0 u(x_j) = u'(x_j) + \frac{h^2}{6} u^{(3)}(x_j) + \mathcal{O}(h^4).$$

We recognize that  $D_b^\pm$  and  $D_b^0$ , respectively, are first and second order accurate approximations to the spatial derivative of a smooth function  $u$  at  $x_j$ .

Consider now the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad (x, t) \in \Omega_x \times \Omega_t,$$

and assume periodic boundary conditions for simplicity. We define the space-time grid  $(x_j, t^n) = (x_0 + jh, nk)$ , and the numerical scheme

$$u_j^{n+1} = G(u_{j-p}^n, \dots, u_j^n, \dots, u_{j+q}^n) = G(u^n)_j.$$

We have changed the arguments from  $U_j^n$  to  $u_j^n$  to indicate that the unknown variables are point values of the solution  $u_j^n \simeq u(x_j, t^n)$ .

Since we have moved from continuous to discrete functions we adopt the definition of the  $p$ -norm ( $0 < p < \infty$ ) as

$$\|u\|_p = \left( \int_{\Omega_x} (u(s))^p ds \right)^{1/p} \Rightarrow \|u\|_{b,p} = \left( b \sum_j (u(s))^p \right)^{1/p}.$$

For  $p = \infty$  we define  $\|\cdot\|_{b,\infty}$  as the max-norm.

When seeking to understand finite difference schemes, we need to gain insight into the behavior of the error,  $\varepsilon_j^n = u_j^n - u(x_j, t^n)$ , as  $h_l, k_l$  approaches zero as  $l \rightarrow \infty$ . The goal of the analysis of finite difference schemes, in fact the goal of the analysis of all computational methods for solving differential equations, is to establish conditions under which we can guarantee that  $\|\varepsilon^n\|_{b,p} \rightarrow 0$  for all  $t$  as  $h_l, k_l$  approaches zero. In such a case we call the scheme convergent. Establishing convergence is the key challenge ahead of us.

## 5.1 • Linear problems

To begin the discussion, let us first restrict attention to the linear hyperbolic problem

$$\frac{\partial u}{\partial t} + A \frac{\partial u}{\partial x} = 0,$$

where  $u = u(x, t) : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  and  $A$  is an  $m \times m$  diagonalizable matrix with real eigenvalues.

We can form schemes directly by using the difference symbols as illustrated in Fig. 5.1. These simple building blocks allow for the construction of schemes of both first and second order accuracy in space or time, as well as both explicit and implicit schemes.

For the scheme in Fig. 5.1(e),  $\sigma$  defines a family of schemes. Taking  $\sigma = h^2/(2k)$  results in the Lax–Friedrichs scheme

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - k A D_b^0 u_j^n,$$

which we already discussed in Chapter 4. Let us take a slightly different approach and consider

$$u(t^{n+1}) = u(t^n) + k \frac{\partial u}{\partial t}(t^n) + \frac{k^2}{2} \frac{\partial^2 u}{\partial t^2}(t^n) + \mathcal{O}(k^3).$$

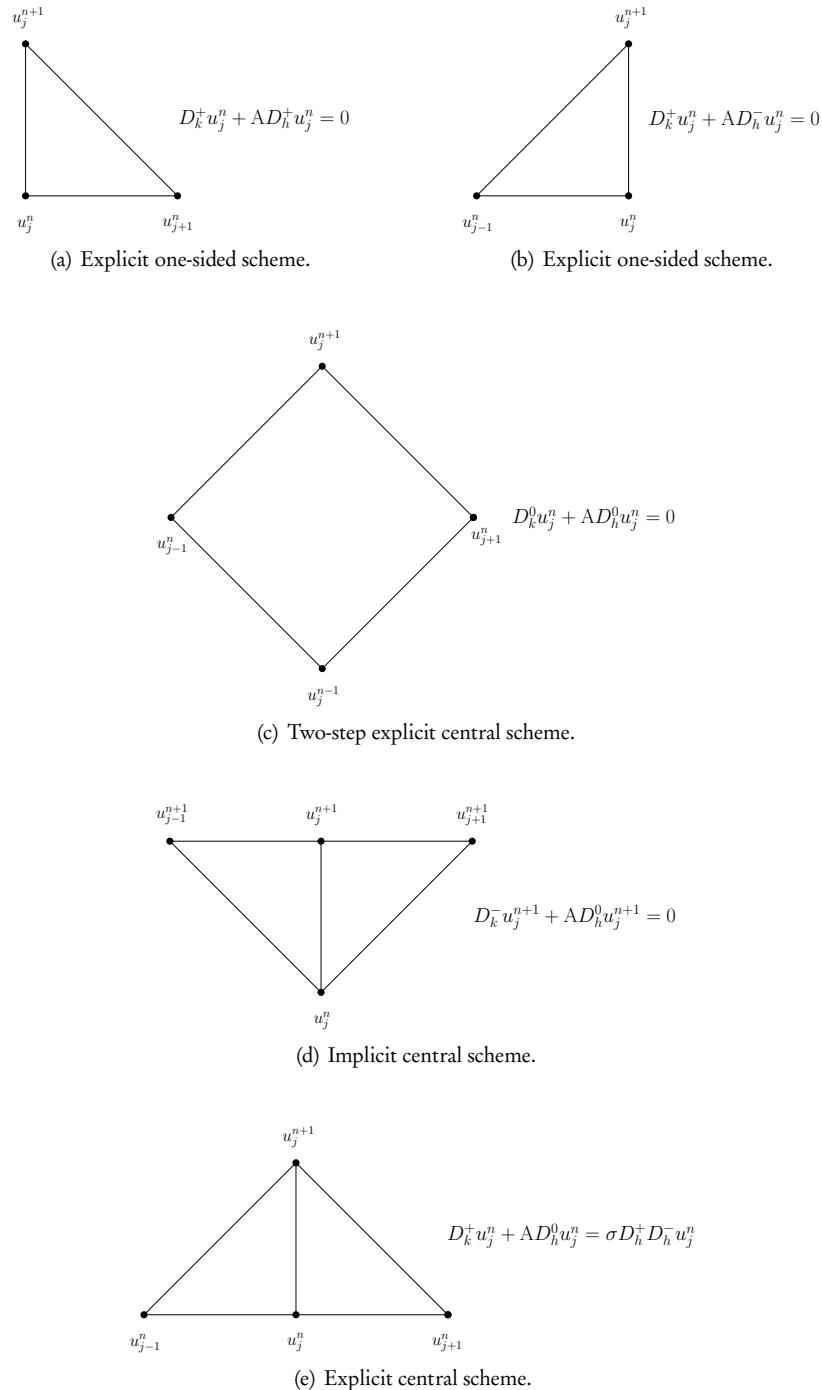


Figure 5.1. Finite difference stencils.

Assuming that  $u(t)$  is sufficiently smooth, we have that

$$\frac{\partial^2 u}{\partial t^2} = -\frac{\partial}{\partial t} A \frac{\partial u}{\partial x} = -\frac{\partial}{\partial x} A \frac{\partial u}{\partial t} = A^2 \frac{\partial^2 u}{\partial x^2},$$

and we recover

$$u_j^{n+1} = u_j^n - k A D_b^0 u_j^n + \frac{k^2}{2} A^2 D_b^+ D_b^- u_j^n.$$

This can be cast in the general form by taking  $\sigma = \frac{k}{2} A^2$ . This latter scheme is recognized as the Lax–Wendroff scheme, encountered in Ex. 4.4.

To gain some insight into the behavior of the different schemes, let us briefly look at the modified equation of the scheme, i.e., we assume that a smooth numerical solution exists and seek an equation for the numerical solution. To ensure consistency as  $h, k$  approach zero, we require that the modified equation coincide with the original equation in this limit.

**Example 5.1.** Consider the linear scalar problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0, \quad (5.1)$$

subject to a smooth initial condition and periodic boundary conditions.

Let us first consider the Lax–Friedrichs scheme

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - k a D_b^0 u_j^n.$$

We seek a smooth solution  $v(x, t)$  that satisfies the numerical scheme exactly at the grid points, i.e.,

$$v(x_j, t^{n+1}) = \frac{1}{2}(v(x_{j+1}, t^n) + v(x_{j-1}, t^n)) - k a \frac{v(x_{j+1}, t^n) - v(x_{j-1}, t^n)}{2h}.$$

By assuming that  $v(x, t)$  is smooth, a Taylor expansion yields

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = \frac{k}{2} \left( \frac{h^2}{k^2} \frac{\partial^2 v}{\partial x^2} - \frac{\partial^2 v}{\partial t^2} \right) + \mathcal{O}(k^2). \quad (5.2)$$

We can simplify this further by differentiating (5.2) with respect to  $t$  and  $x$  and eliminate the cross-term to recover

$$\frac{\partial^2 v}{\partial t^2} = a^2 \frac{\partial^2 v}{\partial x^2} + \mathcal{O}(k).$$

To leading order we can now express the modified equation (5.2) as

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = \frac{k}{2} \left( \frac{h^2}{k^2} - a^2 \right) \frac{\partial^2 v}{\partial x^2} + \mathcal{O}(k^2).$$

If we now define  $\lambda = \frac{ak}{h}$ , which we shall shortly identify as the Courant–Friedrichs–Lowy or the CFL number, we recover

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = \frac{ab}{2\lambda} (1 - \lambda^2) \frac{\partial^2 v}{\partial x^2} + \mathcal{O}(k^2). \quad (5.3)$$

Provided  $|\lambda| \leq 1$ , the numerical solution satisfies an advection-diffusion equation to leading order. This type of equation is known to have smooth solutions [3] and the dissipation smears out steep gradients. This is fully consistent with the results in Ex. 4.4, and the proof of Theorem 4.12.

Now consider the Lax-Wendroff scheme

$$u_j^{n+1} = u_j^n - k a D_b^0 u_j^n + \frac{k^2}{2} a^2 D_b^+ D_b^- u_j^n.$$

The same analysis leads to the modified equation

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = -\frac{ab^2}{6} (1 - \lambda^2) \frac{\partial^3 v}{\partial x^3} + \mathcal{O}(k^3).$$

This equation is fundamentally different from (5.3) since the right-hand side contains a third order term, i.e., it is a dispersive term for which there is no guarantee of smooth decay. Furthermore, the dispersive nature of the term will cause waves to propagate at different speeds, typically resulting in a train of waves, as observed in Ex. 4.4.

Finally, when comparing the two modified equations, we observe that the right-hand side vanishes as  $\mathcal{O}(h)$  for the Lax-Friedrichs scheme, and as order  $\mathcal{O}(h^2)$  for the Lax-Wendroff scheme, indicating that the latter is more accurate. This is also consistent with the observations in Ex. 4.4. ■

### 5.1.1 • Consistency, stability, and convergence

Establishing that a scheme is convergent, i.e.,

$$\lim_{l \rightarrow \infty} \|u(x_j, t^n) - u_j^n(b_l, k_l)\| = 0,$$

is generally a nontrivial task, yet it is a key challenge in the analysis of numerical methods for differential equations. However, if we restrict our attention to linear problems, the analysis is within reach.

Let us again consider the scheme

$$u_j^{n+1} = G(u^n)_j = G(u_{j-1}^n, u_j^n, u_{j+1}^n),$$

where, for simplicity, we restrict attention to schemes that employ only nearest neighbor information, as illustrated in Fig. 5.1. If we assume that the linear problem has a solution  $u(x, t)$  it holds that

$$u(x_j, t^{n+1}) = G(u(t^n))_j + k \tau_j^n.$$

Here we have introduced the truncation error,  $k \tau_j^n$ , as the difference between the exact solution  $u(x, t)$ , evaluated at the grid point, and the numerical solution,  $u_j^n$ . By defining the local error  $\varepsilon_j^{n+1} = u(x_j, t^{n+1}) - G(u^n)_j$  we have  $\varepsilon_j^0 = u(x_j, 0) - u_j^0$  as the error on the initial conditions. We have

$$\varepsilon_j^1 = G(\varepsilon^0)_j + k \tau_j^0,$$

by linearity of  $G(u^n)_j$ . Continuing this iteration, we recover

$$\varepsilon_j^n = G^n(\varepsilon^0)_j + k \sum_{i=0}^{n-1} G^{n-i-1}(\tau^i)_j,$$

where  $G^n = G(G(\dots(G(\cdot))))$ ,  $G^0 = 1$ . If we consider the global error, we have

$$\begin{aligned}\|\varepsilon^n\|_{h,p} &\leq \|G^n(\varepsilon^0)\|_{h,p} + k \sum_{i=0}^{n-1} \|G^{n-i-1}(\tau^i)_j\|_{h,p} \\ &\leq \|G^n\|_{h,p} \|\varepsilon^0\|_{h,p} + kn \max_i \|G^{n-i-1}\|_{h,p} \|\tau^i\|_{h,p},\end{aligned}$$

where we use the standard definition of the subordinate matrix norm

$$\|G^n\|_{h,p} = \sup_{v_j^n} \frac{\|G^n(v^n)_j\|_{h,p}}{\|v_j^n\|_{h,p}}.$$

Recall also that  $kn = t^n$ . We call the scheme consistent if

$$\lim_{l \rightarrow \infty} \left\{ \frac{\|\varepsilon^0\|_{h_l,p}}{\max_i \|\tau^i\|_{h_l,p}} \right\} = 0.$$

Furthermore, we say that a scheme is of order  $(s, t)$  if  $\|\tau\|_{h,p} = \mathcal{O}(h^s, k^t)$ . For example, the Lax–Friedrichs scheme is a  $(1, 1)$  scheme, being first order accurate in both space and time.

Finally, we call the scheme stable if

$$\|G^n\|_{h,p} \leq C,$$

for a sufficiently small value of  $k$ . Under these two conditions, it is immediate that the error is bounded by

$$\|\varepsilon^n\|_{h,p} \leq C(t^n) \max_i \|\tau^i\|_{h,p},$$

which guarantees convergence. This is one part of the celebrated Lax–Richtmyer equivalence theorem [8] which states that, for a well-posed linear problem, solved using a consistent scheme, stability and convergence are equivalent. In other words, there is no such thing as an unstable convergent scheme. The proof that convergence implies stability is a bit more involved, and we refer to [12] for the details.

Hence, to establish convergence of a finite difference scheme, it suffices to establish consistency and stability of the scheme. Turning to consistency first, we need to show that the error in the approximation of the initial conditions and the truncation error vanish when  $h, k$  approach zero. If we assume that the initial condition is sufficiently smooth and that there exists a sufficiently smooth solution to the problem, we can use Taylor expansions to address this question. While consistency of the initial condition is straightforward, an understanding of the truncation error requires us to confirm that the exact solution  $u(x, t)$  satisfies the numerical scheme equation in the limit of  $(h, k)$  vanishing.

**Example 5.2.** Since we assume that the linear system is strongly hyperbolic, it suffices to consider the scalar problem. We consider the general scheme

$$u_j^{n+1} = u_j^n - ak D_b^0 u_j^n + k \sigma D_b^+ D_b^- u_j^n.$$

If the scheme is consistent, the exact solution must satisfy the numerical scheme as

$(h, k)$  vanishes. By the definition of the truncation error

$$k\tau_j^n = u(x_j, t^{n+1}) - G(u(t^n))_j,$$

we recover

$$k\tau_j^n = -k\left(\sigma - \frac{a^2k}{2}\right)\frac{\partial^2 u}{\partial x^2} - k\frac{ab^2}{6}(1 - \lambda^2)\frac{\partial^3 u}{\partial x^3} + \mathcal{O}(kb^3),$$

which results in the truncation error

$$\tau_j^n = -\left(\sigma - \frac{a^2k}{2}\right)\frac{\partial^2 u}{\partial x^2} - \frac{ab^2}{6}(1 - \lambda^2)\frac{\partial^3 u}{\partial x^3} + \mathcal{O}(b^3).$$

First of all we realize that for  $\tau_j^n$  to vanish and ensure consistency,  $\sigma$  must depend on  $h$  or  $k$ . Furthermore, we see that the truncation error is  $\mathcal{O}(h, k)$  provided  $h$  and  $k$  are proportional, as in  $\lambda = ak/h \leq 1$ . However, by choosing  $\sigma = a^2k/2$ , the first term in the truncation error vanishes and we recover a scheme of order  $\mathcal{O}(h^2, k^2)$ . We recognize this particular choice as the Lax–Wendroff scheme and conclude that this is the only second order accurate scheme in this family of schemes. ■

While establishing consistency of a scheme is relatively straightforward, proving its stability is a different matter and at times considerably more challenging.

For the linear scheme, we have the iteration

$$u^{n+1} = Gu^n,$$

where  $G$  is a matrix, called the amplification matrix, that reflects the action of the linear operator  $G(u^n)_j$ . A sufficient condition for stability is

$$\|G\|_{h,p} \leq 1 + \alpha k, \quad (5.4)$$

since

$$\|G^n\|_{h,p} \leq \|G\|_{h,p}^n \leq (1 + \alpha k)^n \leq e^{\alpha nk} \leq e^{\alpha t^n}.$$

However, to establish that a matrix is power bounded is generally far from trivial. A seminal result that enables this is the Kreiss matrix theorem [5, 16].

**Theorem 5.3.** *Assume that we have a family of matrices  $A$  and  $n$  is a natural number. The following conditions are equivalent*

1. *There exists a constant  $C > 0$  such that  $\|A^n\|_{h,p} \leq C$ .*
2. *There exists a constant  $C > 0$  such that*

$$\|(\lambda I - A)^{-1}\|_{h,p} \leq \frac{C}{|\lambda| - 1},$$

*for all  $\lambda \in C$  with  $|\lambda| > 1$ .*

3. There exists a constant  $C > 0$  and a nonsingular matrix  $S$  with  $\|S\|_{h,p}\|S^{-1}\|_{h,p} \leq C$  such that  $T = S^{-1}AS$  is upper triangular and

$$\begin{aligned} 1. \quad |T_{ii}| &\leq 1, & 1 \leq i \leq m, \\ 2. \quad |T_{ij}| &\leq C \min(1 - |T_{ii}|, 1 - |T_{jj}|), & i < j. \end{aligned}$$

4. There exists a constant  $C > 0$  and a matrix  $H$  with  $C^{-1}I \leq H \leq CI$  such that  $A^T H A \leq H$ .

For the proof of this result and a discussion of a number of additional equivalent statements, we refer to [16].

Whereas proving stability for the general case remains a general challenge, we can identify cases which are simpler. For instance, if we assume that  $G$  is uniformly diagonalizable, i.e., there exists a matrix  $S$  with  $\|S\|_{h,p}\|S^{-1}\|_{h,p} \leq C$  for all  $(h,k)$  such that  $\Lambda = S^{-1}AS$  is diagonal, then the Cayley–Hamilton theorem immediately yields

$$\|G^n\|_{h,p} \leq 1 \quad \Leftrightarrow \quad \max_i |\Lambda_{ii}| \leq 1.$$

This statement, often referred to as von Neumann stability, establishes stability for diagonalizable operators. Certain classes of matrices are guaranteed to be uniformly diagonalizable, e.g., symmetric and skew-symmetric matrices. The broadest general class of uniformly diagonalizable matrices are normal matrices, i.e.,  $G^T G = G G^T$ . This follows from the observation that since  $G$  must be diagonalizable by a unitary matrix to be normal, any normal matrix is diagonalizable.

**Example 5.4.** We consider the one-way wave equation

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

with  $a > 0$  and periodic boundary conditions. To discretize the problem we employ the scheme

$$D_k^+ u_j^n + a D_b^- u_j^n = 0 \Rightarrow u_j^{n+1} = u_j^n - \frac{ak}{b} (u_j^n - u_{j-1}^n).$$

In matrix form, this yields the amplification matrix

$$G = \begin{bmatrix} 1 - \lambda & 0 & \dots & 0 & \lambda \\ \lambda & 1 - \lambda & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda & 1 - \lambda \end{bmatrix},$$

with  $\lambda = \frac{ak}{b}$ . One easily shows that  $G$  is a normal  $N \times N$  matrix and, thus, diagonalizable. The eigenvalues,  $\mu(G)$ , are found as the  $N$  roots of the characteristic equation

$$\left( \frac{1 - \lambda - \mu}{-\lambda} \right)^N = 1.$$

Making the substitution of  $y = (1 - \lambda - \mu)/(-\lambda)$ , we seek the solution to

$$y^N = 1 \Rightarrow y_l = \exp\left(i \frac{2\pi}{N} l\right), \quad l = 0, \dots, N-1,$$

to recover

$$\mu_l(G) = 1 - \lambda + \lambda \exp\left(i \frac{2\pi}{N} l\right), \quad l = 0, \dots, N-1.$$

There are only two real eigenvalues:

$$\mu_0(G) = 1 - 2\lambda, \quad \mu_{N/2}(G) = 1.$$

Furthermore, it is easy to see that  $|1 - 2\lambda| \leq |\mu_l(G)| \leq 1$ , provided  $\lambda \leq 1$ . Hence, von Neumann stability requires that  $0 \leq \lambda \leq 1$ , which is a condition we have previously encountered.

If we consider the case of  $\alpha < 0$ , then  $\lambda < 0$  and stability is impossible. This highlights that care must be exercised when choosing the method for a particular problem. ■

While the analysis of eigenvalues of the amplification matrix is often very useful to understand stability properties of more complex schemes, it is worth emphasizing that the von Neumann stability in many cases is only a necessary, but not a sufficient, condition for stability.

**Example 5.5.** Consider the matrix

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

It is clear that the eigenvalues of  $A$  are both one, indicating von Neumann stability. However, if we compute the power of  $A$  we obtain

$$A^n = \begin{bmatrix} 1 & n \\ 0 & 1 \end{bmatrix},$$

which is clearly not power bounded. The iteration would therefore be unstable, even if it is von Neumann stable. This results from  $A$  not being diagonalizable, as it can be expressed as  $A = I + J$  where  $J$  is a Jordan block. ■

If we now further restrict ourselves to cases where  $G$  is of Toeplitz form, the analysis simplifies considerably since the Fourier transform diagonalizes a Toeplitz matrix. At first glance, this may seem overly restrictive. However, a closer inspection reveals that, provided the solution is periodic, all the schemes in Fig. 5.1 result in amplification matrices in Toeplitz form.

Hence, we assume that the spatial variation of the numerical solution can be expressed as the discrete Fourier series

$$u_b(x, t^n) = \sum_{|l| \leq \infty} \hat{u}_l^n \exp\left(i 2\pi l \frac{x}{L}\right),$$

where the coefficients  $\hat{u}_l^n$  are defined by requiring that  $u_b(x_j, t^n) = u_j^n$ . If we can guarantee that all  $\hat{u}_l^n$  remain bounded, stability for  $u_b(x, t)$  follows from Parseval's

identity

$$\|u_b(t^n)\|_b^2 = 2\pi \sum_l |\hat{u}_l^n|^2.$$

The path forward is therefore to consider numerical solutions in the form

$$u_j^n = \hat{u}_l^n \exp\left(i2\pi l \frac{x_j}{L}\right),$$

and analyze the scheme

$$\hat{u}^{n+1} = \hat{G} \hat{u}^n,$$

where  $\hat{G}$  is the symbol of the scheme or, equivalently, the Fourier transform of the amplification matrix. It is important to realize that we have eliminated the dependence of  $h$  on the size of  $\hat{G}$ , which now depends only on the size of the original system of equations and the number of steps in the time-stepping scheme.

Combining (5.4) and linearity, it is clear that stability requires  $\|\hat{G}^n\|_{h,p} \leq 1 + \alpha k$ . Let us illustrate the steps that need to be taken to establish this through an example.

**Example 5.6.** Since we restrict attention to the strongly hyperbolic problem, nothing is lost by focusing the attention on the scalar problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

with  $a > 0$  and periodic boundary conditions. For simplicity we assume that  $L = 2\pi$  and consider the general scheme

$$u_j^{n+1} = u_j^n - ak D_b^0 u_j^n + k\sigma D_b^+ D_b^- u_j^n.$$

Seeking solutions of the form

$$u_j^n = \hat{u}_l^n \exp(ilx_j),$$

we recover

$$\hat{G} = 1 + i\lambda \sin(lh) - 4\nu \sin^2\left(\frac{lh}{2}\right),$$

where  $\lambda = \frac{ak}{h}$  and  $\nu = \frac{\sigma k}{h^2}$ . Now consider

$$|\hat{G}| = \left(1 - 4\nu \sin^2\left(\frac{lh}{2}\right)\right)^2 + \lambda^2 \sin^2(lh).$$

We first of all realize that for  $\nu = 0$ , i.e.,  $\sigma = 0$ ,  $|\hat{G}| \geq 1$  and the scheme is unstable.

Using the identities  $\sin(2x) = 2\sin(x)\cos(x)$  and  $\cos^2(x) + \sin^2(x) = 1$ , we recover the condition for stability

$$(16\nu^2 - 4\lambda^2)\beta^2 + (4\lambda^2 - 8\nu)\beta \leq 0,$$

where  $\beta = \sin^2(lh/2)$  and  $0 \leq \beta \leq 1$ .

If we first consider the case where  $(16\nu^2 - 4\lambda^2) \leq 0$ , requiring that  $\nu \leq \lambda/2$ , the stability condition is a downward parabola. We must require that the maximum is negative, i.e.,

$$\frac{-(4\lambda^2 - 8\nu)}{2(16\nu^2 - 4\lambda^2)} \leq 0.$$

Since both  $\lambda$  and  $\nu$  are positive, the worst case scenario is  $\nu = \lambda/2$ , resulting in the condition that  $\lambda \leq 1$ .

If we consider the other case of  $\nu \geq \lambda/2$ , then  $(16\nu^2 - 4\lambda^2) \geq 0$ . To have any hope of stability, we must require that  $(4\lambda^2 - 8\nu) \leq 0$ . Since the worst case scenario is for  $\beta = 1$ , we recover the condition that  $2\nu - 1 \leq 0$  or  $\nu \leq 1/2$ , which again implies that  $0 \leq \lambda \leq 1$ .

To connect this to the previous discussion of the general scheme, recall that  $\sigma = \frac{b^2}{2k}$  yields the Lax-Friedrichs scheme. This implies that  $\nu = 1/2 \geq \lambda/2$ , provided that  $\lambda \leq 1$ . For the Lax-Wendroff scheme, we have  $\sigma = \frac{a^2 k}{2}$ , which results in  $\nu \leq \frac{\lambda}{2}$ , requiring again that  $\lambda \leq 1$ .

Whereas this kind of analysis can be algebraically complex, the process is straightforward and systematic. The extension to the system case with a matrix  $A$  requires that the stability condition, in this case  $\lambda \leq 1$ , holds for all eigenvalues of  $A$ , i.e.,

$$\forall \mu_i(A): \frac{k|\mu_i(A)|}{h} \leq 1,$$

where  $\mu_i(A)$  represents the  $i$ th real eigenvalues of  $A$ . ■

As has become clear during the discussion, one often recovers the requirement  $|\lambda| \leq 1$  to ensure stability. It is instructive to understand the importance and physical underpinning of this condition. Let us rewrite it as

$$|a| \frac{k}{h} \leq 1, \quad (5.5)$$

and recall that  $a$  is the velocity by which the wave propagates. Likewise, we can define a numerical velocity as  $v_{num} = \frac{b}{k}$  and express (5.5) as

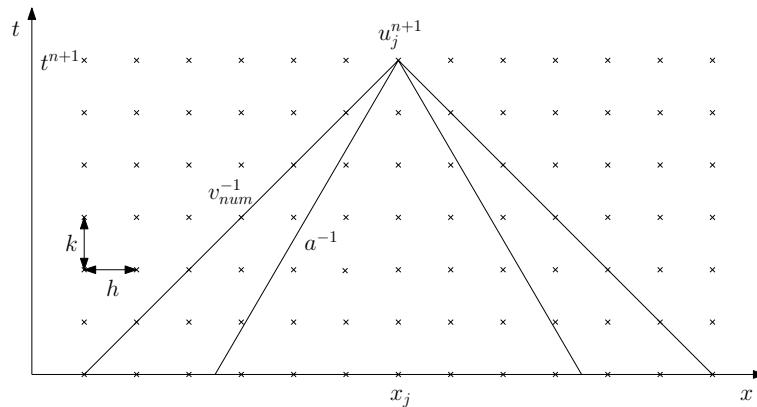
$$v_{num} \geq |a|,$$

i.e., the numerical velocity must be larger than its physical counterpart. As discussed in Chapter 2 and illustrated in Fig. 5.2, this implies that the domain of dependence of the numerical scheme must contain the domain of dependence of the physical problem.

This condition, which is both natural and intuitive, is the celebrated CFL condition, named after the seminal paper of Courant, Friedrichs, and Lewy [1]. This paper is regarded as the first paper in which the analysis of numerical schemes for partial differential equations is pursued. This condition is often only a necessary condition as additional conditions may be needed to guarantee stability, e.g., as in Ex. 5.6, where  $\sigma > 0$  is needed.

### 5.1.2 • Nonsmooth problems

So far, the discussion has focused on linear problems with smooth solutions. A first step toward nonlinear problems is to understand the accuracy that can be expected



**Figure 5.2.** Illustration of the CFL condition, highlighting that the domain of dependence of the numerical scheme must contain that of the physical problem. This implies that  $|a|^{-1} \geq v_{num}^{-1}$  or  $|a|v_{num}^{-1} \leq 1$  with  $v_{num} = h/k$  expressing the velocity of the numerical method.

for linear problems with discontinuous solutions. To pursue this, let us consider an example, borrowed from [9].

**Example 5.7.** We consider the linear problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

for  $x \in \mathbb{R}$  and with the initial condition

$$u_0(x) = \begin{cases} 1, & x \leq 0, \\ -1, & x > 0. \end{cases}$$

Its exact solution is  $u(x, t) = u_0(x - at)$ . We solve the problem using the Lax–Friedrichs scheme

$$u_j^{n+1} = \frac{u_{j+1}^n + u_{j-1}^n}{2} - ak \frac{u_{j+1}^n - u_{j-1}^n}{2h},$$

which is  $\mathcal{O}(h, k)$  accurate when  $u(x, t)$  is smooth.

Reaching back to Ex. 5.1, we recall that the numerical solution,  $u_h(x, t)$  satisfies the modified equation

$$\frac{\partial u_h}{\partial t} + a \frac{\partial u_h}{\partial x} = \varepsilon \frac{\partial^2 u_h}{\partial x^2},$$

where  $\varepsilon = \mathcal{O}(h)$ . To seek its solution, let us first introduce a new variable  $v(\xi, \tau) = u_h(x - at, t)$ . One easily sees that  $v(\xi, \tau)$  must satisfy

$$\frac{\partial v}{\partial \tau} = \varepsilon \frac{\partial^2 v}{\partial \xi^2}.$$

Recalling Green's function for the heat equation

$$\phi(\xi, \tau) = \frac{1}{\sqrt{4\epsilon\pi\tau}} \exp\left(\frac{-\xi^2}{4\epsilon\tau}\right),$$

we recover the solution by convolution as

$$v(\xi, \tau) = \frac{2}{\sqrt{4\epsilon\pi\tau}} \int_{-\infty}^0 \exp\left(-\left(\frac{(\xi-y)}{\sqrt{4\epsilon\tau}}\right)^2\right) dy.$$

Integration by substitution yields

$$v(\xi, \tau) = \frac{2}{\sqrt{\pi}} \int_z^\infty \exp(-y^2) dy = \text{erfc}(z),$$

where  $z = \xi/\sqrt{4\epsilon\tau}$  and  $\text{erfc}(z)$  is the complementary error function. If we now measure the  $L^1$ -error we obtain

$$\begin{aligned} \|v_0(\xi) - v(\xi, \tau)\|_1 &= 2 \int_{-\infty}^0 \left(1 - \text{erfc}\left(\frac{\xi}{\sqrt{4\epsilon\tau}}\right)\right) d\xi \\ &= 2\sqrt{4\epsilon\tau} \int_{-\infty}^0 (1 - \text{erfc}(z)) dz = 4\sqrt{\epsilon\tau}. \end{aligned}$$

Recall that  $\epsilon = \mathcal{O}(h)$  for the Lax–Friedrichs scheme such that

$$\|u_0(x - at) - u_h(x, t)\|_1 = C\sqrt{ht},$$

i.e., the accuracy is  $\mathcal{O}(h^{1/2})$ . ■

As established in Theorem 4.12, the accuracy of the monotone scheme is limited to first order. However, this example indicates that we may generally expect a poorer accuracy for problems with discontinuous solutions. Indeed, the general result is as follows [19].

**Theorem 5.8.** *Assume that the monotone linear finite difference scheme*

$$u_j^{n+1} = \sum_{l=-p}^q c_l(\lambda) u_j^n,$$

*with  $\lambda = ak/h$ , is a consistent approximation to a linear scalar one-dimensional conservation law. Then for any constant  $M > 0$ ,  $h$  sufficiently small, and  $\|u_0\|_{TV} \leq M$ , it holds that*

$$\begin{aligned} s(p, q)M \sum_{\substack{l=-p \\ l \neq l_0}}^q \sqrt{c_l(\lambda)} \sqrt{\frac{t^n}{\lambda}} \sqrt{b} &\leq \|u(\cdot, t^n) - u^n\|_{h,1} \\ &\leq M \left[ 2 \sqrt{\sum_{l=-p}^q l^2 c_l(\lambda) - \lambda^2 a^2} \sqrt{\frac{t^n}{\lambda}} \sqrt{b} + b \right], \end{aligned}$$

*where  $s(p, q) > 0$  is a constant,  $l_0 = \arg \max_l c_l(\lambda)$ , and  $\|\cdot\|_{TV}$  signifies the TV-norm.*

## 5.2 ■ Nonlinear problems

Let us now return to nonlinear conservation laws and develop suitable finite difference schemes for such problems. As can be expected, such developments are at times more complex and less general than what is possible for the linear problems.

The most natural approach is to generalize what we have developed for the linear case to the nonlinear case. Thus, we consider the conservation form

$$u_j^{n+1} = u_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n].$$

The question is how to express the numerical flux for the nonlinear problem. As we experienced for the linear case, stability is a central concern. However, if the scheme is monotone, Theorem 4.5 ensures stability and recovery of the entropy solution.

**Example 5.9.** In Ex. 4.4 we considered Burgers' equation with the flux

$$F(u, v) = \begin{cases} f(u), & s \geq 0, \\ f(v), & s < 0, \end{cases} \quad s = \frac{f(u) - f(v)}{u - v}, \quad (5.6)$$

where  $s$  is the speed given by the Rankine–Hugoniot condition. This flux is the one-dimensional version of what is known as the Roe flux. In Ex. 4.4 we observed that this may produce an entropy violating solution for the situation where  $u_l \leq 0 \leq u_r$ . In this case the entropy condition requires that  $f'(u_s) = 0$ , i.e., the center of the rarefaction wave cannot move. The point of  $u_s$  where  $f'(u_s) = 0$  is called the sonic point.

A slightly modified flux that overcomes this problem is given as [9]

$$F(u, v) = \begin{cases} \max_{u_l \leq u \leq u_r} f(u), & u_l > u_r, \\ \min_{u_l \leq v \leq u_r} f(v), & u_r \geq u_l. \end{cases}$$

One quickly realizes that the entropy violating solution appears for  $u_l \leq 0 \leq u_r$ , such that  $f(0)$  is the sonic solution, provided the flux is convex.

Another approach to overcome this problem, proposed in [14], is to use the Roe flux (5.6) everywhere except where  $f'(u_{j+1}^n) \cdot f'(u_j^n) \leq 0$ . At these places, the Lax–Friedrichs flux can be applied. Since the Lax–Friedrichs flux is monotone, this overcomes the problem, provided the flux is convex. ■

Consider the Lax–Friedrichs flux

$$F(u, v) = \frac{f(u) + f(v)}{2} - \frac{\alpha}{2}(v - u),$$

which, as discussed in Ex. 4.6, is monotone, provided  $\alpha \geq \max_w |f'(w)|$ . A slightly less dissipative but still monotone scheme is the local Lax–Friedrichs (or Rusanov) scheme for which  $\alpha \geq \max_{w \in [u, v]} |f'(w)|$ , i.e., we seek a local maximum rather than a global maximum of the wave speed.

For the Lax–Wendroff flux, the extension to the nonlinear case is less immediate. Consider

$$u(t^{n+1}) = u(t^n) + k \frac{\partial}{\partial t} u(t^n) + \frac{k^2}{2} \frac{\partial^2}{\partial t^2} u(t^n) + \mathcal{O}(k^3).$$

Now we utilize that

$$\frac{\partial}{\partial t} u = -\frac{\partial f(u)}{\partial x}, \quad \frac{\partial^2}{\partial t^2} u = \frac{\partial}{\partial x} \left( f'(u) \frac{\partial f(u)}{\partial x} \right),$$

and the approximation

$$\frac{\partial f(x_j, t^n)}{\partial x} = \frac{f(u_{j+1}^n) - f(u_{j-1}^n)}{2h} + \mathcal{O}(h^2)$$

to recover the Lax–Wendroff numerical flux as

$$F(u, v) = \frac{f(u) + f(v)}{2} - \frac{k}{2h} f'(\bar{u})(f(v) - f(u)), \quad \bar{u} = \frac{u + v}{2}.$$

The extension of these formulations to the system case is straightforward and requires only a minor reformulation. For the Lax–Friedrichs scheme,  $\alpha$  must include the dynamics of the multiple waves as

$$\alpha \geq \max_w (\mu_{max}), \quad \mu_{max} = \max_l |\mu_l(\nabla_u f(w))|,$$

where the maximum of  $w$  is taken globally or locally. Similarly, the extension to the Lax–Wendroff scheme is obtained by replacing  $f'(\bar{u})$  with the flux Jacobian evaluated at  $\bar{u}$ .

The extension and generalization of other fluxes to the system case is slightly more complicated and often leads to problem-specific formulations. We revisit this in more detail in section 6.2.

Considering the accuracy of the monotone finite difference scheme for scalar conservation laws, the primary result is due to Kuznetsov [7],

**Theorem 5.10.** *The approximate solution generated by a monotone, conservative, and consistent numerical scheme to the scalar conservation law with a Lipschitz-continuous numerical flux converges to the entropy solution for any initial condition. Furthermore, if  $\|u_0\|_{TV}$  is bounded, we have*

$$\|u(t^n) - u^n\|_{b,1} \leq C(t^n) \sqrt{h}.$$

The sharpness of this result is proven in [13] through the construction of a particular example. This result is in line with the result for the linear case in Theorem 5.8 and, thus, not a surprise. Results for the multidimensional scalar conservation laws are given in [2].

For special cases, slightly better convergence rates can be expected. If one restricts the attention to the case of finitely many noninteracting shocks, [4] shows that the error is  $\mathcal{O}(h)$  away from the shocks and  $\mathcal{O}(h^\gamma)$ ,  $\gamma < 1$ , in the neighborhood of each shock. This result holds also for systems. For scalar conservation laws with a convex flux and initial conditions which contain all shocks, the  $L^1(\Omega_x)$ -error is  $\mathcal{O}(h|\log h|)$  [6]. Further work shows  $\mathcal{O}(h)$  for piecewise smooth solutions [20]. An extensive theory for pointwise accuracy has been developed in [17, 18].

## 5.3 ▪ Finite difference methods in action

So far, focus has been on the development of a thorough understanding of the properties of finite difference methods when used to solve conservation laws. However, true understanding of the methods is not achieved until the methods are brought to life through an actual implementation. In the following we present and discuss implementations of the basic schemes and demonstrate their performance for both linear and nonlinear problems.

While we discuss the particular algorithms with a focus on the test cases introduced in section 1.4, the structure of all implementations is the same and comprises three main pieces. The driver routine initializes the grid, the initial conditions, and other parameters of global importance. This calls the outer routine, which advances the problem in time based on the temporal integration of the numerical scheme. As we shall soon find, only the last routine will require substantial changes when a new equation or scheme is considered. Such a structure is both general and flexible enough to accommodate a wide range of problems and schemes.

We consider schemes in conservation form

$$u_j^{n+1} = u_j^n - \lambda (F_{j+1/2}^n - F_{j-1/2}^n),$$

where

$$F_{j+1/2}^n = F(u_j^n, u_{j+1}^n), \quad F_{j-1/2}^n = F(u_{j-1}^n, u_j^n).$$

The numerical flux  $F(u, v)$  must be specified for the problem at hand.

### 5.3.1 ▪ Boundary conditions

Before proceeding, we need to address the question of boundary conditions. Since we restrict ourselves to Cartesian grids, it is most natural to define an extended grid and define external grid values in such a way that the desired boundary conditions are imposed.

For the monotone schemes with  $p + q = 1$ , we need only extend the grid with one cell at each end of the computational grid. To impose the boundary conditions we utilize a mirror principle. If we assume that  $x_0$  is the first physical cell with  $U_0^n$  being the boundary value, we define

$$U_{-1}^n = -U_1^n + 2g(t),$$

for a Dirichlet boundary condition  $u(x_0, t) = g(t)$ . Likewise, for a homogeneous Neumann boundary condition we define  $U_{-1}^n = U_1^n$ . Finally, for a periodic boundary condition, we use  $U_{-1}^n = U_{N-1}^n$ . The extension and definition of boundary conditions is facilitated by `extend.m`

**Script 5.1. `extend.m`: Routine to impose boundary conditions on scalar functions by extension.**

---

```
function [xe, ue] = extend(x, u, h, m, BC1, ul, BCr, ur)
% Purpose: Extend dependent and independent vectors (x, u), by m cells
% subject to appropriate boundary conditions.
% BC = "D" - Dirichlet; BC = "N" - Neumann; BC = "P" - periodic
% ul/ur - BC value - only active for Dirichlet BC
```

```

x1 = min(x); xr = max(x); N = length(u);
xe = zeros(N+2*m,1); ue = zeros(N+2*m,1); q = [1:m];

% Extend x
xe(m-q+1) = x1-q*h; xe(N+m+q) = xr + q*h; xe((m+1):(N+m)) = x(1:N);

% Periodic extension of u
if (BCl=='P') | (BCr=='P')
    ue(m-q+1) = u(N-q); ue(N+m+q) = u(q+1); ue((m+1):(N+m)) = u(1:N);
    return;
end;

% Left extension
if BCl=='D'
    ue(m-q+1) = -u(q+1)+2*u1;
else
    ue(m-q+1) = u(q+1);
end

% Right extension
if BCr=='D'
    ue(N+m+q) = -u(N-q)+2*ur;
else
    ue(N+m+q) = u(N-q);
end
ue((m+1):(N+m)) = u(1:N);
return

```

---

### 5.3.2 ▪ Linear wave equation

We begin by considering the linear wave problem

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

with the initial condition given in (1.6).

In LinwaveMDriver1D.m we define the driver. It sets the initial condition and specifies the computational domain, the grid size  $h$ , and the CFL number. The linear wave equation is integrated in time through LinwaveM1D.m.

**Script 5.2.** *LinwaveMDriver1D.m: Driver routine for solving the linear wave equation using a conservative finite difference method.*

---

```

% Driver script for solving the 1D wave equation using a monotone scheme
clear all

% Set problem parameters
L = 2; FinalTime = 4.0; N = 2048; h = L/N; CFL = 0.90;

% Define domain and initial conditions
x = [-1:h:1]';
[u] = wavetest(x,0.5,-0.7,0.005,10,log(2)/(36*0.005^2));

% Solve Problem
[u] = LinwaveM1D(x,u,h,CFL,FinalTime);

```

---

---

**Script 5.3.** *LinwaveM1D.m: Temporal integration routine for solving the linear wave equation using a conservative finite difference method.*

---

```
function [u] = LinwaveM1D(x,u,h,CFL,FinalTime)
% function [u] = LinwaveM1D(x,u,h,CFL,FinalTime)
% Purpose : Integrate 1D wave equation until using a monotone scheme.
time = 0; tstep = 0;

% Set the time step
k = CFL*h;

% Integrate scheme
while (time<FinalTime)
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    u = u + k*LinwaveMrhs1D(x,u,h,k,1);
    time = time+k; tstep = tstep+1;
end
return
```

---

Calling LinwaveMrhs1D.m returns the computation of the right-hand side of the linear wave equation and boundary conditions are imposed.

---

**Script 5.4.** *LinwaveMrhs1D.m: Definition of right-hand side for the linear wave equation*

---

```
function [du] = LinwaveMrhs1D(x,u,h,k,maxvel)
% function [du] = LinwaveMrhs1D(x,u,b,k,maxvel);
% Purpose: Evaluate right hand side for linear wave equation
% using a monotone method

N = length(x);
% Periodic boundary conditions
[xe,ue] = extend(x,u,h,1,'P',0,'P',0);

% Compute RHS - Change numerical flux here
du = -(LinwaveLF(ue(2:N+1),ue(3:N+2),k/h,maxvel) - ...
        LinwaveLF(ue(1:N),ue(2:N+1),k/h,maxvel))/h;
return
```

---

The only routine that remains problem specific is the one specifying the numerical flux. We consider the global Lax-Friedrichs flux (LinwaveLF.m) and the Lax-Wendroff flux (LinwaveLW.m).

---

**Script 5.5.** *LinwaveLF.m: Definition of the Lax-Friedrichs numerical flux,  $F(u,v)$  for the linear wave equation.*

---

```
function [numflux] = LinwaveLF(u,v,lambda,maxvel)
% function [numflux] = LinwaveLF(u,v,lambda,maxvel);
% Purpose: Evaluate Lax Friedrich numerical flux for wave equation

numflux = (u+v)/2 - maxvel/2*(v-u);
end
```

---

**Script 5.6.** *LinwaveLW.m: Definition of the Lax–Wendroff numerical flux,  $F(u, v)$  for the linear wave equation.*

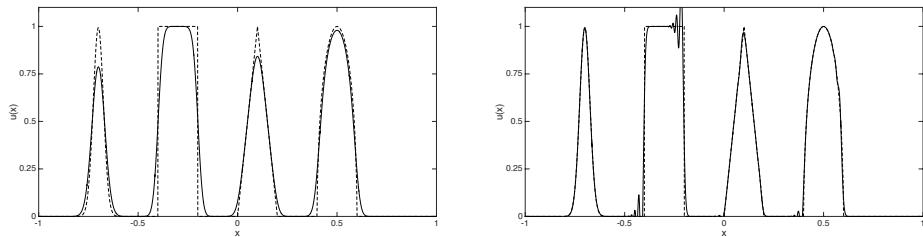
---

```
function [ numflux ] = LinwaveLW( u , v , lambda , maxvel )
% function [ numflux ] = LinwaveLW( u , v , lambda , maxvel );
% Purpose: Evaluate Lax–Wendroff numerical flux for wave equation

numflux = (u+v)/2 - lambda/2*(v-u);
end
```

---

In Fig. 5.3 we illustrate the solutions at  $T = 4$  for  $N = 2048$  grid points, computed using the Lax–Friedrichs scheme and the Lax–Wendroff scheme, respectively. We clearly observe the dissipative behavior of the first order Lax–Friedrichs scheme and the substantially improved accuracy obtained with the second order Lax–Wendroff scheme. However, it is also clear that this is obtained at the cost of introducing artificial oscillations in the neighborhood of the nonsmooth parts of the solution.



**Figure 5.3.** *Solution of the linear wave problem at  $T = 4$  with  $N = 2048$ . On the left is shown the solution obtained with the Lax–Friedrichs scheme while the result on the right is obtained with the Lax–Wendroff scheme. The dashed line shows the exact solution.*

While the convergence of the Lax–Friedrichs scheme is only first order, the scheme does indeed converge to the exact solution, as shown in Fig. 5.4.

### 5.3.3 • Burgers' equation

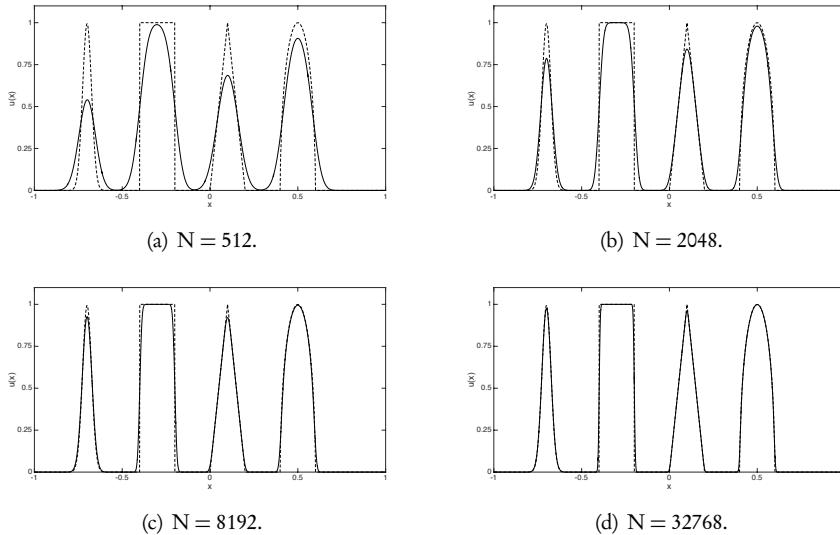
We consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

and seek a solution with

$$k = \text{CFL} h / \alpha, \quad \alpha = \max_u |f'(u)|.$$

In BurgersMDriver1D.m we illustrate the driver, setting the initial condition and defining the computational domain, the grid size  $h$ , and the CFL number.



**Figure 5.4.** Solution of the linear wave problem at  $T = 4$  obtained with the Lax-Friedrichs scheme with increasing resolution  $N$ . The dashed line shows the exact solution.

**Script 5.7. BurgersMDriver1D.m:** Driver routine for solving Burgers' equation using a conservative finite difference method.

---

```
% Driver script for solving the 1D Burgers equation using a monotone scheme
clear all

% Set problem parameters
L = 2; FinalTime = 0.5; N = 1024; h = L/N; CFL = 0.90;

% Define domain and initial conditions
x = [0:h:1]';

u = sin(2*pi*x); %periodic BC needed
% u = (1-sign(x-0.2))/2+1; % Constant BC needed

% Solve Problem
[u] = BurgersM1D(x,u,h,CFL,FinalTime);
```

---

Burgers' equation is integrated in time using BurgersM1D.m. The time step,  $k$ , is set by continuously computing  $|f'(u)|$  to evaluate the maximum velocity of the system. This is the only problem specific part of this routine.

**Script 5.8. BurgersM1D.m:** Temporal integration routine for solving Burgers' equation using a conservative finite difference method.

---

```
function [u] = BurgersM1D(x,u,h,CFL,FinalTime)
% function [u] = BurgersM1D(x,u,h,CFL,FinalTime)
% Purpose : Integrate 1D Burgers equation until FinalTime
% using a monotone scheme.
time = 0; tstep = 0;

% integrate scheme
```

---

```

while (time<FinalTime)
  % Decide on timestep
  maxvel = max(2*abs(u)); k = CFL*h/maxvel;
  if (time+k>FinalTime) k = FinalTime-time; end
  % Update solution
  u = u + k*BurgersMrhs1D(x,u,h,k,maxvel);
  time = time+k; tstep = tstep+1;
end
return

```

---

Calling `BurgersMrhs1D.m` returns the computation of the right-hand side of Burgers' equation and boundary conditions are set.

**Script 5.9.** *BurgersMrhs1D.m: Definition of the right-hand side for Burgers' equation.*

---

```

function [du] = BurgersMrhs1D(x,u,h,k,maxvel)
% function [du] = BurgersMrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for Burgers equation
% using monotone method
N = length(x);

[xe,ue] = extend(x,u,h,1,'P',0,'P',0); % Periodic boundary conditions
%[xe,ue] = extend(x,u,h,1,'D',2,'N',0); % Constant boundary conditions

% Change numerical flux here
du = - (BurgersLF(ue(2:N+1),ue(3:N+2),0,maxvel) - ...
         BurgersLF(ue(1:N),ue(2:N+1),0,maxvel))/h;
return

```

---

The only routine that is truly problem specific is the one specifying the numerical flux as this depends on  $f(u)$ . We illustrate the global Lax–Friedrichs flux (`BurgersLF.m`), the Lax–Wendroff flux (`BurgersLW.m`), and the Roe flux (`BurgersRoe.m`) in the following.

**Script 5.10.** *BurgersLF.m: Definition of the Lax–Friedrichs numerical flux,  $F(u,v)$ , for Burgers' equation.*

---

```

function [numflux] = BurgersLF(u,v,lambda,maxvel)
% function [numflux] = BurgersLF(u,v,lambda,maxvel);
% Purpose: Evaluate the Lax Friedrich numerical flux for Burgers equation

fu = u.^2; fv = v.^2;
numflux = (fu+fv)/2 - maxvel/2*(v-u);
end

```

---

**Script 5.11.** *BurgersLW.m: Definition of the Lax–Wendroff numerical flux,  $F(u,v)$ , for Burgers' equation.*

---

```

function [numflux] = BurgersLW(u,v,lambda,maxvel)
% function [numflux] = BurgersLW(u,v,lambda,maxvel);
% Purpose: Evaluate the Lax Wendroff numerical flux for Burgers equation

fu = u.^2; fv = v.^2; alpha = lambda*2*(u+v)/2;
numflux = (fu+fv)/2 - alpha/2.*(fv-fu);
end

```

---

**Script 5.12.** *BurgersRoe.m: Definition of the Roe numerical flux,  $F(u, v)$ , for Burgers' equation. No sonic fix is included.*

---

```
function [ numflux ] = BurgersRoe(u, v, lambda, maxvel)
% function [numflux] = BurgersRoe(u, v, lambda, maxvel);
% Purpose: Evaluate Roe numerical flux for Burgers equation.
% No sonic fix.

fu = u.^2; fv = v.^2; alpha = u+v;
numflux = (alpha>=0).* fu + (alpha<0).* fv ;
end
```

---

To evaluate the performance of the schemes, consider the case of

$$u_0(x) = \begin{cases} 2, & x \leq 0.2, \\ 1, & x > 0.2, \end{cases}$$

where the exact solution is  $u(x, t) = u_0(x - 3t)$ . In Table 5.1 we list the errors, measured in the discrete  $L^1$ -norm, for three different choices of the numerical flux. All three schemes result in a first order accuracy with only minor quantitative differences in the results. It also shows that Theorem 5.10, indicating an error of  $\mathcal{O}(h^{1/2})$  may, in some cases, be overly conservative, and confirms the theoretical result that  $\mathcal{O}(h)$  should be expected for piecewise smooth solutions [20] to scalar conservation laws.

**Table 5.1.** *Errors for Burgers' equation at  $T = 0.2$ , solved using a conservative finite difference method with different choices of the numerical flux. The error is measured in  $\|\cdot\|_{h,1}$ .*

N	Lax-Friedrichs	Lax-Wendroff	Roe flux
128	1.1e-2	6.9e-3	5.5e-3
512	2.7e-3	1.1e-3	1.3e-3
2048	7.0e-4	4.4e-4	3.5e-4
8192	1.7e-4	7.4e-5	8.5e-5
32768	4.3e-5	2.8e-5	2.1e-5

### 5.3.4 • Maxwell's equations

Let us now consider Maxwell's equations and the example of a one-dimensional cavity, possibly filled with two materials. The details of the problem and the exact solution are given in subsection 1.4.1.

As for Burgers' equation, the implementation contains a few subroutines, all being initiated by *MaxwellMDriver1D.m* which defines the setup, including the domain, the final time, and the CFL number for the simulation. It also defines the setup of the cavity materials and computes the initial conditions by calling *CavityExact.m*, which implements the exact solution discussed in subsection 1.4.1.

**Script 5.13.** *MaxwellMDriver1D.m: Driver routine for solving Maxwell's equations using a conservative finite difference method.*

---

```
% Driver script for solving the 1D Maxwell's equations using a monotone scheme
clear all
```

```
% Set problem parameters
```

```

L = 2; FinalTime = pi/2; N = 1024; h = L/N; CFL = 0.90;
ep1 = 1.0; mul = 1.0; epr = 2.25; mur = 1.0;

% Define domain, materials and initial conditions
x = [-1:h:1]';
[Ef Hf ep mu] = CavityExact(x, ep1, epr, mul, mur, 0);

% Solve Problem
EM = [Ef Hf];
[EM] = MaxwellM1D(x, EM, ep, mu, h, CFL, FinalTime);

```

---

**Script 5.14. *CavityExact.m*: Computation of the exact solution to the perfectly conducting cavity with a material interface at  $x = 0$ .**

---

```

function [Ef Hf ep mu] = CavityExact(x, epl, epr, mul, mur, time);
% function [Ef Hf ep mu] = CavityExact(x, epl, epr, mul, mur, time);
% Purpose: Set up exact solution to EM cavity problem
xL = length(x); ii = sqrt(-1.0); n1 = sqrt(epl*mul); n2 = sqrt(epr*mur);
Ef = zeros(xL,1); Hf = zeros(xL,1); ep = zeros(xL,1); mu = zeros(xL,1);

% Compute omega to match coefficients - set initial guess to obtain
% different solutions
omega = fzero(@(x) (n1*tan(n2*x) + n2*tan(n1*x)), 5);

% Set up exact solution
A1 = complex(n2*cos(omega*n2)/(n1*cos(n1*omega)));
A2 = exp(-ii*omega*(n1+n2));
B1 = A1*exp(-ii*2*n1*omega); B2 = A2*exp(ii*2*n2*omega);
for j=1:xL
    if (x(j)<= 0)
        A = A1; B = B1; n = n1; ep(j)=epl; mu(j)=mul;
    else
        A = A2; B = B2; n = n2; ep(j)=epr; mu(j)=mur;
    end
    Eh = (A*exp(ii*n*omega*x(j)) - B*exp(-ii*n*omega*x(j))) ...
        *exp(-ii*omega*time);
    Hh = n*(A*exp(ii*n*omega*x(j)) + B*exp(-ii*n*omega*x(j))) ...
        *exp(-ii*omega*time);
    Ef(j) = real(Eh); Hf(j) = real(Hh);
end;
return

```

---

The temporal integration of the problem is accomplished in **MaxwellM1D.m**, which relies on **MaxwellMrhs1D.m** to compute the numerical flux. As for Burgers' equation, this routine is the only routine that contains essential information about the problem, expressed through the associated flux function.

---

**Script 5.15. *MaxwellM1D.m*: Temporal integration routine for solving Maxwell's equation using a conservative finite difference method.**

---

```

function [EM] = MaxwellM1D(x, EM, ep, mu, h, CFL, FinalTime)
% function [EM] = MaxwellM1D(x, EM, ep, mu, CFL, FinalTime)
% Purpose : Integrate 1D Maxwell's equations until FinalTime using
% a monotone scheme.
time = 0; tstep = 0;

% Set timestep
cvel = 1./sqrt(ep.*mu); k = CFL*h/max(cvel);

```

---

```
% integrate scheme
while (time<FinalTime)
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    EM = EM + k*MaxwellMrhs1D(x,EM,ep, mu,h,k,max(cvel));
    time = time+k; tstep = tstep+1;
end
return
```

---

**Script 5.16. MaxwellMrhs1D.m: Definition of the right-hand side for Maxwell's equations.**

---

```
function [dEH] = MaxwellMrhs1D(x,EM,ep, mu,h,k,maxvel)
% function [dEH] = MaxwellMrhs1D(x,EM,ep, mu,h,k,maxvel);
% Purpose: Evaluate right hand side for Maxwell's equations
% using a monotone method
N = length(x);

% PEC boundary conditions by mirror principle
[xe,Ee] = extend(x,EM(:,1),h,1,'D',0,'D',0);
[xe,He] = extend(x,EM(:,2),h,1,'N',0,'N',0);
EH = [Ee(2:N+1) He(2:N+1)];
EHp = [Ee(3:N+2) He(3:N+2)]; EHm = [Ee(1:N) He(1:N)];

% Change numerical flux here
dEH = - (MaxwellLF(EM,EHp,ep, mu,k/h,maxvel) - ...
          MaxwellLF(EHm,EM,ep, mu,k/h,maxvel))/h;
return
```

---

As boundary conditions we use homogeneous Dirichlet conditions on  $E$  and homogeneous Neumann conditions on  $H$ . The global Lax-Friedrichs flux for Maxwell's equations is given in MaxwellLF.m and the Lax-Wendroff flux for Maxwell's equations is illustrated in MaxwellLW.m.

**Script 5.17. MaxwellLF.m: Definition of the global Lax-Friedrichs numerical flux,  $F(u, v)$ , for Maxwell's equation.**

---

```
function [numflux] = MaxwellLF(u, v, ep, mu, lambda, maxvel)
% function [numflux] = MaxwellLF(u, v, ep, mu, lambda, maxvel);
% Purpose: Evaluate Lax Friedrich numerical flux
% for Maxwell's equations

fu = [u(:,2)./ep u(:,1)./mu]; fv = [v(:,2)./ep v(:,1)./mu];
numflux = (fu+fv)/2 - maxvel/2*(v-u);
return
```

---

**Script 5.18. MaxwellLW.m: Definition of the Lax-Wendroff numerical flux,  $F(u, v)$ , for Maxwell's equation.**

---

```
function [numflux] = MaxwellLW(u, v, ep, mu, lambda, maxvel)
% function [numflux] = MaxwellLW(u, v, ep, mu, lambda, maxvel);
% Purpose: Evaluate Lax Wendroff numerical flux for Maxwell's equations

fu = [u(:,2)./ep u(:,1)./mu]; fv = [v(:,2)./ep v(:,1)./mu];
A2 = [ 1./ep.*mu 1./ep.*mu ];
numflux = (fu+fv)/2 - lambda/2*A2.*(v-u);
return
```

---

**Table 5.2.** Errors for Maxwell's equations at  $T = \pi$  with a smooth solution, solved using a conservative finite difference method with two different choices of the numerical flux. The error is measured in  $\|\cdot\|_{h,1}$ .

N	Lax-Friedrichs		Lax-Wendroff	
	E	H	E	H
128	6.0e-2	9.5e-2	5.8e-3	4.0e-3
512	1.4e-2	2.4e-2	3.7e-4	2.3e-4
2048	3.6e-3	6.0e-3	2.3e-5	1.4e-5
8192	8.9e-4	1.5e-3	1.4e-6	8.7e-7
32768	2.2e-4	3.7e-4	9.0e-8	5.4e-8

**Table 5.3.** Errors for Maxwell's equations at  $T = \pi$  with a nonsmooth solution, solved using a conservative finite difference method with two different choices of the numerical flux. The error is measured in  $\|\cdot\|_{h,1}$ .

N	Lax-Friedrichs		Lax-Wendroff	
	E	H	E	H
128	3.4e-1	6.4e-1	5.7e-2	6.2e-2
512	9.4e-2	1.9e-1	1.6e-2	1.5e-2
2048	2.4e-2	4.8e-2	4.3e-3	3.8e-3
8192	6.0e-3	1.2e-2	1.1e-3	1.0e-3
32768	1.5e-3	3.0e-3	2.7e-4	2.4e-4

To evaluate the quality of the solvers, we consider two test cases. We first assume that all material parameters are one, yielding a sinusoidal wave that oscillates in the cavity.

The results are shown in Table 5.2, where we list the errors, measured in the discrete  $L^1$ -norm, for two different choices of the numerical flux. As is expected for this smooth linear case, the schemes achieve design accuracy, i.e.,  $\mathcal{O}(h)$  for the Lax-Friedrichs scheme and  $\mathcal{O}(h^2)$  for the Lax-Wendroff scheme.

While the results are encouraging and confirm the analysis, they also show that if one can evaluate the numerical fluxes with adequate accuracy, higher than first order is indeed possible, i.e., even if the conservation form looks like a simple first order scheme, it is not. This observation plays a central role in the developments of high-order schemes in Part III of this book.

Let us also consider a case where the fields are continuous but not smooth. We achieve this by choosing  $\varepsilon_2 = 2.25$  while all other material coefficients are held at unity. This corresponds to a cavity half filled with glass, as illustrated in Fig. 1.6. The results are shown in Table 5.3, listing the errors, measured in the discrete  $L^1$ -norm, for the two different choices of the numerical flux. As expected, the lack of smoothness has an adverse effect on the accuracy, which is  $\mathcal{O}(h)$  for both fluxes. It is worth noticing, however, that in spite of the loss of the formal order of accuracy for the Lax-Wendroff scheme, the accuracy is substantially better than what is obtained with the formally first order Lax-Friedrichs scheme.

### 5.3.5 • Euler equations

Let us finally evaluate the performance of the different schemes for the more challenging case of the Euler equations of compressible gas dynamics. The details of the problem are provided in subsection 1.4.1.

In EulerMDriver1D.m the computational domain, the resolution, the CFL condition, and the initial conditions are specified, calling EulerM1D.m which controls the time step through the maximum velocity,  $c + |u|$ , as derived in Ex. 3.3.

**Script 5.19. EulerMDriver1D.m: Driver routine for solving the Euler equations using a conservative finite difference method.**

---

```
% Driver script for solving the 1D Euler equations using a monotone scheme
clear all

% Set problem parameters
L = 1; FinalTime = 0.2; N = 2048; h = L/N; CFL = 0.90; gamma = 1.4;

% Define domain, materials and initial conditions
r = zeros(N+1,1); ru = zeros(N+1,1); E = zeros(N+1,1);

% Initialize for Sod's problem
x = [0:h:1]';
for i=1:N+1;
    if x(i)<0.5
        r(i)=1.0; E(i) = 1/(gamma-1);
    else
        r(i) = 0.125; E(i) = 0.1/(gamma-1);
    end
end

% Initialize for shock entropy problem
% x = [-5:h:5]';
% for i=1:N+1;
%     if x(i)<-4
%         rb = 3.857143; u = 2.629369; p = 10.33333;
%     else
%         rb = 1+0.2*sin(pi*x(i)); u = 0; p = 1;
%     end
%     r(i) = rb; ru(i) = rb*u; E(i) = p/(gamma-1)+0.5*rb*u^2;
% end

% Solve Problem
q = [r ru E];
[q] = EulerM1D(x,q,h,CFL,gamma,FinalTime);
```

---

**Script 5.20. EulerM1D.m: Temporal integration routine for solving the Euler equations using a conservative finite difference method.**

---

```
function [q] = EulerM1D(x,q,h,CFL,gamma,FinalTime)
% function [q] = EulerMDriver1D(x,q,CFL,gamma,FinalTime)
% Purpose: Integrate the 1D Euler equations until FinalTime
% using a monotone scheme.
time = 0; tstep = 0;

% integrate scheme
while (time<FinalTime)
    % Set timestep
    p = (gamma-1)*(q(:,3) - 0.5*q(:,2).^2./q(:,1)); c = sqrt(gamma*p./q(:,1));
    maxvel = max(c+abs(q(:,2)./q(:,1))); k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    q = q + k*EulerMrhs1D(x,q,gamma,h,k,maxvel);
    time = time+k; tstep = tstep+1;
end
return
```

---

The conservation form is expressed in `EulerMrhs1D.m`, where we also impose boundary conditions on the conserved variables. In this case we simply impose the assumption that the solution remains constant outside the domain of interest.

**Script 5.21.** *EulerMrhs1D.m: Definition of the right-hand side for the Euler equations.*

---

```
function [dq] = EulerMrhs1D(x,q, gamma, h, k, maxvel)
% function [dq] = EulerMrhs1D(x,q, gamma, h, k, maxvel);
% Purpose: Evaluate right hand side for the Euler equations
% using a monotone method

% Extend data and assign boundary conditions
[ xe , re ] = extend(x,q(:,1),h,1,'D',1.0,'D',0.125);
[ xe , me ] = extend(x,q(:,2),h,1,'D',0,'N',0);
[ xe , Ee ] = extend(x,q(:,3),h,1,'D',2.5,'N',0);

% [xe , re] = extend(x,q(:,1),h,1,'D',3.857143,'N',0);
% [xe , me] = extend(x,q(:,2),h,1,'D',10.141852,'D',0);
% [xe , Ee] = extend(x,q(:,3),h,1,'D',39.166661,'N',0);

% Compute RHS - change numerical flux here
N = length(x); q = [ re(2:N+1) me(2:N+1) Ee(2:N+1) ];
qp = [ re(3:N+2) me(3:N+2) Ee(3:N+2) ]; qm = [ re(1:N) me(1:N) Ee(1:N) ];
dq = - (EulerLF(q,qp, gamma, k/h, maxvel) - EulerLF(qm,q, gamma, k/h, maxvel))/h;
return
```

---

The definition of the numerical flux is the key element of the algorithm and we shall discuss three different fluxes: the global Lax-Friedrichs (`EulerLF.m`), the local Lax-Friedrichs (`EulerLLF.m`), and the Lax-Wendroff numerical flux (`EulerLW.m`). For the latter flux, we need the flux Jacobian, derived in Ex. 3.3.

**Script 5.22.** *EulerLF.m: Definition of the Lax-Friedrichs numerical flux,  $F(u, v)$  for the Euler equations.*

---

```
function [numflux] = EulerLF(u,v, gamma, lambda, maxvel)
% function [numflux] = EulerLF(u,v, gamma, lambda, maxvel);
% Purpose: Evaluate global Lax Friedrich numerical flux for
% the Euler equations

% Compute flux for u
r = u(:,1); ru = u(:,2); E = u(:,3); pu = (gamma-1)*(E - 0.5*ru.^2./r);
fu = [ ru (ru.^2./r+pu) (E+pu).*ru./r ];

% Compute flux for v
r = v(:,1); ru = v(:,2); E = v(:,3); pv = (gamma-1)*(E - 0.5*ru.^2./r);
fv = [ ru (ru.^2./r+pv) (E+pv).*ru./r ];

% Evaluate numerical flux
numflux = (fu+fv)/2 - maxvel/2*(v-u);
return
```

---

**Script 5.23.** *EulerLLF.m: Definition of the local Lax-Friedrichs numerical flux,  $F(u, v)$ , for the Euler equations.*

---

```
function [numflux] = EulerLLF(u,v, gamma, lambda, maxvel)
% function [numflux] = Eulerflux(u,v, gamma, lambda, maxvel);
% Purpose: Evaluate the local Lax Friedrich numerical flux for
% the Euler equations
```

```
% Compute flux for u
r = u(:,1); ru = u(:,2); E = u(:,3);
pu = (gamma-1)*(E - 0.5*ru.^2./r); cu = sqrt(gamma*pu./u(:,1));
fu = [ru (ru.^2./r+pu) (E+pu).*ru./r];

% Compute flux for v
r = v(:,1); ru = v(:,2); E = v(:,3);
pv = (gamma-1)*(E - 0.5*ru.^2./r); cv = sqrt(gamma*pv./v(:,1));
fv = [ru (ru.^2./r+pv) (E+pv).*ru./r];

% Evaluate numerical flux
localvel = max((cu+abs(u(:,2)./u(:,1))), (cv+abs(v(:,2)./v(:,1)))); 
numflux = (fu+fv)/2 - bsxfun(@times, v-u, localvel)/2;
return
```

---

**Script 5.24.** *EulerLW.m: Definition of the Lax–Wendroff numerical flux,  $F(u, v)$ , for the Euler equations.*

---

```
function [ numflux ] = EulerLW(u,v,gamma,lambda,maxvel)
% function [numflux] = EulerLW(u,v,gamma,lambda,maxvel);
% Purpose: Evaluate Lax–Wendroff numerical flux for the Euler equations

% Compute flux for u
r = u(:,1); ru = u(:,2); E = u(:,3); pu = (gamma-1)*(E - 0.5*ru.^2./r);
fu = [ru (ru.^2./r+pu) (E+pu).*ru./r];

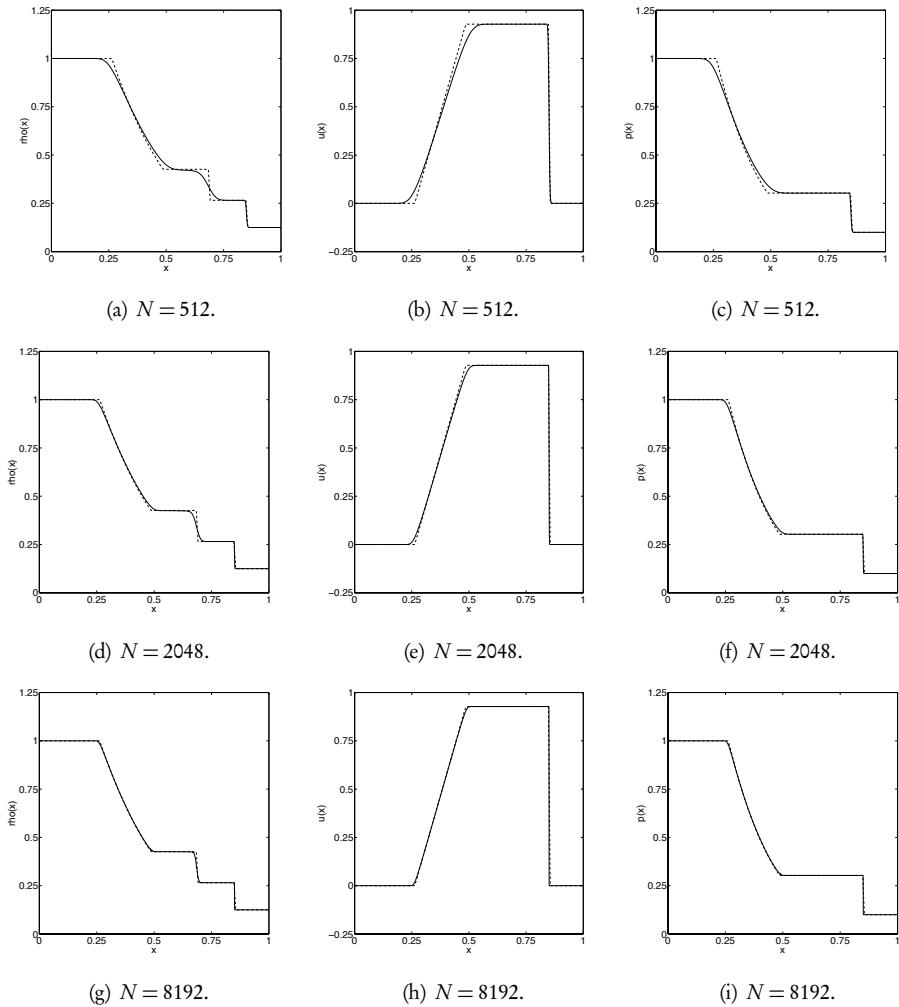
% Compute flux for v
r = v(:,1); ru = v(:,2); E = v(:,3); pv = (gamma-1)*(E - 0.5*ru.^2./r);
fv = [ru (ru.^2./r+pv) (E+pv).*ru./r];

% Evaluate numerical flux
fw = fv-fu; w = (u+v)/2; rw = w(:,1); ruw = w(:,2);
Ew = w(:,3); uw = ruw./rw; wL = length(rw);
A21 = -(3-gamma)/2*uw.^2; A22 = (3-gamma)*uw; A23 = (gamma-1)*ones(wL,1);
A31 = -gamma*Ew.*uw./rw + (gamma-1)*uw.^3;
A32 = gamma*Ew./rw - 3*(gamma-1)/2*uw.^2; A33 = gamma*uw;
LFvec = zeros(wL,3); LFvec(:,1) = fw(:,2);
LFvec(:,2) = A21.*fw(:,1)+A22.*fw(:,2)+A23.*fw(:,3);
LFvec(:,3) = A31.*fw(:,1)+A32.*fw(:,2)+A33.*fw(:,3);
numflux = ((fu+fv) - lambda*LFvec)/2;
return
```

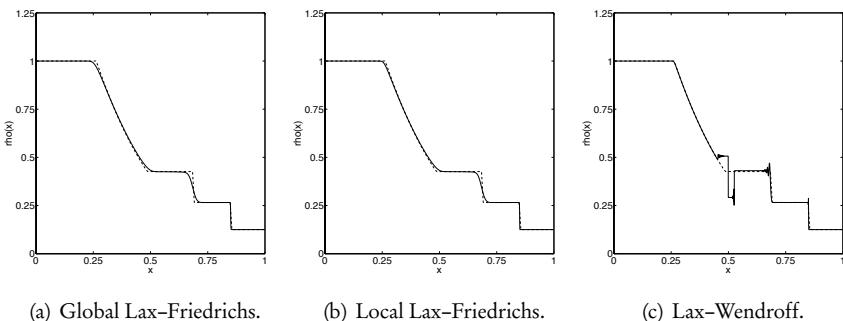
---

To appreciate the performance of the different schemes, we first consider Sod’s problem and show in Fig. 5.5 the solution obtained using the Lax–Friedrichs flux. We observe, as expected, convergence of the results as  $N$  increases and find no oscillations near the shocks. We also observe considerably more smearing of the contact discontinuity ( $x = 0.685$ ) than of the shock ( $x = 0.850$ ). This is caused by the compressive nature of the shock where the characteristics run into the shock to help keep the shock sharp. The contact discontinuity essentially behaves as a linear wave where the effect of the dissipation accumulates.

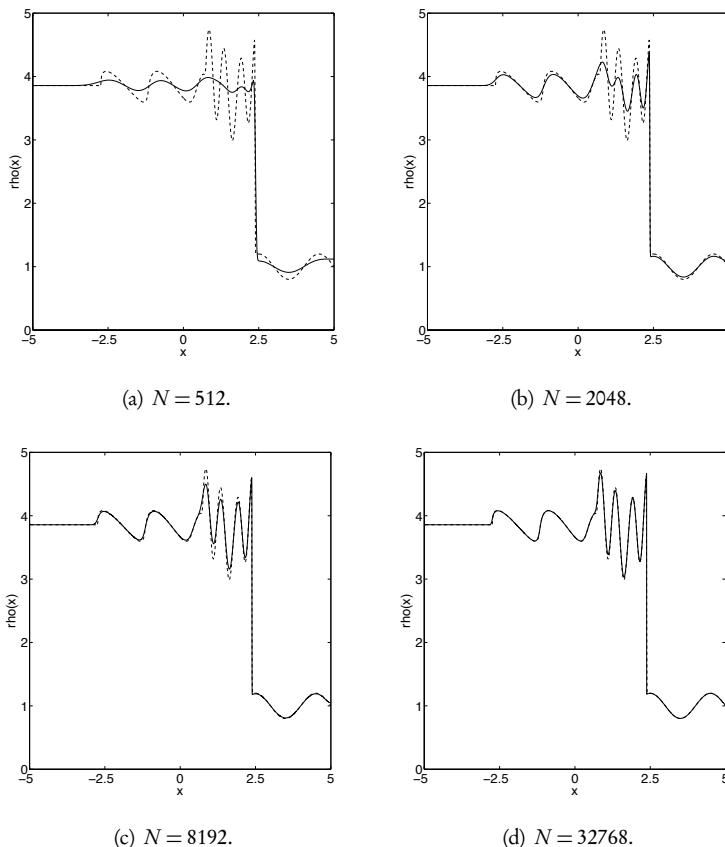
To evaluate the effect of changing the flux, we compare in Fig. 5.6 the density  $\rho$  computed at  $T = 0.2$  with  $N = 2048$ , using three different numerical fluxes. While the differences between the two results, obtained using the global and the local Lax–Friedrichs fluxes, are minimal, the solution obtained with the nonmonotone Lax–Wendroff scheme is very poor. Although there are clear improvements in the location



**Figure 5.5.** Computed solution for Sod's problem at  $T = 0.2$  using the Lax-Friedrichs scheme. Columns are the density  $\rho$ , the velocity  $u$ , and the pressure  $p$ . The dashed lines indicate the exact solution.



**Figure 5.6.** Computed density  $\rho$  for Sod's problem at  $T = 0.2$  with  $N = 2048$  using three different numerical fluxes. The dashed line represents the exact solution.

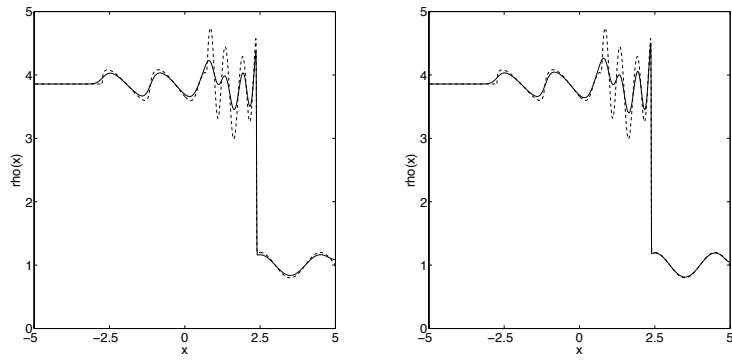


**Figure 5.7.** Computed density  $\rho$  for the shock entropy wave problem for the Euler equations at  $T = 1.8$ , computed using the global Lax–Friedrichs method. The dashed lines represent a high accuracy reference solution and  $N$  is the number of grid points used in each case.

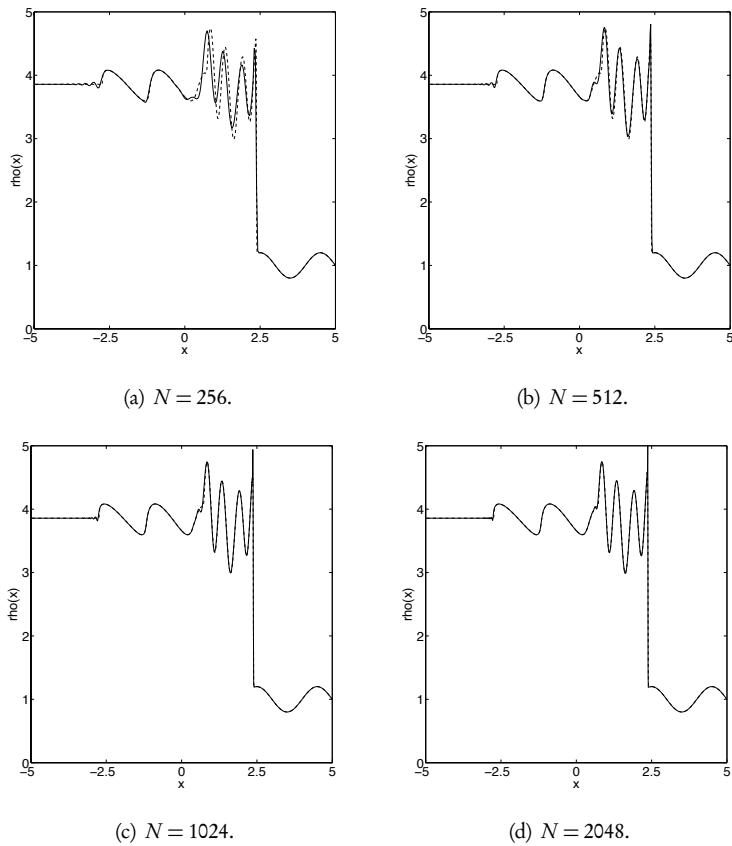
of the shock and the contact discontinuity, the substantial oscillation introduced around the rarefaction wave is clearly problematic and illustrates the importance of monotonicity.

To consider a problem with more structure, we show in Fig. 5.7 the solution to the shock entropy problem, obtained using the global Lax–Friedrichs scheme. While convergence with increasing  $N$  is clear, the impact of the dissipation of the Lax–Friedrichs flux is substantial. Changing the flux to a local Lax–Friedrichs flux yields some improvements in the resolution of the shock and the trailing waves, as shown in Fig. 5.8.

For Sod’s problem we experienced problems with the Lax–Wendroff numerical flux, resulting in substantial oscillations. In Fig. 5.9 we show the results for the shock entropy problem, computed using the Lax–Wendroff flux. When compared to the results in Fig. 5.7 the improvement in accuracy is dramatic, both with respect to the location of the shock and the resolution of the wave structures upstream of the shock. While there are minor oscillations, the potential benefits of a higher-order method are very significant when judged by this example.



**Figure 5.8.** Computed density  $\rho$  for the entropy wave problem for the Euler equations at  $T = 1.8$ , computed using a global Lax-Friedrichs method (left) and a local Lax-Friedrichs method. In both cases, we use 2048 grid points and the dashed line represents a high accuracy reference solution.



**Figure 5.9.** Computed density  $\rho$  for the shock entropy problem for the Euler equations at  $T = 1.8$ , obtained using a Lax-Wendroff method. The dashed lines represent a high accuracy reference solution and  $N$  is the number of grid points used in each case.

## References

- [1] Richard Courant, Kurt Friedrichs, and Hans Lewy. Über die partiellen differenzen-gleichungen der mathematischen physik. *Mathematische Annalen*, 100(1):32–74, 1928.
- [2] Michael G. Crandall and Andrew Majda. Monotone difference approximations for scalar conservation laws. *Mathematics of Computation*, 34(149):1–21, 1980.
- [3] Lawrence C. Evans. *Partial Differential Equations*, 2nd ed., American Mathematical Society, 1998.
- [4] Jonathan Goodman and Zhouping Xin. Viscous limits for piecewise smooth solutions to systems of conservation laws. *Archive for Rational Mechanics and Analysis*, 121(3):235–265, 1992.
- [5] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. *Time Dependent Problems and Difference Methods*, volume 24 of Pure and Applied Mathematics. John Wiley & Sons, 1995.
- [6] Eduard Harabetian. Rarefactions and large time behavior for parabolic equations and monotone schemes. *Communications in Mathematical Physics*, 114(4):527–536, 1988.
- [7] N. N. Kuznetsov. Accuracy of some approximate methods for computing the weak solutions of a first-order quasi-linear equation. *USSR Computational Mathematics and Mathematical Physics*, 16(6):105–119, 1976.
- [8] Peter D. Lax and Robert D. Richtmyer. Survey of the stability of linear finite difference equations. *Communications on Pure and Applied Mathematics*, 9(2):267–293, 1956.
- [9] Randall J. LeVeque. *Numerical Methods for Conservation Laws*, Lectures in Mathematics, ETH Zürich. Birkhäuser-Verlag, 1992.
- [10] Randall J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*, volume 98 of Other Titles in Applied Mathematics. SIAM, 2007.
- [11] Keith W. Morton and David Francis Mayers. *Numerical Solution of Partial Differential Equations: An Introduction*. Cambridge University Press, 2005.
- [12] Robert D. Richtmyer and Keith W. Morton. *Difference Methods for Initial-Value Problems*, 2nd ed., Krieger, 1994.
- [13] Florin Šabac. The optimal convergence rate of monotone finite difference methods for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 34(6):2306–2318, 1997.
- [14] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.
- [15] John C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*, 2nd ed., volume 88 of Other Titles in Applied Mathematics, SIAM, 2004.

- 
- [16] John C. Strikwerda and Bruce A. Wade. A survey of the Kreiss matrix theorem for power bounded families of matrices and its extensions. *Banach Center Publications*, 38:339–360, 1997.
  - [17] Eitan Tadmor. Local error estimates for discontinuous solutions of nonlinear hyperbolic equations. *SIAM Journal on Numerical Analysis*, 28(4):891–906, 1991.
  - [18] Eitan Tadmor and Tao Tang. Pointwise error estimates for scalar conservation laws with piecewise smooth solutions. *SIAM Journal on Numerical Analysis*, 36(6):1739–1758, 1999.
  - [19] Tao Tang and Zhen Huan Teng. The sharpness of Kuznetsov’s  $\mathcal{O}(\sqrt{\Delta h})$   $L^1$ -error estimate for monotone difference schemes. *Mathematics of Computation*, 64(210):581–589, 1995.
  - [20] Tao Tang and Zhen-Huan Teng. Viscosity methods for piecewise smooth solutions to scalar conservation laws. *Mathematics of Computation*, 66(218):495–526, 1997.
  - [21] Wikipedia contributors. *Euler equations (fluid dynamics)*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Euler\\_equations\\_\(fluid\\_dynamics\)](https://en.wikipedia.org/wiki/Euler_equations_(fluid_dynamics)) (accessed August 2016).
  - [22] Wikipedia contributors. *John von Neumann*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/John\\_von\\_Neumann](https://en.wikipedia.org/wiki/John_von_Neumann) (accessed August 2016).

# Chapter 6

# Finite volume methods

The development of finite difference methods for conservation laws is based on the differential form of the conservation law. As we experienced in Chapter 5, this leads to simple and effective schemes. In this chapter we pursue a different approach and develop methods based on the integral form. While one could expect that this difference in origin would result in very different schemes, we will soon realize that they are similar and, in many cases, equivalent to the ones we have already discussed at length. However, the difference in derivation allows us to gain additional insight that will prove essential once we begin the discussion of higher-order accurate schemes.

## 6.1 • Godunov's method

Let us again introduce the grid  $(x_j, t^n) = (jh, kn)$ , where  $h$  and  $k$  represent the spatial and temporal grid size, respectively, and consider the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to appropriate boundary and initial conditions. Let us also define a space-time cell as  $[x_{j-1/2}, x_{j+1/2}] \times [t^n, t^{n+1}]$  with  $x_{j\pm 1/2} = x_j \pm h/2$  marking the cell boundaries. The integral form of the conservation law defined on the cell centered at  $x_j$  is

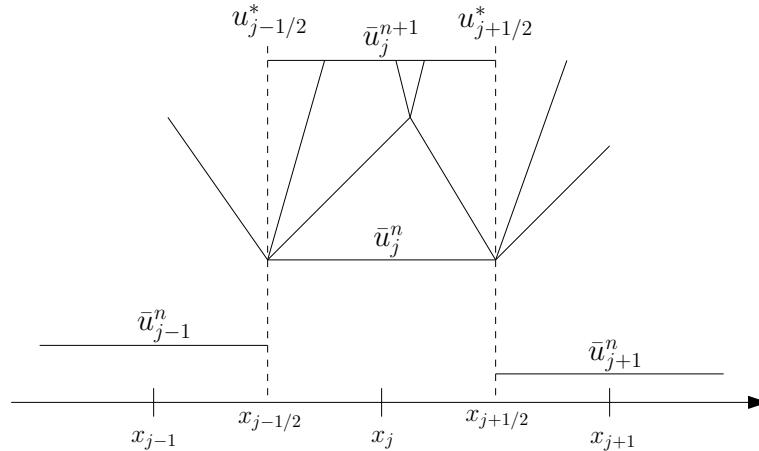
$$\int_{x_{j-1/2}}^{x_{j+1/2}} [u(x, t^{n+1}) - u(x, t^n)] dx = - \int_{t^n}^{t^{n+1}} [f(u(x_{j+1/2}, t)) - f(u(x_{j-1/2}, t))] dt.$$

Let us now introduce the cell average

$$\bar{u}_j^n = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t^n) dx,$$

and the flux

$$F_{j+1/2}^n = \frac{1}{k} \int_{t^n}^{t^{n+1}} f(u(x_{j+1/2}, t)) dt, \quad (6.1)$$



**Figure 6.1.** Sketch of the construction of the solution for the Godunov scheme, highlighting the characteristics emanating from the cell interfaces.

to recover the scheme

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n],$$

known as a finite volume method. The first thing to note is that the scheme is in conservation form by construction, hence ensuring correct shock speeds. Furthermore, we observe that the unknown is the cell average rather than the point value as in the finite difference scheme. However, to turn this into an actual scheme, we need to evaluate  $F_{j\pm 1/2}^n$ , defined in (6.1).

Guided by Fig. 6.1, let us first assume that  $u_{j+1/2}^* = u(x_{j+1/2}, t^n)$  can be reconstructed from  $(\bar{u}_{j-p}^n, \dots, \bar{u}_{j+q}^n)$  by a polynomial,  $p(x)$ , such that  $u_{j+1/2}^* = p(x_{j+1/2})$ . Here  $p(x)$  must be constructed to ensure conservation

$$\forall l: j-p \leq l \leq j+q, \int_{x_{l-1/2}}^{x_{l+1/2}} p(x) dx = \bar{u}_l^n.$$

The simplest example is  $u_{j+1/2}^* = \frac{1}{2}(\bar{u}_j^n + \bar{u}_{j+1}^n)$ . In Part III we shall discuss more accurate alternatives to this reconstruction.

The time-evolution of  $u_{j+1/2}^*$  is obtained by solving the Riemann problem, as discussed in detail in section 3.2. Since the solution is constant between waves we recover  $F_{j+1/2}^n = h f(u_{j+1/2}^*)$ , i.e., once  $u_{j+1/2}^*$  is determined it will remain constant at the cell interface over the time step  $k$ , provided  $k$  is sufficiently small. This was also illustrated in Ex. 3.4.

The upper bound on  $k$  must ensure that  $u_{j+1/2}^*$  remains constant. Thus, we require that the fastest wave is unable to reach a neighborhood interface  $x_{j+1/2\pm 1}$  within one time step. If the maximum wave speed is  $|\alpha|$ , this implies that  $h \geq |\alpha|k$ , or  $k/(|\alpha|h) \leq 1$ , which we recognize as the CFL condition, discussed in section 5.1.

The only remaining task is the determination of  $u_{j+1/2}^*$  for a general conservation law. With our understanding of the Riemann problem, this amounts to finding the

intermediate state of the Riemann problem between the two states  $u_l$  and  $u_r$  at  $x_{j+1/2}$ , i.e.,

$$u_l = \lim_{\varepsilon \rightarrow 0^+} p_l(x_{j+1/2} - \varepsilon), \quad u_r = \lim_{\varepsilon \rightarrow 0^+} p_r(x_{j+1/2} + \varepsilon),$$

where  $p_{l,r}$  reflects that the reconstruction may be different for the left and right side of the limit.

If we consider Burgers' equation, an intuitive scheme is

$$u_{j+1/2}^* = \begin{cases} u_l, & s \geq 0, \\ u_r, & s < 0, \end{cases} \quad s = \frac{f(u_l) - f(u_r)}{u_l - u_r}. \quad (6.2)$$

Looking back to Chapter 4 and Ex. 4.4, we recognize this scheme as being closely related to the Roe flux. We express the scheme as

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{h} [f(u_{j+1/2}^*) - f(u_{j-1/2}^*)].$$

If we assume that the reconstruction  $p(x)$  depends only on cells neighboring  $x_{j+1/2}$  and set  $f(u_{j+1/2}^*) = F(\bar{u}_j^n, \bar{u}_{j+1}^n)$ , it is easy to see that the finite difference scheme with the Roe flux and the finite volume method are identical, even though their origins are different.

As we discussed in Ex. 5.9, the simple scheme in (6.2) does not guarantee that the correct entropy solution is computed in the presence of a sonic point. One approach, leading to the celebrated Godunov's method [6], is to require that the local Riemann problem be solved exactly. This is the first example of the family of schemes known as finite volume methods.

**Example 6.1.** Let us consider Godunov's method:

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{h} [f(u_{j+1/2}^*) - f(u_{j-1/2}^*)],$$

where  $u^*$  is computed by solving the Riemann problem exactly and selecting the correct entropy solution. If we assume the existence of a convex entropy pair,  $(\eta, \psi)$ , any such piecewise constant solution  $u^e$  will satisfy the cell entropy inequality

$$\begin{aligned} \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u^e, t^{n+1}) dx &\leq \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u^e, t^n) dx \\ &\quad - \int_{t^n}^{t^{n+1}} [\psi(u^e(x_{j+1/2}, t)) - \psi(u^e(x_{j-1/2}, t))] dt. \end{aligned}$$

Since  $u^e$  is exact at  $x_{j\pm 1/2}$  along  $[t^n, t^{n+1}]$  we have that

$$\frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u^e, t^{n+1}) dx \leq \eta(\bar{u}_j^n) - \frac{k}{h} [\psi(u_{j+1/2}^*) - \psi(u_{j-1/2}^*)].$$

However, since  $\eta(u)$  is convex, we have

$$b\eta(\bar{u}_j^{n+1}) = \eta \left( \int_{x_{j-1/2}}^{x_{j+1/2}} u^e(x, t^{n+1}) dx \right) \leq \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u^e, t^{n+1}) dx$$

by Jensen's inequality, and we recover the cell entropy condition

$$\eta(\bar{u}_j^{n+1}) \leq \eta(\bar{u}_j^n) - \frac{k}{h} [\psi(u_{j+1/2}^*) - \psi(u_{j-1/2}^*)].$$

This guarantees that the scheme recovers the entropy solution.  $\blacksquare$

Following [4], we can express the Godunov flux as

$$F(u, v) = \begin{cases} f(u), & u \geq v, f(u) \geq f(v), \\ f(v), & u \geq v, f(u) < f(v), \\ f(u), & u < v, f'(u) \geq 0, \\ f(v), & u < v, f'(v) < 0, \\ f(f^{-1}(0)), & \text{otherwise.} \end{cases}$$

For the scalar case, one quickly realizes that this is simply the Roe flux with the sonic fix, as discussed in Ex. 5.9. Inspection reveals that  $F(\uparrow, \downarrow)$ , confirming that the Godunov flux is monotone.

The requirement of using an exact entropy satisfying solution of the Riemann problem is a bottleneck for the use of Godunov's methods for more complex problems. However, if we restrict ourselves to linear problems, the Riemann problem is solvable as the discussion in section 3.2 showed.

To see how to approach this, consider again (3.5):

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{A} \frac{\partial \mathbf{u}}{\partial x} = 0,$$

where  $\mathbf{u} : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$ . We recall that

$$\mathbf{u}^* = \bar{\mathbf{u}}_j^n + \sum_{\lambda_i < 0} (\beta_i - \alpha_i) \mathbf{s}_i = \bar{\mathbf{u}}_{j+1}^n - \sum_{\lambda_i > 0} (\beta_i - \alpha_i) \mathbf{s}_i,$$

where

$$\bar{\mathbf{u}}_j^n = \sum_{i=1}^m \alpha_i \mathbf{s}_i, \quad \bar{\mathbf{u}}_{j+1}^n = \sum_{i=1}^m \beta_i \mathbf{s}_i,$$

and  $(\lambda_i, \mathbf{s}_i)$  are the eigenpairs of  $\mathbf{A}$ . For simplicity, we assume that the problem is strictly hyperbolic. We consider the numerical flux

$$\begin{aligned} f(\mathbf{u}^*) &= \mathbf{A} \mathbf{u}^* = \mathbf{A} \bar{\mathbf{u}}_j^n + \sum_{\lambda_i < 0} (\beta_i - \alpha_i) \lambda_i \mathbf{s}_i = \mathbf{A} \bar{\mathbf{u}}_j^n + \mathbf{A}^- (\bar{\mathbf{u}}_{j+1}^n - \bar{\mathbf{u}}_j^n) \\ &= \mathbf{A} \bar{\mathbf{u}}_{j+1}^n - \sum_{\lambda_i > 0} (\beta_i - \alpha_i) \lambda_i \mathbf{s}_i = \mathbf{A} \bar{\mathbf{u}}_{j+1}^n - \mathbf{A}^+ (\bar{\mathbf{u}}_{j+1}^n - \bar{\mathbf{u}}_j^n), \end{aligned}$$

where  $\mathbf{A} = \mathbf{S}^{-1} \Lambda \mathbf{S} = \mathbf{S}^{-1} (\Lambda^+ + \Lambda^-) \mathbf{S} = \mathbf{A}^+ + \mathbf{A}^-$ , and  $\Lambda_{ii}^+ \geq 0$ ,  $\Lambda_{ii}^- < 0$ . If we now associate

$$f(u_{j+1/2}^*) = F(\bar{\mathbf{u}}_j^n, \bar{\mathbf{u}}_{j+1}^n) = \mathbf{A} \bar{\mathbf{u}}_j^n + \mathbf{A}^- (\bar{\mathbf{u}}_{j+1}^n - \bar{\mathbf{u}}_j^n) = \mathbf{A} \bar{\mathbf{u}}_j^n + \mathbf{A}^- \Delta^+ \bar{\mathbf{u}}_j^n$$

and

$$f(u_{j-1/2}^*) = F(\bar{\mathbf{u}}_{j-1}^n, \bar{\mathbf{u}}_j^n) = \mathbf{A} \bar{\mathbf{u}}_j^n - \mathbf{A}^+ (\bar{\mathbf{u}}_j^n - \bar{\mathbf{u}}_{j-1}^n) = \mathbf{A} \bar{\mathbf{u}}_j^n - \mathbf{A}^+ \Delta^- \bar{\mathbf{u}}_j^n,$$

we recover the exact scheme

$$\begin{aligned}\bar{\mathbf{u}}_j^{n+1} &= \bar{\mathbf{u}}_j^n - \frac{k}{h} \left[ A^- (\bar{\mathbf{u}}_{j+1}^n - \bar{\mathbf{u}}_j^n) + A^+ (\bar{\mathbf{u}}_j^n - \bar{\mathbf{u}}_{j-1}^n) \right] \\ &= (I - \lambda (A^- \Delta^+ + A^+ \Delta^-)) \bar{\mathbf{u}}_j^n.\end{aligned}$$

A quick inspection reveals that this is nothing else but upwinding on the waves, depending on whether they are propagating left or right.

If we instead add the two expressions for  $A\mathbf{u}^*$ , we recover the numerical flux

$$\begin{aligned}f(\mathbf{u}^*) &= A\mathbf{u}^* = \frac{1}{2} (A\bar{\mathbf{u}}_j^n + A\bar{\mathbf{u}}_{j+1}^n) - \frac{1}{2} |A| (\bar{\mathbf{u}}_{j+1}^n - \bar{\mathbf{u}}_j^n) \\ &= \frac{1}{2} (A\bar{\mathbf{u}}_j^n + A\bar{\mathbf{u}}_{j+1}^n) - \frac{1}{2} |A| \Delta^+ \bar{\mathbf{u}}_j^n,\end{aligned}$$

where  $|A| = S^{-1}(\Lambda^+ - \Lambda^-)S$ . We recognize this as the Lax–Friedrichs numerical flux, discussed extensively in Chapter 5.

**Example 6.2.** Consider the one-dimensional Maxwell's equations:

$$\begin{bmatrix} \varepsilon(x) & 0 \\ 0 & \mu(x) \end{bmatrix} \frac{\partial}{\partial t} \begin{bmatrix} E \\ H \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \frac{\partial}{\partial x} \begin{bmatrix} E \\ H \end{bmatrix} = 0. \quad (6.3)$$

The details of the problem are discussed in subsection 1.4.1. Following the notation of Ex. 3.5, the flux is given as  $A\mathbf{q}$  and the eigenvalues of  $M^{-1}A$  are  $\pm(\varepsilon\mu)^{-1/2}$ , reflecting two light waves counter-propagating at the local speed of light,  $c = (\varepsilon\mu)^{-1/2}$ .

If we proceed to solve the Riemann problem by using the Rankine–Hugoniot conditions, we obtain

$$\begin{aligned}c_l M_l (\mathbf{q}^* - \mathbf{q}_l) + (A\mathbf{q}^* - A\mathbf{q}_l) &= 0, \\ -c_r M_r (\mathbf{q}^* - \mathbf{q}_r) + (A\mathbf{q}^* - A\mathbf{q}_r) &= 0,\end{aligned}$$

where  $\mathbf{q}^*$  refers to the intermediate state. Simple manipulations yield

$$(c_r M_l + c_l M_r) A\mathbf{q}^* = c_r M_r A\mathbf{q}_l + c_l M_l A\mathbf{q}_r + c_l c_r M_l M_r (\mathbf{q}_l - \mathbf{q}_r),$$

$$H^* = \frac{1}{[Z]} \left( \{ZH\} + \frac{1}{2} [E] \right), \quad E^* = \frac{1}{[Y]} \left( \{YE\} + \frac{1}{2} [H] \right),$$

where

$$Z = \sqrt{\frac{\mu}{\varepsilon}} = (Y)^{-1}$$

represents the impedance of the medium, and  $\{\cdot\}$  is the average and  $[\cdot]$  the jump of the two arguments, respectively.

If we consider the simplest case of a continuous medium, things further simplify to

$$H^* = \{H\} + \frac{Y}{2} [E], \quad E^* = \{E\} + \frac{Z}{2} [H],$$

which is simply the Lax–Friedrichs flux since

$$\frac{Y}{\varepsilon} = \frac{Z}{\mu} = \frac{1}{\sqrt{\varepsilon\mu}} = c$$

is the speed of light, i.e., the fastest wave speed in the system. ■

With the close relationship between the monotone finite difference schemes and the finite volume schemes, it is no surprise that all the key results on accuracy and stability, discussed in some detail in section 5.2 and Theorem 5.10, carry over directly. Hence we can generally only assume first order accuracy, and for discontinuous problems the accuracy may reduce to  $\mathcal{O}(\sqrt{h})$ .

## 6.2 • Approximate Riemann solvers

The key element of the finite volume method is the exact solution of the Riemann problem or, as a minimum, the recovery of an accurate approximation to its solution. While this is relatively straightforward for scalar problems and linear systems, the exact solution for general nonlinear systems is problematic, expensive and, at times, simply impossible. Consequently, a central element in the broader application of the finite volume method is the development of approximate Riemann solvers. In the following we discuss several approaches to this, guided by a wish to balance complexity and computational cost.

### 6.2.1 • Roe fluxes

Consider the conservation law

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u})}{\partial x} = 0,$$

where  $\mathbf{u} : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  and  $\mathbf{f} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ . For the Riemann problem we have the left and right states,  $\mathbf{u}_l$  and  $\mathbf{u}_r$ . We seek an approximate numerical flux via the linearization

$$\mathbf{f}^*(\mathbf{u}^*) = \mathbf{A}^* \mathbf{u}^*, \quad (6.4)$$

where both  $\mathbf{A}^*$  and  $\mathbf{u}^*$  depend on  $\mathbf{u}_l$  and  $\mathbf{u}_r$ . As usual we assume that  $\mathbf{A}^*$  is diagonalizable

$$\mathbf{A}^* \mathbf{s}_i = \lambda_i \mathbf{s}_i,$$

and that  $\lambda_i$  is real. Using the results of section 3.2 for the linear Riemann problem we have

$$\mathbf{f}^* = \mathbf{f}(\mathbf{u}_l) - \sum_{i=1}^m \lambda_i^+ \alpha_i \mathbf{s}_i = \mathbf{f}(\mathbf{u}_r) + \sum_{i=1}^m \lambda_i^- \alpha_i \mathbf{s}_i, \quad (6.5)$$

where

$$\mathbf{u}_l - \mathbf{u}_r = \sum_{i=1}^m \alpha_i \mathbf{s}_i$$

and

$$\lambda_i^+ = \lambda_i \vee 0, \quad \lambda_i^- = \lambda_i \wedge 0.$$

Averaging the two expressions in (6.5) yields the Roe numerical flux

$$f^* = \frac{f(\mathbf{u}_l) + f(\mathbf{u}_r)}{2} - \frac{1}{2} |\mathbf{A}^*| (\mathbf{u}_r - \mathbf{u}_l),$$

where  $|\mathbf{A}^*| = \mathbf{S}^{-1} |\Lambda| \mathbf{S}$ ,  $\Lambda_{ii} = \lambda_i$  is the absolute value of  $\Lambda^*$ .

For the nonlinear case, we must place certain conditions on the definition of  $\mathbf{A}^*$ . It is reasonable to require consistency in the sense that

$$\mathbf{A}^* \rightarrow \nabla_{\mathbf{u}} f \text{ as } (\mathbf{u}_l, \mathbf{u}_r) \rightarrow \mathbf{u}.$$

Furthermore, we require that  $\mathbf{A}^*$  is diagonalizable with real eigenvalues. Let us express the jump in flux as

$$f(\mathbf{u}_r) - f(\mathbf{u}_l) = \int_0^1 \frac{df(\mathbf{u}(\xi))}{d\xi} d\xi = \int_0^1 \nabla_{\mathbf{u}} f \frac{d\mathbf{u}}{d\xi} d\xi,$$

and assume a linear dependence, often known as Roe linearization after [13], as

$$\mathbf{u}(\xi) = \mathbf{u}_l + (\mathbf{u}_r - \mathbf{u}_l)\xi, \quad \xi \in [0, 1].$$

Insertion into the above expression yields the Roe condition

$$f(\mathbf{u}_r) - f(\mathbf{u}_l) = \mathbf{A}^* (\mathbf{u}_r - \mathbf{u}_l), \quad (6.6)$$

where

$$\mathbf{A}^* = \int_0^1 \nabla_{\mathbf{u}} f(\mathbf{u}(\xi)) d\xi \quad (6.7)$$

is known as the Roe matrix. Given the essential importance of the Roe matrix, it is reasonable to ask whether its existence is guaranteed. The answer to this question is the following result [7].

**Theorem 6.3.** *Provided a system of conservation laws has a convex entropy, a Roe matrix with purely real eigenvalues exists.*

**Proof.** From (6.6) we have

$$f_r - f_l = \mathbf{A}^* (\mathbf{u}_r - \mathbf{u}_l).$$

Now assume that there exists a vector function  $\mathbf{v} = g(\mathbf{u}) : \mathbb{R}^m \rightarrow \mathbb{R}^m$  and that

$$\mathbf{v}(\xi) = \mathbf{v}_l + (\mathbf{v}_r - \mathbf{v}_l)\xi, \quad \xi \in [0, 1].$$

This yields

$$\begin{aligned} \mathbf{v}_r - \mathbf{v}_l &= \int_0^1 \frac{d}{d\xi} \mathbf{v}(\xi) d\xi = \int_0^1 \frac{d}{d\xi} g(\mathbf{u}_l + (\mathbf{u}_r - \mathbf{u}_l)\xi) d\xi \\ &= \int_0^1 (\mathbf{u}_r - \mathbf{u}_l) \nabla_{\mathbf{u}} g d\xi = \mathbf{Q}(\mathbf{u}_r - \mathbf{u}_l). \end{aligned}$$

In an equivalent manner, we recover

$$f_r - f_l = B(v_r - v_l).$$

with

$$B = \int_0^1 \nabla_v f(v(\xi)) d\xi, \quad Q = \int_0^1 \nabla_u g(u(\xi)) d\xi.$$

By the Rankine–Hugoniot condition, this implies that  $A^* = BQ$ .

If we now choose  $g(u) = \nabla_u \eta(u)$ , Theorem 3.11 guarantees that  $B$  is symmetric (or can be symmetrized). Furthermore, since  $\eta$  is convex,  $Q$  is symmetric positive definite. Since  $A^*$  is similar to the symmetric operator  $Q^{1/2}BQ^{1/2}$ , all eigenvalues of  $A^*$  are real.  $\square$

While (6.7) reflects a natural definition of  $A^*$  and its exact evaluation may be possible, it is at times complex and the process is problem dependent.

If we restrict attention to the scalar conservation law, we recover

$$|A^*| \leq \max_{w \in [u_l, u_r]} |f'(w)|,$$

and the local Lax–Friedrichs flux.

For more general cases, we may seek an approximation to (6.7) as

$$A^* = \nabla_u f\left(\frac{u_l + u_r}{2}\right),$$

or

$$A^* = \frac{1}{2}(\nabla_u f(u_l) + \nabla_u f(u_r)).$$

However, both of these definitions may be problematic, e.g., the former approximation may violate the Rankine–Hugoniot condition while the latter may result in a flux Jacobian  $A^*$  that violates hyperbolicity.

Before we continue, it is illustrative to derive the Roe condition (6.6) in a slightly different way. If we assume that the Riemann problem is dominated by one strong wave propagating at speed  $s$ , the Rankine–Hugoniot condition requires

$$f(u_r) - f(u_l) = s(u_r - u_l).$$

If the solution is also a solution to the linearized Riemann problem, we must require that

$$A^*(u_r - u_l) = s(u_r - u_l).$$

Hence, the Roe condition reflects the assumption that the solution is dominated by a single strong wave. This may be a reasonable condition, except in cases where strong shocks interact. In such cases, a Riemann solver based on the above principle will be less effective.

As we experienced for the scalar conservation law, the linearized Roe flux encounters a problem in situations where there is a rarefaction wave and a sonic point, i.e., a point where  $f(u_l) < 0 < f(u_r)$  in the scalar case.

Techniques for addressing this problem are known as entropy fixes and there are several strategies available, with many being of a problem specific nature. To discuss a more general approach we follow the idea of the entropy fix presented in [7].

One way to understand the challenge is to consider the Roe flux as

$$f^* = \frac{f(\mathbf{u}_l) + f(\mathbf{u}_r)}{2} - \frac{1}{2} |\mathbf{A}^*| (\mathbf{u}_r - \mathbf{u}_l) = \frac{f(\mathbf{u}_l) + f(\mathbf{u}_r)}{2} - \frac{1}{2} \sum_{i=1}^m |\lambda_i^*| \alpha_i s_i^*.$$

In the sonic case some  $|\lambda_k^*|$  is very close to zero and the stabilizing term does not add sufficient dissipation in this case. The goal is therefore to redefine this to increase the dissipation on the particular wave.

Let us assume that we have a linear wave with  $(\lambda^*, \mathbf{s}^*)$  as an eigenpair of  $\mathbf{A}^*$ , corresponding to wave  $k$ . The problem arises when  $\lambda_k(\mathbf{A}(\mathbf{u}_l)) = \lambda_l \leq \lambda_r = \lambda_k(\mathbf{A}(\mathbf{u}_r))$ . We refer to  $\mathbf{s}_l$  and  $\mathbf{s}_r$  as the left and right eigenvectors corresponding to  $\lambda_l$  and  $\lambda_r$ , respectively. Now split the linear wave  $\mathbf{s}^*$  into these two parts,  $\mathbf{s}_l$  and  $\mathbf{s}_r$ , and require that

$$\mathbf{s}^* = \beta \mathbf{s}_l + (1 - \beta) \mathbf{s}_r, \quad \beta \lambda_l \mathbf{s}_l + (1 - \beta) \lambda_r \mathbf{s}_r = \lambda^* \mathbf{s}^*,$$

where the latter condition ensures conservation. It follows that

$$\beta = \frac{\lambda_r - \lambda^*}{\lambda_r - \lambda_l}.$$

We can use this to redefine  $\mathbf{A}^*$  with different eigenvectors. However, it is equivalent to redefine  $\lambda^* = \beta \lambda_l$  which can then be used in the Roe flux. Alternative ways to deal with the sonic point are discussed in [10, 16].

**Example 6.4.** The Roe flux and the Lax–Friedrichs flux require the evaluation of  $|\mathbf{A}| = \mathbf{S}^{-1} |\Lambda| \mathbf{S}$ . This decomposition is often computationally demanding and may even be impossible for complex wave systems. In such situations, an approximation is needed.

A natural approach is to seek approximations to the function  $|\cdot|$ , based either on polynomials or rational functions. In the present context, we must choose such approximations with some care to ensure that the dissipation, added by the jump term, is sufficiently strong. If we assume that  $\tau_{2p}$  is an approximation to  $|\cdot|$  we must require that  $\tau_{2p}(x) \geq |x|$  for all values of  $x$ . If this is violated, the dissipation may be insufficient and the scheme may become unstable.

Following [3], we consider two different approximations. Given the nonperiodic nature of  $|\cdot|$ , it is natural to consider an approximation based on Chebyshev polynomials  $T_l(x)$  of the form

$$\tau_{2p}(x) = \sum_{l=0}^p \frac{4}{c_l \pi} \frac{(-1)^{l+1}}{(2l-1)(2l+1)} T_{2l}(x), \quad x \in [-1, 1],$$

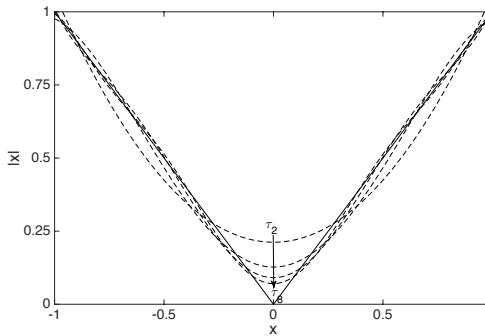
where  $c_l = 1 + \delta_{0,l}$  and  $T_{2l}(x) = \cos(2l \arccos(x))$ . It holds that

$$\| |x| - \tau_{2p}(x) \|_\infty = \frac{2}{\pi} \frac{1}{2p+1},$$

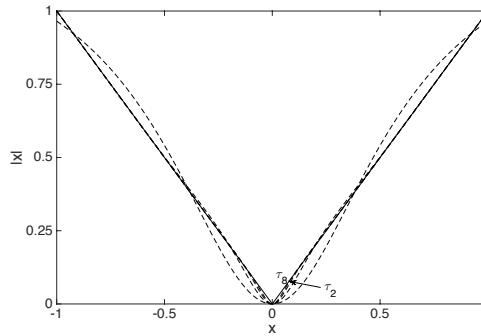
i.e., it is uniformly convergent. Although methods for the evaluation of functions of matrices are well developed [8], it is advantageous to consider the evaluation of  $T_{2l}(x)$  by its recurrence as

$$T_0(x) = 1, \quad T_2(x) = 2x^2 - 1, \quad T_{2l}(x) = 2T_2(x)T_{2l-2}(x) - T_{2l-4}(x).$$

This extends to matrices in a straightforward manner.



(a) Chebyshev approximation.



(b) Rational Newman approximation.

**Figure 6.2.** Approximations of  $|x|$  of increasing order. The dashed lines show the approximations of order 2–8 and the solid line is the exact function.

If we consider Fig. 6.2 carefully, we observe that the stability requirement  $\tau_{2p}(x) \geq |x|$  is violated due to the nature of the polynomial approximation. A simple work around is to define

$$\tau_{2p}^\varepsilon(x) = \tau_{2p}(x) + \varepsilon,$$

where  $\varepsilon > 0$  is chosen so that the stability condition is fulfilled. In [3] it is proposed to use

$$\varepsilon = \max ||x^*| - \tau_{2p}(x^*)|,$$

where  $\tau'_{2p}(x^*) = 1$  identifies points of maximum distance from  $|x|$ . The complete approximation of  $|A|$  is then given as

$$|A| \simeq |A|_{2p} = \beta \tau_{2p}^\varepsilon(\beta^{-1}A),$$

where  $\beta$  is an upper bound for the maximum eigenvalue of  $A$ .

In Chebtau2p.m we illustrate the evaluation of the Chebyshev approximation of the absolute value of a matrix. Furthermore, in Chebeps.m we show the computation of the shift to ensure stability.

---

**Script 6.1. Chebtau2p.m:** Routine to compute Chebyshev approximation of order  $2p$  to the absolute value of  $x$ .

---

```
function tau2p = Chebtau2p(x,p);
% function tau2p = Chebtau2p(x,p);
% Purpose: Evaluate Chebyshev abs approximation
% Note: x can be a matrix
tau2p = 2/pi + 0*x;
for k=1:p
    tau2p = tau2p + 4/pi*(-1)^(k+1)./(2*k-1)./(2*k+1).*cos(2*k*acos(x));
end
return
```

---

**Script 6.2. Chebeps.m:** Routine to compute shift of Chebyshev approximation of order  $2p$  to the absolute value of  $x$  to ensure stability.

---

```
function Ceps = Chebeps(p);
% function Ceps = Chebeps(p);
% Purpose: Compute shift of Chebychev based approximation to absolut value
% to ensure stability.

xs = fzero(@(x) Chebtaup(x,p)-1,0); Ceps = abs(xs) - abs(Chebtau2p(xs,p));
return

function taup = Chebtaup(x,p);
% function taup = Chebtaup(x,p);
% Purpose: Evaluate derivative of abs approximation to find shift
taup = 0;
for k=1:p
    taup = taup + 4/pi*(-1)^(k+1)./(2*k-1)./(2*k+1) ...
        .*2*k*sin(2*k*acos(x))./sqrt(1-x^2);
end
return
```

---

As an alternative to the polynomial approximation, one can consider a rational approximation. One such option is the Newman rational function [12] in the form

$$\tau_p = x \frac{r(x) - r(-x)}{r(x) + r(-x)}, \quad r(x) = \prod_{l=1}^p (x + x_l),$$

where the definition of  $x_l$  determines the accuracy of the rational approximation to  $|x|$ . This provides an interpolation at  $[-x_p, x_p]$ .

Choosing  $x_l = \xi^l$  with  $\xi = \exp(-p^{-1/2})$  yields a uniform approximation with an exponentially small error  $\mathcal{O}(\exp(-c\sqrt{p}))$  [12]. An alternative choice is the mapped Chebyshev points  $x_l = \cos(\pi(2l-1)/(4p))$ , which yield an order of approximation as  $\mathcal{O}(1/p \log p)$  [2]. As shown in Fig. 6.2, the Newman rational approximation also violates the stability condition. This issue can be resolved using the same approach discussed above.

We recover the scheme for approximating the dissipative term as

$$|A| \simeq |A|_p = \frac{\beta}{x_p} \tau_{2p}^{\varepsilon} \left( \frac{x_p}{\beta} A \right),$$

where  $\beta$  is a upper bound for the maximum eigenvalue of  $A$ .

We refer to [3] for additional approximations to  $|x|$  as well as a comparative evaluation that confirms the excellent behavior of this approach, even for very complex problems. ■

### 6.2.2 • Engquist–Osher fluxes

Let us again consider the general system of conservation laws,

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u})}{\partial x} = 0,$$

and seek the approximate solution to the Riemann problem associated with the left and right states,  $\mathbf{u}_l$  and  $\mathbf{u}_r$ . A natural extension of (6.2) is

$$\mathbf{f}^* = \mathbf{A}^+(\mathbf{u}_r) + \mathbf{A}^-(\mathbf{u}_l),$$

where  $\mathbf{A}^\pm = \mathbf{S}^{-1} \Lambda^\pm \mathbf{S}$  and  $\pm \lambda_i^\pm \geq 0$  are the eigenvalues of  $\mathbf{A}(\mathbf{u}) = \nabla_{\mathbf{u}} \mathbf{f}$ .

The assumption of the Engquist–Osher flux [5] is that there exist two functions,  $\mathbf{f}^\pm(\mathbf{u})$ , such that

$$\mathbf{f}(\mathbf{u}) = \mathbf{f}^+(\mathbf{u}) + \mathbf{f}^-(\mathbf{u}), \quad \nabla_{\mathbf{u}} \mathbf{f}^\pm(\mathbf{u}) = \mathbf{A}^\pm(\mathbf{u}).$$

Under this assumption we have

$$\int_{\mathbf{u}_l}^{\mathbf{u}_r} \mathbf{A}^\pm(\mathbf{u}) d\mathbf{u} = \mathbf{f}^\pm(\mathbf{u}_r) - \mathbf{f}^\pm(\mathbf{u}_l).$$

Following the developments for the linear Riemann problem in section 3.2, we express

$$\mathbf{f}^* = \mathbf{f}(\mathbf{u}_l) - \int_{\mathbf{u}_l}^{\mathbf{u}_r} \mathbf{A}^+(\mathbf{u}) d\mathbf{u} = \mathbf{f}(\mathbf{u}_r) + \int_{\mathbf{u}_l}^{\mathbf{u}_r} \mathbf{A}^-(\mathbf{u}) d\mathbf{u}, \quad (6.8)$$

or, in the spirit of the Roe flux,

$$\mathbf{f}^* = \frac{1}{2} (\mathbf{f}(\mathbf{u}_l) + \mathbf{f}(\mathbf{u}_r)) - \frac{1}{2} \int_{\mathbf{u}_l}^{\mathbf{u}_r} |\mathbf{A}(\mathbf{u})| d\mathbf{u}. \quad (6.9)$$

Let us first consider the scalar case. If  $f'(\mathbf{u})$  has a constant sign in  $[\mathbf{u}_l, \mathbf{u}_r]$ , one of the two expressions (6.8) vanishes and we recover (6.2). For the case  $f'(\mathbf{u}_l) < 0 < f'(\mathbf{u}_r)$ , we have

$$\mathbf{f}^* = \frac{1}{2} (\mathbf{f}(\mathbf{u}_l) + \mathbf{f}(\mathbf{u}_r)) - \frac{1}{2} \left( \int_{\mathbf{u}_s}^{\mathbf{u}_r} f'(\mathbf{u}) d\mathbf{u} - \int_{\mathbf{u}_l}^{\mathbf{u}_s} f'(\mathbf{u}) d\mathbf{u} \right) = \mathbf{f}(\mathbf{u}_s),$$

as required in the entropy satisfying scheme discussed in Ex. 5.9. It is easy to show that this flux is monotone.

For the case where  $f'(\mathbf{u}_l) > 0 > f'(\mathbf{u}_r)$ , the resulting flux is different from that in Ex. 5.9. We have

$$\mathbf{f}^* = \mathbf{f}(\mathbf{u}_l) + \mathbf{f}(\mathbf{u}_r) - \mathbf{f}(\mathbf{u}_s)$$

by direct evaluation of the flux. However, the flux remains consistent and does not lead to entropy violating solutions.

The extension of the Engquist–Osher flux to the system case is given in (6.8)–(6.9) but additional steps are needed to make this practical. As for the Roe linearization, this typically involves a linearization as  $\mathbf{u}(\xi) = \mathbf{u}_l + (\mathbf{u}_r - \mathbf{u}_l)\xi$  to enable the evaluation of the integral. A simple-minded approach is to consider each of the  $m$  waves in the system, use the scheme for the scalar problem, and integrate accordingly. However, as discussed at length in [16], this introduces an ordering problem by having to decide which wave to treat first in the integration. More elaborate integration schemes and approximations are discussed in [1, 16], including schemes for the Euler equations.

### 6.2.3 ▪ Harten–Lax–van Leer (HLL) fluxes

As discussed above, the Roe flux rests on an assumption that the Riemann problem is dominated by a single strong wave. While this is appropriate for many problems, this assumption may be too simple for more complex problems, possibly introducing spurious phenomena.

To overcome this restriction, the most immediate remedy is to assume that the Riemann problem consists of three states, separated by two waves  $s^-$  and  $s^+$  with  $s^+ > s^-$ . The Rankine–Hugoniot conditions require

$$f^* - f_l = s^-(\mathbf{u}^* - \mathbf{u}_l), \quad f_r - f^* = s^+(\mathbf{u}_r - \mathbf{u}^*),$$

which, after eliminating  $\mathbf{u}^*$ , yields

$$f^*(\mathbf{u}_l, \mathbf{u}_r) = \frac{s^+ f(\mathbf{u}_l) - s^- f(\mathbf{u}_r) + s^+ s^-(\mathbf{u}_r - \mathbf{u}_l)}{s^+ - s^-}. \quad (6.10)$$

In the special case  $s^- = -\alpha = -s^+$  and  $\alpha = \max_u |\nabla_u f|$ , the Lax–Friedrichs flux is recovered.

The extension of the flux to cases of pure upwinding yields the numerical flux

$$f^*(\mathbf{u}_l, \mathbf{u}_r) = \begin{cases} f(\mathbf{u}_l), & s^- \geq 0, \\ \frac{s^+ f(\mathbf{u}_l) - s^- f(\mathbf{u}_r) + s^+ s^-(\mathbf{u}_r - \mathbf{u}_l)}{s^+ - s^-}, & s^- \leq 0 \leq s^+, \\ f(\mathbf{u}_r), & s^+ \leq 0, \end{cases} \quad (6.11)$$

known as the Harten–Lax–van Leer (HLL) flux [7]. It is straightforward to see that such a flux is monotone, provided  $|s^\pm|$  are larger than the fastest wave speed of the system. Clearly, if  $s^- \geq 0$ , all information propagates to the right, and upwinding is appropriate; a similar conclusion follows for  $s^+ \leq 0$ .

We are left with the need to estimate the wave speeds in the system. One approach is to use

$$s^- = \min_i (\lambda_i(\mathbf{A}(\mathbf{u}_l)), \lambda_i(\mathbf{A}(\mathbf{u}^*))), \quad s^+ = \max_i (\lambda_i(\mathbf{A}(\mathbf{u}_r)), \lambda_i(\mathbf{A}(\mathbf{u}^*))),$$

where the extrema are sought over the eigenvalues of the linearized flux Jacobian. The intermediate state,  $\mathbf{u}^*$ , is evaluated by considering the conservation law on integral form over the control volume centered at  $x_{i+1/2}$ :

$$\int_{x_j}^{x_{j+1}} [\mathbf{u}(x, t^{n+1}) - \mathbf{u}(x, t^n)] dx = - \int_{t^n}^{t^{n+1}} [f(x_{j+1}, t) - f(x_j, t)] dt.$$

Assuming that  $\mathbf{u}(x_j, t) = \mathbf{u}_l$  and  $\mathbf{u}(x_{j+1}, t) = \mathbf{u}_r$ , we recover

$$\int_{x_j}^{x_{j+1}} \mathbf{u}(x, t^{n+1}) dx = \frac{h}{2}(\mathbf{u}_l + \mathbf{u}_r) - k(f(\mathbf{u}_r) - f(\mathbf{u}_l)), \quad (6.12)$$

provided  $k$  is sufficiently small. This result reflects consistency. By evaluating the integral across the two waves, we obtain

$$\begin{aligned} \int_{x_j}^{x_{j+1}} \mathbf{u}(x, t^{n+1}) dx &= \int_{x_j}^{x_{j+1/2} + ks^-} \mathbf{u}(x, t^{n+1}) dx \\ &\quad + \int_{x_{j+1/2} + ks^-}^{x_{j+1/2} + ks^+} \mathbf{u}(x, t^{n+1}) dx + \int_{x_{j+1/2} + ks^+}^{x_{j+1}} \mathbf{u}(x, t^{n+1}) dx. \end{aligned} \quad (6.13)$$

Assuming that

$$k \leq \frac{h}{2|s^\pm|} \Rightarrow \frac{k|s^\pm|}{h} \leq \frac{1}{2},$$

the combination of (6.12)–(6.13) yields

$$\int_{x_{j+1/2} + ks^-}^{x_{j+1/2} + ks^+} \mathbf{u}(x, t^{n+1}) dx = k(s^+ \mathbf{u}_r - s^- \mathbf{u}_l + f(\mathbf{u}_l) - f(\mathbf{u}_r)).$$

By defining the intermediate state  $\mathbf{u}^*$  as the average we recover

$$\mathbf{u}^* = \frac{1}{k(s^+ - s^-)} \int_{x_{j+1/2} + ks^-}^{x_{j+1/2} + ks^+} \mathbf{u}(x, t^{n+1}) dx = \frac{s^+ \mathbf{u}_r - s^- \mathbf{u}_l + f(\mathbf{u}_l) - f(\mathbf{u}_r)}{s^+ - s^-},$$

recognized as the Roe average [13].

Naturally, the fundamental idea of the HLL numerical flux construction can be extended to include additional waves. For the Euler equations an additional wave can be added, thus completing the wave system and, one would hope, achieve additional accuracy. Such an extension, known as the HLLC flux [17, 16], results in a numerical flux as

$$f^*(\mathbf{u}_l, \mathbf{u}_r) \begin{cases} f(\mathbf{u}_l), & s^- \geq 0, \\ f_l^*, & s^* \geq 0 \geq s^-, \\ f_r^*, & s^+ \geq 0 \geq s^*, \\ f(\mathbf{u}_r), & s^+ \leq 0. \end{cases} \quad (6.14)$$

The intermediate fluxes can be found through the Rankine–Hugoniot condition, and the intermediate values,  $\mathbf{u}_l^*$  and  $\mathbf{u}_r^*$ , by considerations similar to those leading to the Roe average. In the case  $s^* = \mathbf{u}$ , the HLL flux is recovered. We refer to [16] for further details.

Efforts to further improve on the above schemes by linearly combining a variety of numerical fluxes, leading to HLLX $\omega$ -schemes, are discussed in [14]. Similar to the HLL flux, an advantage of these schemes is that only the fastest speed of the system is needed.

## 6.3 • Central schemes

Let us consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to appropriate boundary and initial conditions. While we develop the schemes for the scalar problem, the extension to the system case is immediate.

In contrast to the approach leading to the Godunov scheme, let us now define the control volume as  $[x_j, x_{j+1}] \times [t^n, t^{n+1}]$ , i.e., with its central point  $x_{j+1/2} = x_j + b/2$ . The integral form of the conservation law on the cell is

$$\int_{x_j}^{x_{j+1}} [u(x, t^{n+1}) - u(x, t^n)] dx = - \int_{t^n}^{t^{n+1}} [f(u(x_{j+1}, t)) - f(u(x_j, t))] dt.$$

If we define the cell average

$$\bar{u}_{j+1/2}^n = \frac{1}{b} \int_{x_j}^{x_{j+1}} u(x, t^n) dt,$$

we recover the scheme

$$\bar{u}_{j+1/2}^{n+1} = \bar{u}_{j+1/2}^n - \frac{1}{b} \int_{t^n}^{t^{n+1}} [f(u(x_{j+1}, t)) - f(u(x_j, t))] dt.$$

The formulation of a scheme requires the construction of  $\bar{u}_{j+1/2}^n$  from the known cell averages,  $\bar{u}_j^n$ . Furthermore, we need to evaluate the integrals which require the reconstruction of  $u(\cdot, t)$ .

As for the Godunov scheme, the first step is to form an interpolating polynomial,  $p(x)$ , such that

$$\int_{x_{l-1/2}}^{x_{l+1/2}} p(x) dx = \bar{u}_l^n,$$

and  $l$  runs over the stencil. Naturally, this polynomial is unique to each control volume, i.e.,  $p(x) = p_j(x)$ . From this we construct  $\bar{u}_{j+1/2}^n$  as

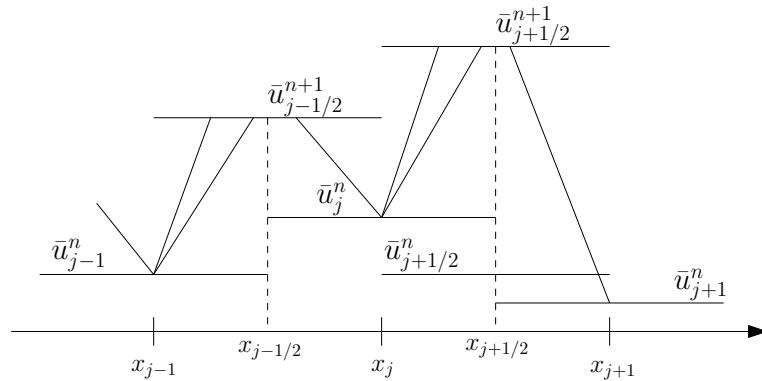
$$\bar{u}_{j+1/2}^n = \frac{1}{b} \left[ \int_{x_j}^{x_{j+1/2}} p_j(x) dx + \int_{x_{j+1/2}}^{x_{j+1}} p_{j+1}(x) dx \right],$$

the simplest expression of which is

$$\bar{u}_{j+1/2}^n = \frac{\bar{u}_j^n + \bar{u}_{j+1}^n}{2}.$$

If we denote the fastest wave speed in the system as  $\alpha$ , it is clear that, provided  $\alpha k/b \leq 1/2$ , waves from  $x_{j\pm 1}$  never reach  $x_{j\pm 1/2}$  within a time step, and the solution remains smooth, as illustrated in Fig. 6.3. This implies that the integration is simple to carry out and, in particular, no Riemann solver is needed. We recover

$$\int_{t^n}^{t^{n+1}} f(u(x_{j+1}, t)) dt = k f(\bar{u}_{j+1}^n),$$



**Figure 6.3.** Sketch of the construction of the solution for the central scheme, illustrating the characteristic waves originating from the cell centers.

which results in the scheme

$$\bar{u}_{j+1/2}^n = \frac{\bar{u}_j^n + \bar{u}_{j+1}^n}{2} - \frac{k}{h} (f(\bar{u}_{j+1}^n) - f(\bar{u}_j^n)).$$

We recognize this as the Lax–Friedrichs scheme, albeit formulated using a staggered grid, which results in a stricter CFL condition.

While the scheme is convergent and simple to implement, a disadvantage is the need to account for solutions at two sets of grids,  $x_j$  and  $x_{j+1/2}$ . To overcome this, nonstaggered versions of the central scheme has been developed [9]. Restricting attention to schemes of first order, it suffices to define

$$\bar{u}_j^n = \frac{1}{2} (\bar{u}_{j+1/2}^n + \bar{u}_{j-1/2}^n),$$

which results in the nonstaggered first order central scheme

$$\bar{u}_j^{n+1} = \frac{1}{4} (\bar{u}_{j+1}^n + 2\bar{u}_j^n + \bar{u}_{j-1}^n) - \frac{k}{2h} (f(\bar{u}_{j+1}^n) - f(\bar{u}_{j-1}^n)).$$

We again recognize this as being closely related to the Lax–Friedrichs scheme, albeit with a slightly different computation of the local approximation to  $\bar{u}_j^n$ .

Such methods, known as central schemes, have been developed in [11, 15], and their main advantage is the promise of Riemann free solvers for complex problems. While the simplest formulation results in a well-known scheme, higher-order versions are unique. We shall discuss these in more detail in section 10.3.

## 6.4 • Finite volume methods in action

Whereas the derivation of the finite volume schemes is fundamentally different from that of finite difference schemes, we have seen that the resulting first order schemes often are the same. Hence, for the majority of these schemes, very minor, if any, changes are needed in the implementations presented in section 5.3.

A central difference is the more advanced fluxes discussed in this chapter. We shall pursue an evaluation of such fluxes with a focus on methods for the Euler equations.

### 6.4.1 ▪ The Euler equations

We consider the one-dimensional Euler equations, discussed in detail in subsection 1.4.1. In the following we outline the details of the implementation of the Roe flux, the HLL flux, and the HLLC for the one-dimensional Euler equations.

#### Roe flux

The numerical Roe flux is given as

$$f^*(\mathbf{u}_l, \mathbf{u}_r) = \frac{f(\mathbf{u}_l) + f(\mathbf{u}_r)}{2} - \frac{1}{2} \sum_{i=1}^3 \alpha_i |\lambda_i^*| s_i^*,$$

where

$$\mathbf{u}_r - \mathbf{u}_l = \sum_{i=1}^3 \alpha_i s_i^*,$$

and  $(\lambda_i^*(\mathbf{u}^*), s_i^*(\mathbf{u}^*))$  are the eigenpairs of the Roe matrix  $\mathbf{A}(\mathbf{u}^*)$ .

The intermediate state  $\mathbf{u}^*$  for the Euler equations can be expressed as [13, 16]

$$\mathbf{u}^* = \frac{\sqrt{\rho_l} \mathbf{u}_l + \sqrt{\rho_r} \mathbf{u}_r}{\sqrt{\rho_l} + \sqrt{\rho_r}}, \quad H^* = \frac{\sqrt{\rho_l} H_l + \sqrt{\rho_r} H_r}{\sqrt{\rho_l} + \sqrt{\rho_r}},$$

where

$$H = \frac{E + p}{\rho}, \quad c^* = \sqrt{(\gamma - 1)(H^* - \frac{1}{2}(\mathbf{u}^*)^2)}$$

represent the enthalpy and the speed of sound, respectively.

The weights  $\alpha_i$  can be expressed explicitly as

$$\alpha_2 = \frac{\gamma - 1}{(c^*)^2} ((\rho_r - \rho_l)(H^* - (\mathbf{u}^*)^2) + \mathbf{u}^* ((\rho \mathbf{u})_r - (\rho \mathbf{u})_l) - (E_r - E_l)),$$

$$\alpha_1 = \frac{1}{2c^*} ((\rho_r - \rho_l)(\mathbf{u}^* + c^*) - ((\rho \mathbf{u})_r - (\rho \mathbf{u})_l) - c^* \alpha_2), \quad \alpha_3 = (\rho_r - \rho_l) - (\alpha_1 + \alpha_2).$$

The eigenvectors  $s_i^*$  are given in Ex. 3.3, completing the definition of the Roe numerical flux.

The implementation of this approach is illustrated in `EulerRoe.m`.

**Script 6.3. EulerRoe.m: Definition of the Roe numerical flux,  $F(\mathbf{u}, \mathbf{v})$ , for the Euler equations.**

---

```

function [numflux] = EulerRoe(u,v,gamma,lambda,maxvel)
% function [numflux] = EulerRoe(u,v,gamma,lambda,maxvel);
% Purpose: Evaluate Roe numerical flux for Euler's equation. No entropy fix
% Compute flux for u
ru = u(:,1); ruu = u(:,2); Eu = u(:,3); uu = ruu./ru;
pu = (gamma-1)*(Eu - 0.5*ruu.^2./ru); Hu = (Eu+pu)./ru;
fu = [ruu (ruu.^2./ru+pu) (Eu+pu).*ruu./ru];

```

```

% Compute flux for v
rv = v(:,1); ruv = v(:,2); Ev = v(:,3); uv = ruv./rv;
pv = (gamma-1)*(Ev - 0.5*ruv.^2./rv); Hv = (Ev+pv)./rv;
fv = [ruv (ruv.^2./rv+pv) (Ev+pv).*ruv./rv];

% Compute Roe averaged variables
uh = (sqrt(ru).*uu+sqrt(rv).*uv)./(sqrt(ru)+sqrt(rv));
Hh = (sqrt(ru).*Hu+sqrt(rv).*Hv)./(sqrt(ru)+sqrt(rv));
ch = sqrt((gamma-1)*(Hh - 0.5*uh.^2));

a2 = (gamma-1)*(rv-ru).*(Hh-uh.^2) + uh.* (ruv-ruu) - (Ev-Eu)./(ch.^2);
a1 = ((rv-ru).*(uh+ch) - (ruv-ruu) - ch.*a2)./(2*ch);
a3 = (rv-ru) - (a1+a2);

Unit = ones(length(uh),1);
s1 = [Unit uh-ch Hh-uh.*ch];
s2 = [Unit uh uh.^2/2];
s3 = [Unit uh+ch Hh+uh.*ch];

% Compute numerical flux
numflux = (fu+fv)/2 - 0.5*(bsxfun(@times,s1,a1.*abs(uh-ch)) + ...
    bsxfun(@times,s2,a2.*abs(uh)) + bsxfun(@times,s3,a3.*abs(uh+ch)));
return

```

---

## HLL flux

The implementation of the HLL flux follows closely the general discussion in subsection 6.2.3 by first defining the intermediate state:

$$\mathbf{u}^* = \frac{s^+ \mathbf{u}_r - s^- \mathbf{u}_l + f(\mathbf{u}_l) - f(\mathbf{u}_r)}{s^+ - s^-},$$

where the velocities of the fastest waves are estimated as

$$s^- = \min_i(\lambda_i(\mathbf{A}(\mathbf{u}_l)), \lambda_i(\mathbf{A}(\mathbf{u}^*))), \quad s^+ = \max_i(\lambda_i(\mathbf{A}(\mathbf{u}_r)), \lambda_i(\mathbf{A}(\mathbf{u}^*))).$$

The eigenvalues, derived in Ex. 3.3, are given as  $(u - c, c, u + c)$ .

The intermediate flux is given as

$$f^*(\mathbf{u}_l, \mathbf{u}_r) = \frac{s^+ f(\mathbf{u}_l) - s^- f(\mathbf{u}_r) + s^+ s^- (\mathbf{u}_r - \mathbf{u}_l)}{s^+ - s^-},$$

and the resulting numerical flux is given in (6.11). The implementation of the numerical flux is outlined in EulerHLL.m.

**Script 6.4. EulerHLL.m: Definition of the HLL numerical flux,  $F(u, v)$ , for the Euler equations.**

---

```

function [ numflux ] = EulerHLL(u,v,gamma,lambda,maxvel)
% function [numflux] = EulerHLL(u,v,gamma,lambda,maxvel);
% Purpose: Evaluate HLL numerical flux for Euler's equation

% Compute flux for u
r = u(:,1); ru = u(:,2); E = u(:,3); pu = (gamma-1)*(E - 0.5*ru.^2./r);
fu = [ru (ru.^2./r+pu) (E+pu).*ru./r];
cu = sqrt(gamma*pu./r); uu = ru./r; sm = min([uu-cu uu uu+cu], [], 2);

```

```

% Compute flux for v
r = v(:,1); ru = v(:,2); E = v(:,3); pv = (gamma-1)*(E - 0.5*ru.^2./r);
fv = [ ru (ru.^2./r+pv) (E+pv).*ru./r ];
cv = sqrt(gamma*pv./r); uv = ru./r; sp = max([ uv-cv uv uv+cv ],[],2);

% Compute Roe average and extreme velocities
us = bsxfun(@times,bsxfun(@times,v,sp) - ...
    bsxfun(@times,u,sm)+fu-fv),1./(sp-sm));
r = us(:,1); ru = us(:,2); E = us(:,3);
ps = (gamma-1)*(E - 0.5*ru.^2./r); cs = sqrt(gamma*ps./r); uus = ru./r;
sm = min([ sm uus-cs uus uus+cs ],[],2); sp = max([ sp uus-cs uus uus+cs ],[],2);

% Compute numerical flux
fs = bsxfun(@times,(bsxfun(@times, fu , sp)-bsxfun(@times, fv , sm)+ ...
    bsxfun(@times, v-u , sp.*sm)),1./(sp-sm));
numflux = bsxfun(@times, fu , (sm>0))+ ...
    bsxfun(@times, fs , ((sm<=0).*(sp>=0))+bsxfun(@times, fv , (sp<0));
return

```

---

### HLLC flux

As for the HLL flux, the challenge in the HLLC flux is to compute the velocities and, in this particular case, the intermediate velocity  $s^*$ . Following [16, 17] we define

$$s^* = \frac{p_r - p_l + (\rho u)_l(s^- - u_l) - (\rho u)_r(s^+ - u_r)}{\rho_l(s^- - u_l) - \rho_r(s^+ - u_r)},$$

where  $(\rho, u)$  are the primitive variables of the Euler equations and  $s^\pm$  are estimates of the maximum and minimum wave speeds as

$$s^- = \min(u_l - c_l, u_l, u_l + c_l), \quad s^+ = \max(u_r - c_r, u_r, u_r + c_r).$$

Defining the pressure

$$p_{lr} = \frac{1}{2} (p_r + p_l + \rho_l(s^- - u_l)(s^* - u_l) + \rho_r(s^* - u_r)(s^+ - u_r)),$$

and  $D = [0, 1, s^*]^T$ , the HLLC flux is given by the two intermediate fluxes

$$f_l^* = \frac{s^*(s^- u_l - f(u_l)) + s^- p_{lr} D}{s^- - s^*}, \quad f_r^* = \frac{s^*(s^+ u_r - f(u_r)) + s^+ p_{lr} D}{s^+ - s^*}.$$

The numerical flux is finally obtained by (6.14). The complete implementation is given in EulerHLLC.m

---

**Script 6.5. EulerHLLC.m: Definition of the HLLC numerical flux  $F(u, v)$  for the Euler equations.**

---

```

function [ numflux ] = EulerHLLC(u,v,gamma,lambda,maxvel)
% function [ numflux ] = EulerHLLC(u,v,gamma,lambda,maxvel);
% Purpose: Evaluate HLLC numerical flux for Euler's equation.

% Compute flux for u
ru = u(:,1); ruu = u(:,2); E = u(:,3); pu = (gamma-1)*(E - 0.5*ruu.^2./ru);
fu = [ ruu (ruu.^2./ru+pu) (E+pu).*ruu./ru ];

```

```

cu = sqrt (gamma*pu./ ru) ; uu = ruu./ ru; sm = min([uu-cu uu uu+cu],[],2);

% Compute flux for v
rv = v(:,1); ruv = v(:,2); E = v(:,3); pv = (gamma-1)*(E - 0.5*ruv.^2./ rv);
fv = [ruv (ruv.^2./ rv+pv) (E+pv).* ruv./ rv];
cv = sqrt (gamma*pv./ rv); uv = ruv./ rv; sp = max([uv-cv uv uv+cv],[],2);

% Compute Roe pressure
ss = (pv-pu + ruu.* (sm-uu) - ruv.* (sp-uv))./ (ru .* (sm-uu)-rv .* (sp-uv));
Plr = 0.5*(pu+pv+ru .* (sm-uu) .* (ss-uu) + rv .* (sp-uv) .* (ss-uv));

% Compute numerical flux
Ds = 0*u; Ds(:,2)=1; Ds(:,3)=ss;
fus = bsxfun (@times, bsxfun (@times, bsxfun (@times, u,sm)-fu , ss )+ ...
bsxfun (@times, Ds, Plr.*sm) ,1./ (sm-ss));
fvs = bsxfun (@times, bsxfun (@times, bsxfun (@times, v,sp)-fv , ss )+ ...
bsxfun (@times, Ds, Plr.*sp) ,1./ (sp-ss));
numflux = bsxfun (@times, fu , (sm>=0))+bsxfun (@times, fus , ((sm<=0).* (ss>0)))+ ...
bsxfun (@times, fvs , ((ss<=0).* (sp>=0)))+bsxfun (@times, fv ,
(sp<=0));
return

```

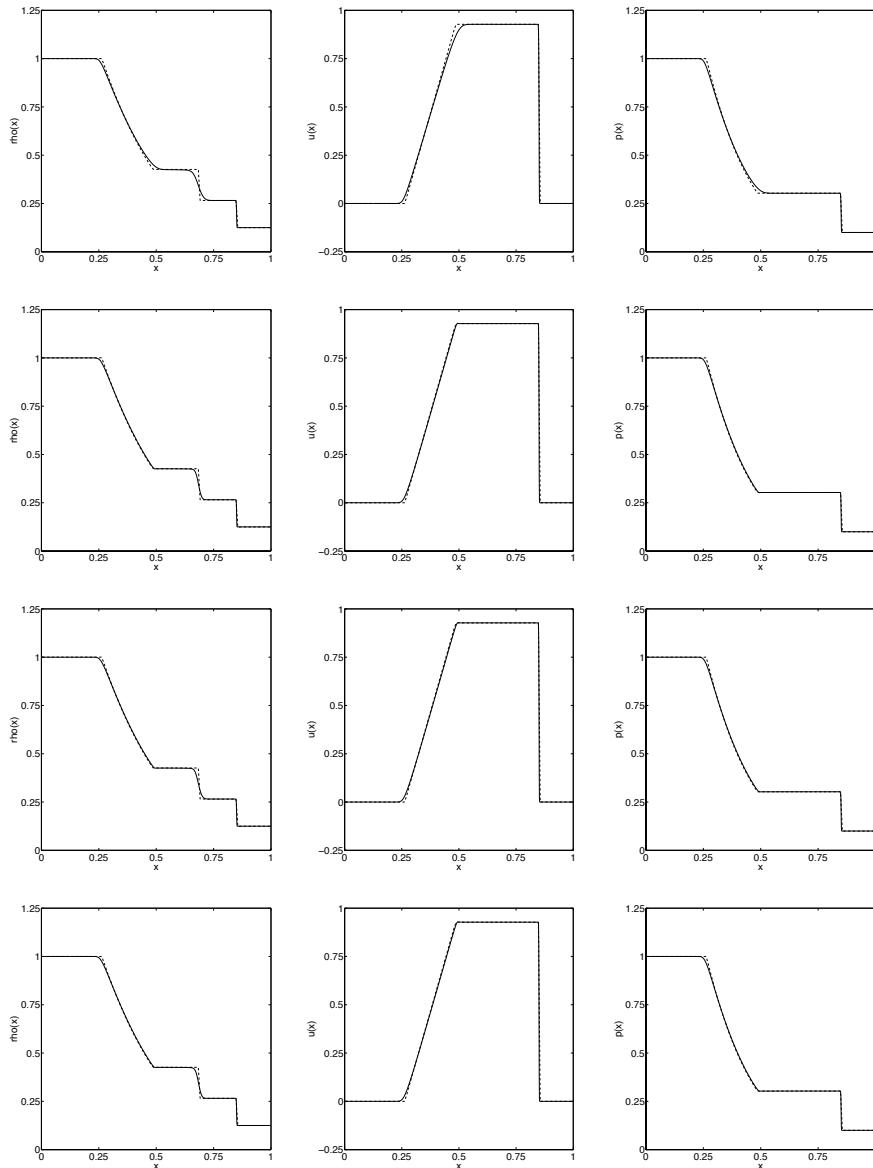
We first consider Sod's problem, introduced in subsection 1.4.1, to evaluate the different fluxes. Figure 6.4 illustrates the results obtained with the three fluxes and, as a reference, the local Lax–Friedrichs flux. While the use of the more advanced fluxes yields visible improvements when compared to the local Lax–Friedrichs flux, the differences between them are minimal. The only visible improvement is a slight decrease in the smearing of the contact discontinuity when the HLLC numerical flux is applied.

To further investigate the value of a carefully designed numerical flux, we show in Fig. 6.5 the solution of the shock-entropy problem, defined in subsection 1.4.1, obtained with different numerical fluxes. As for the previous case, use of the more advanced fluxes clearly improves the quality of the solution when compared to the results obtained with the local Lax–Friedrichs flux.

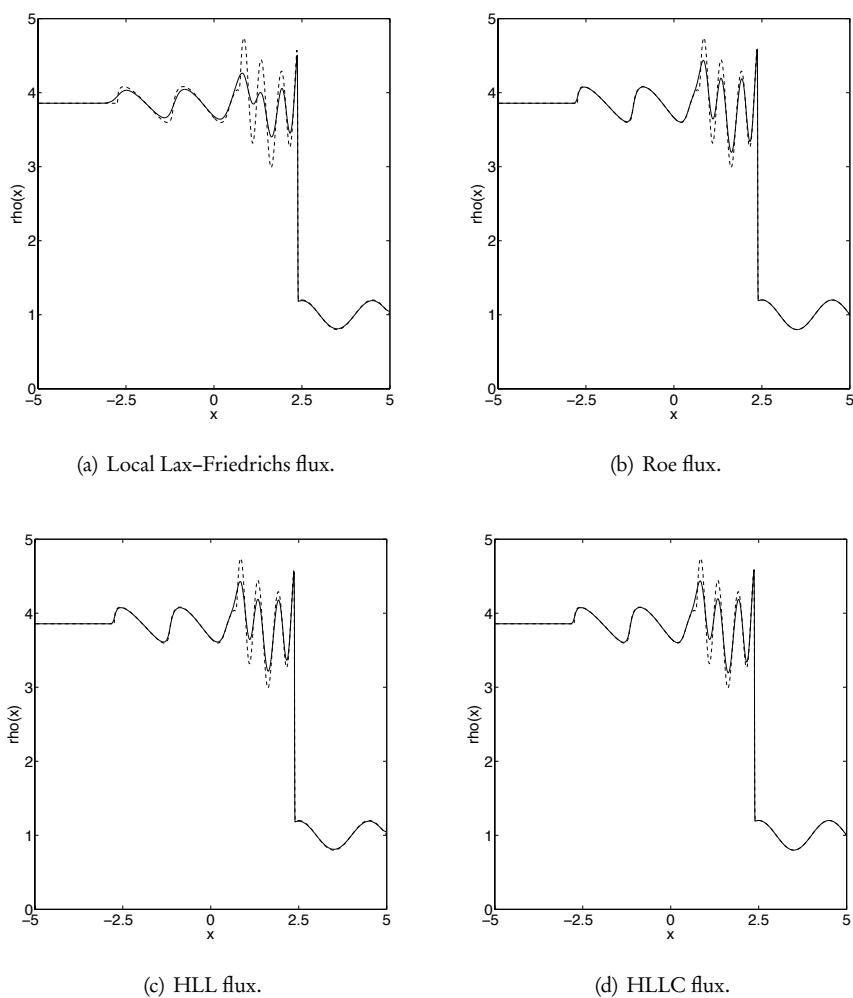
Whereas the accuracy clearly supports the use of more advanced fluxes, one should not forget the increased computational cost associated with these methods. Table 6.1 lists approximate and scaled execution times for solving Sod's problem using four different fluxes. As expected, doubling the number of grid points results in, approximately, a fourfold increase in execution time. All the advanced fluxes yield, roughly, the same computational costs, and twice that of the local Lax–Friedrichs flux. However, given the improved accuracy, the advanced numerical fluxes remain the most appropriate choice.

**Table 6.1.** Approximate (scaled) execution times for the solving the Sod problem with different numerical fluxes.

N	Local Lax–Friedrichs	Roe flux	HLL flux	HLLC flux
512	1.0	1.5	2.0	1.8
1024	2.5	4.0	5.1	5.1
2048	7.1	11.6	14.0	13.6
4096	23.5	39.0	47.1	47.0



**Figure 6.4.** Solution of Sod's problem using different numerical fluxes. All computations use  $N = 1024$  grid points. First row employs a local Lax-Friedrichs flux, second row a Roe flux, third row an HLL flux, and last row an HLLC numerical flux. Columns reflect the density,  $\rho$ , the velocity,  $u$ , and the pressure,  $p$ . The dashed lines indicate the exact solution at  $T = 0.2$ .



**Figure 6.5.** Shock-entropy wave solution to the Euler equations, obtained with different numerical fluxes and  $N = 2048$ . The dashed lines represent a high accuracy numerical reference solution.

## References

- [1] John B. Bell, Phillip Colella, and John A. Trangenstein. Higher order Godunov methods for general systems of hyperbolic conservation laws. *Journal of Computational Physics*, 82(2):362–397, 1989.
- [2] Lev Brutman and Eli Passow. Rational interpolation to  $|x|$  at the Chebyshev nodes. *Bulletin of the Australian Mathematical Society*, 56(01):81–86, 1997.
- [3] Manuel J. Castro, José M. Gallardo, and Antonio Marquina. A class of incomplete Riemann solvers based on uniform rational approximations to the absolute value function. *Journal of Scientific Computing*, 60(2):363–389, 2014.

- [4] Michael G. Crandall and Andrew Majda. Monotone difference approximations for scalar conservation laws. *Mathematics of Computation*, 34(149):1–21, 1980.
- [5] Björn Engquist and Stanley Osher. One-sided difference approximations for nonlinear conservation laws. *Mathematics of Computation*, 36(154):321–351, 1981.
- [6] Sergei K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 89(3):271–306, 1959.
- [7] Amiram Harten, Peter D. Lax, and Bram van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, 1983.
- [8] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*, volume 104 in Other Titles in Applied Mathematics. SIAM, 2008.
- [9] G.-S. Jiang, D. Levy, C.-T. Lin, S. Osher, and E. Tadmor. High-resolution nonoscillatory central schemes with nonstaggered grids for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 35(6):2147–2168, 1998.
- [10] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*, volume 31 of Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002.
- [11] Haim Nessyahu and Eitan Tadmor. Non-oscillatory central differencing for hyperbolic conservation laws. *Journal of Computational Physics*, 87(2):408–463, 1990.
- [12] David J. Newman. Rational approximation to the absolute value of x. *Michigan Mathematics Journal*, 11:11–14, 1964.
- [13] Philip L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.
- [14] Birte Schmidtmann and Manuel Torrilhon. *A Hybrid Riemann Solver for Large Hyperbolic Systems of Conservation Laws*. preprint, arXiv:1607.05721, 2016.
- [15] Eitan Tadmor. Approximate solutions of nonlinear conservation laws. In: Alfio Quarteroni, editor, *Advanced Numerical Approximations of Nonlinear Hyperbolic Equations*, volume 1697 of Lecture Notes in Mathematics, pages 1–149. Springer, 1998.
- [16] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer Science+Business Media, 2009.
- [17] Eleuterio F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4(1):25–34, 1994.

## Chapter 7

# Beyond one dimension

While the primary focus of this text is on one-dimensional problems, it is valuable to realize that the extension of the general concepts and related computational building blocks to problems beyond one spatial dimension is relatively straightforward.

Let us briefly consider the general conservation law

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}) = 0, \quad (\mathbf{x}, t) \in \Omega_x \times \Omega_t,$$

where  $\Omega_t = [0, T]$  and  $\Omega_x \subset \mathbb{R}^d$ . In this case,  $\mathbf{u}(\mathbf{x}, t) : \Omega_x \times \Omega_t \rightarrow \mathbb{R}^m$  and  $\mathbf{f}(\mathbf{u}) : \mathbb{R}^m \rightarrow \mathbb{R}^m$ .

If we restrict our attention to Cartesian computational domains, the majority of the computational tools considered so far can be applied with minor modifications. To realize this, let us integrate the conservation law over a control volume  $\omega_x \times \omega_t$  to obtain

$$\int_{\omega_x} \mathbf{u}(\mathbf{x}, t_2) - \mathbf{u}(\mathbf{x}, t_1) d\mathbf{x} = - \int_{\omega_t} \int_{\omega_x} \nabla \cdot \mathbf{f}(\mathbf{u}, t) d\mathbf{x} dt.$$

We now apply Gauss's rule to recover

$$\int_{\omega_x} \mathbf{u}(\mathbf{x}, t_2) - \mathbf{u}(\mathbf{x}, t_1) d\mathbf{x} = - \int_{\omega_t} \int_{\partial \omega_x} \hat{\mathbf{n}} \cdot \mathbf{f}(\mathbf{u}, t) d\mathbf{x} dt,$$

where  $\hat{\mathbf{n}}$  represents the outward pointing normal along the boundary,  $\partial \omega_x$  of  $\omega_x$ .

If we restrict attention to a simple two-dimensional Cartesian domain,  $\omega_x = [x_1, y_1] \times [x_2, y_2]$ , we have

$$\begin{aligned} \int_{\omega_x} \mathbf{u}(\mathbf{x}, t_2) - \mathbf{u}(\mathbf{x}, t_1) d\mathbf{x} \\ = - \int_{\omega_t} \left[ \int_{y_1}^{y_2} f(\mathbf{u}(x_2, y)) - f(\mathbf{u}(x_1, y)) dy + \int_{x_1}^{x_2} f(\mathbf{u}(x, y_2)) - f(\mathbf{u}(x, y_1)) dx \right] dt. \end{aligned}$$

Recalling (1.1) as the original derivation of the conservation law in integral form, we immediately recognize this as a generalization of the one-dimensional case.

Introducing a grid  $(x_i, y_j) = (x_0 + i b_x, y_0 + j b_y)$  with  $b_x = \frac{L_x}{N_x}$  and  $b_y = \frac{L_y}{N_y}$ , we can define the cell average as

$$\bar{u}_{ij}(t) = \frac{1}{b_x b_y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} u(x, t) dx.$$

If we now approximate the surface integral by the second order accurate mid-point rule, we recover Godunov's method of the form

$$\int_{\omega_t} \int_{y_{j-1/2}}^{y_{j+1/2}} f(u(x_2, y)) - f(u(x_1, y)) dy = b_y (f(u^*(x_{i+1/2}, y_j)) - f(u^*(x_{i-1/2}, y_j))),$$

and

$$\int_{\omega_t} \int_{x_{i-1/2}}^{x_{i+1/2}} f(u(x, y_2)) - f(u(x, y_1)) dx = b_x (f(u^*(x_i, y_{j+1/2})) - f(u^*(x_i, y_{j-1/2}))).$$

Assuming that  $\Omega_t = [t^n, t^{n+1}]$ , we recover the two-dimensional finite volume method:

$$\begin{aligned} \bar{u}_{ij}^{n+1} = \bar{u}_{ij}^n - k \left( \frac{f(u^*(x_{i+1/2}, y_j)) - f(u^*(x_{i-1/2}, y_j))}{b_x} \right. \\ \left. + \frac{f(u^*(x_i, y_{j+1/2})) - f(u^*(x_i, y_{j-1/2}))}{b_y} \right). \end{aligned}$$

The only open question is the computation of  $u^*$ , which is a problem similar to what we discussed for the one-dimensional case. While there are genuinely multi-dimensional ways to accomplish this [2, 9, 7, 5], we focus here on the extension of the one-dimensional methods, applied in a dimension-by-dimension fashion, i.e., to evaluate  $f(u^*(x_{i+1/2}, y_j))$ , one assumes  $y_j$  is a constant and constructs  $u^*$  along  $x$  only.

The extension to more spatial dimensions is immediate, provided we restrict ourselves to Cartesian grids. However, the above derivation is not restricted to such grids and can be developed also for much more general grids, albeit at the cost of additional complexity. While such methods on unstructured grids are used widely, their properties are not substantially different from what we discuss here, and we shall not discuss them in detail. We refer to [3, 8, 10, 12, 11] for a thorough discussion of such schemes and their practical implementation and use.

With the close connection to the one-dimensional schemes, there are very few essential differences from the results of the analysis of the one-dimensional cases. In particular, for Cartesian grids, the results of the error analysis carry over from the one-dimensional case, with an  $L^1$ -error of  $\mathcal{O}(\sqrt{h})$ , where  $h = \max(b_x, b_y)$  [3, 1]. For more general grids, this deteriorates to  $\mathcal{O}(h^{1/4})$  [3, 1].

The remaining minor detail is the definition of the CFL condition, which we recall as

$$\frac{k|a|}{h} \leq 1,$$

for the one-dimensional wave problem with the speed of propagation  $\alpha$ . For the  $d$ -dimensional case, this generalizes as

$$k \sum_{i=1}^d \frac{|\alpha_i|}{h_i} \leq 1,$$

where  $\alpha_i$  represents the maximum velocity along direction  $i$ , using a grid spacing of  $h_i$ .

## 7.1 • Two-dimensional monotone schemes in action

To illustrate the relative ease by which the one-dimensional methods can be extended to two-dimensional problems on Cartesian domains, we return to the test cases introduced in subsection 1.4.2.

### 7.1.1 • Burgers' equation

We consider the direct extension of Burgers' equation as

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial u^2}{\partial y} = 0, \quad (x, y) \in [0, 1]^2,$$

subject to periodic boundary conditions and the initial condition

$$u(x, y) = \sin(4\pi(x + y/2)).$$

As for one-dimensional cases, discussed in detail in subsection 1.4.2, we choose  $k = \text{CFL} \min(h_x, h_y)/\alpha$ , where

$$\alpha = \max_{\Omega} |\sqrt{2}f'(u)| = 2\sqrt{2} \max_{\Omega} |u|$$

is an estimate of the maximum velocity.

In the driver `BurgersMDriver2D.m` we set the initial condition, specify the computational domain, the grid size  $h_x$  and  $h_y$ , and the CFL number.

**Script 7.1. *BurgersMDriver2D.m: Driver routine for solving the two-dimensional Burgers' equation using a monotone scheme.***

---

```
% Driver script for solving the 2D Burgers' equations using a monotone scheme
clear all

% Set problem parameters
Lx = 1; Ly = 1; Nx = 256; Ny = 256; hx = Lx/Nx; hy = Ly/Ny;
FinalTime = 0.1; CFL = 0.9;

% Define domain and initial conditions
xv = [0:hx:Lx]; yv = [0:hy:Ly]; [x,y] = meshgrid(xv,yv);
u = sin(4*pi*(x+y/2));

% Solve Problem
[u] = BurgersM2D(x,y,u,hx,hy,CFL,FinalTime);
```

---

The Burgers' equation is integrated in time in `BurgersM2D.m`.

---

**Script 7.2. *BurgersM2D.m*: Temporal integration routine for solving the two-dimensional Burgers' equation using a monotone scheme.**

---

```
function [u] = BurgersM2D(x,y,u,hx,hy,CFL,FinalTime)
% function [u] = BurgersM2D(u,hx,hy,CFL,FinalTime)
% Purpose : Integrate 2D Burgers equation until FinalTime
% using a monotone scheme.

time = 0; tstep = 0;

% integrate scheme
while (time<FinalTime)
    % Decide on timestep
    maxvel = max(max(2*sqrt(2.0)*abs(u))); k = CFL*min(hx,hy)/maxvel/2;
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    u = u + k*BurgersMrhs2D(x,y,u,hx,hy,k,maxvel);
    time = time+k; tstep = tstep+1;
end
return
```

---

Function BurgersMrhs2D.m returns the right-hand side of the two-dimensional Burgers' equation and imposes boundary conditions.

---

**Script 7.3. *BurgersMrhs2D.m*: Definition of right-hand side for the two-dimensional Burgers' equation.**

---

```
function [du] = BurgersMrhs2D(x,y,u,hx,hy,k,maxvel)
% function [du] = BurgersMrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D Burgers equation
% using a monotone scheme
Nxy = size(x); Nx = Nxy(2); Ny = Nxy(1); du = zeros(Ny,Nx);

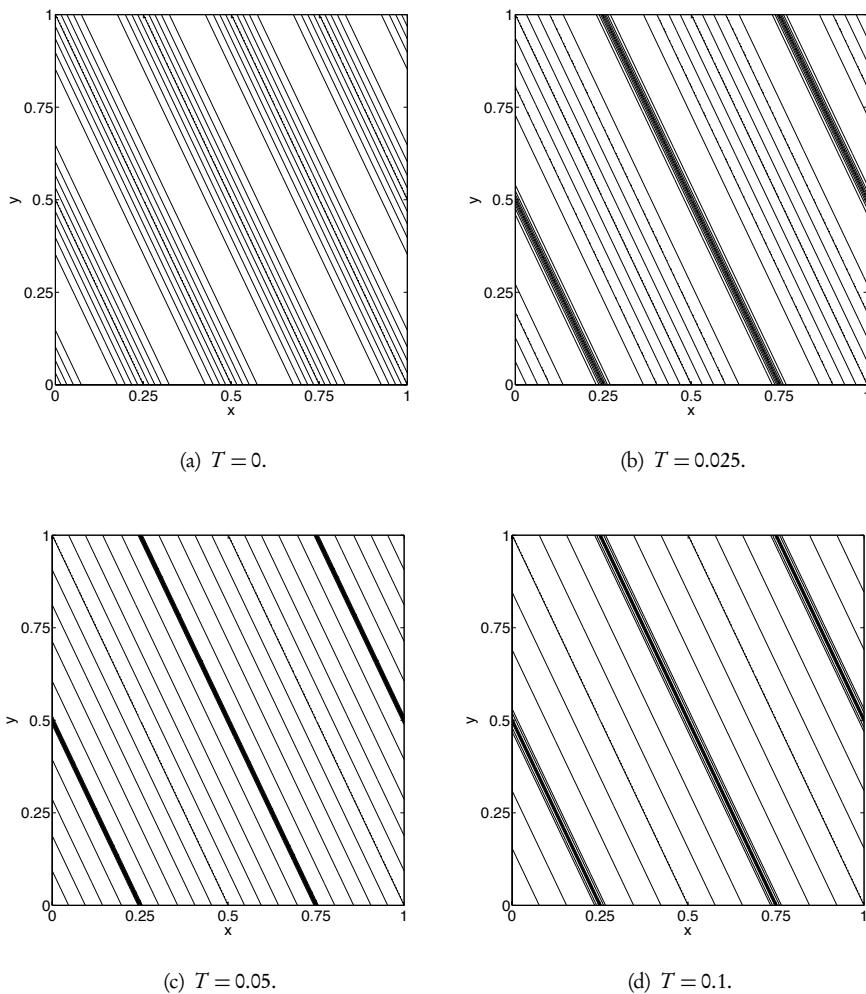
% Extend data and assign boundary conditions in x-direction
for i=1:Ny
    [xe,ue] = extend(x(i,:),u(i,:),hx,1,'P',0,'P',0);
    % Update residual
    du(i,:) = -(BurgersLF(ue(2:Nx+1),ue(3:Nx+2),0,maxvel) - ...
                BurgersLF(ue(1:Nx),ue(2:Nx+1),0,maxvel))/hx;
end

% Extend data and assign boundary conditions in y-direction
for j=1:Nx
    [xe,ue] = extend(y(:,j),u(:,j),hy,1,'P',0,'P',0);
    % Update residual
    du(:,j) = du(:,j) - (BurgersLF(ue(2:Ny+1),ue(3:Ny+2),0,maxvel) - ...
                           BurgersLF(ue(1:Ny),ue(2:Ny+1),0,maxvel))/hy;
end
return
```

---

The numerical flux is identical to the one-dimensional flux discussed at length in subsection 5.3.3. In this particular case we choose the Lax-Friedrichs flux, but others can likewise be used.

Figure 7.1 shows contours of the solution at different times. As should be expected from the understanding of the conservation law, the smooth initial condition steepens and forms a stationary shock which, ultimately, dissipates.



**Figure 7.1.** Shock formation in the two-dimensional Burgers' equation at different times.  $N_x = N_y = 256$ .

### 7.1.2 • Nonconvex problem

Let us also consider the slightly more complex nonconvex scalar conservation law discussed in subsection 1.4.2. Following the layout of the implementation for Burgers' equation, we show in `KPPMDriver2D.m` the driver routine for the two-dimensional Kolmogorov–Petrovskii–Piskunov (KPP) equation.

**Script 7.4.** `KPPMDriver2D.m`: Driver routine for solving the two-dimensional KPP equation using a monotone scheme.

---

```
% Driver script for solving the 2D KPP equations using a monotone scheme
clear all
```

```
% Set problem parameters
```

---

```

Nx = 399; Ny = 399; hx = 4.0/Nx; hy = 4.0/Ny;
FinalTime = 1.0; CFL = 0.9;

% Define domain and initial conditions
xv = [-2:hx:2]; yv = [-2.5:hy:1.5]; [x,y] = meshgrid(xv,yv);
u = pi/4 + (sqrt(x.^2+y.^2)<=1)*13*pi/4;

% Solve Problem
[u] = KPPM2D(x,y,u,hx,hy,CFL,FinalTime);

```

---

The two-dimensional KPP equation is integrated in time in `KPPM2D.m`, while calling `KPPMrhs2D.m` returns the computation of the right-hand side of the two dimensional KPP equation. Boundary conditions are likewise imposed in this routine.

**Script 7.5. `KPPM2D.m`: Temporal integration routine for solving the two-dimensional KPP equation using a monotone scheme.**

---

```

function [u] = KPPM2D(x,y,u,hx,hy,CFL,FinalTime);
% function [u] = KPPM2D(x,y,u,hx,hy,CFL,FinalTime);
% Purpose : Integrate 2D KPP equation until FinalTime
% using a monotone scheme.
time = 0; tstep = 0;

% integrate scheme
k = CFL*min(hx,hy)/2; maxvel=1;
while (time<FinalTime)
    % Decide on timestep
    if (time+k>FinalTime) k = FinalTime-time; end
    u = u + k*KPPMrhs2D(x,y,u,hx,hy,k,maxvel);
    time = time+k; tstep = tstep+1;
end
return

```

---

**Script 7.6. `KPPMrhs2D.m`: Definition of right hand side for the two-dimensional KPP equation.**

---

```

function [du] = KPPMrhs2D(x,y,u,hx,hy,k,maxvel);
% function [du] = KPPMrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D KPP equation
% using a monotone method
Nxy = size(x); Nx = Nxy(2); Ny = Nxy(1); du = zeros(Ny,Nx);

% Extend data and assign boundary conditions in x-direction
for i=1:Ny
    [xe,ue] = extend(x(i,:),u(i,:),hx,1,'D',pi/4,'D',pi/4);
    % Update residual
    du(i,:) = -(KPPxLF(ue(2:Nx+1),ue(3:Nx+2),0,maxvel) - ...
                KPPxLF(ue(1:Nx),ue(2:Nx+1),0,maxvel))/hx;
end

% Extend data and assign boundary conditions in y-direction
for j=1:Nx
    [xe,ue] = extend(y(:,j),u(:,j),hy,1,'D',pi/4,'D',pi/4);
    % Update residual
    du(:,j) = du(:,j) - (KPPyLF(ue(2:Ny+1),ue(3:Ny+2),0,maxvel) - ...
                           KPPyLF(ue(1:Ny),ue(2:Ny+1),0,maxvel))/hy;
end
return

```

---

Since the flux is more complex than in the case of Burgers' equation, we need to define the two flux terms individually. This is as shown in `KPPxLF.m` and `KPPyLF.m`, which define the  $x$  and  $y$  flux. The numerical flux is a Lax–Friedrichs flux.

**Script 7.7. `KPPxLF.m`: Lax–Friedrichs flux along  $x$  for KPP equation.**

---

```
function [numflux] = KPPxLF(u,v,lambda,maxvel)
% function [numflux] = KPPxLF(u,v,lambda,maxvel);
% Purpose: Evaluate Lax Friedrich numerical flux
% for x component of KPP equation

fu = sin(u); fv = sin(v);
numflux = (fu+fv)/2 - maxvel/2*(v-u);
return
```

---

**Script 7.8. `KPPyLF.m`: Lax–Friedrichs flux along  $y$  for KPP equation.**

---

```
function [numflux] = KPPyLF(u,v,lambda,maxvel)
% function [numflux] = KPPyLF(u,v,lambda,maxvel);
% Purpose: Evaluate Lax Friedrich numerical flux
% for y component of KPP equation

fu = cos(u); fv = cos(v);
numflux = (fu+fv)/2 - maxvel/2*(v-u);
return
```

---

In Fig. 7.2 we illustrate convergence of the monotone scheme to the reference solution for increasing resolution. Although the theoretical foundation for the monotone schemes does not extend to the nonconvex flux, the behavior of the scheme is clearly as expected.

### 7.1.3 • The Euler equations

Consider the two-dimensional version of the Euler equations as discussed in detail in subsection 1.4.2.

The software is developed following the standard pattern, with the driver routine setting up the initial condition, defining the computational domain, and the parameters in `EulerMDriver2D.m`

**Script 7.9. `EulerMDriver2D.m`: Driver routine for solving the two-dimensional Euler equation using a monotone scheme.**

---

```
% Driver script for solving the 2D Euler equations using a monotone scheme
clear all

% Set problem parameters for isentropic vortex
CFL = 0.5; gamma= 1.4;

% Define domain and initial conditions
% FinalTime = 1.0;
% Lx = 10; Ly = 10; Nx = 399; Ny = 399; bx = Lx/Nx; by = Ly/Ny;
% xv = [-5:bx:5]; yv = [-5:by:5]; [x,y] = meshgrid(xv,yv);
% x0=0.0; y0=0.0; u0=1.0; v0=0; beta=5.0;
% [r,ru,rv,E] = IsentropicVortex2D(x,x0,u0,y,y0,v0,gamma,beta,0);

% Set problem parameters for 2D Riemann problems
```

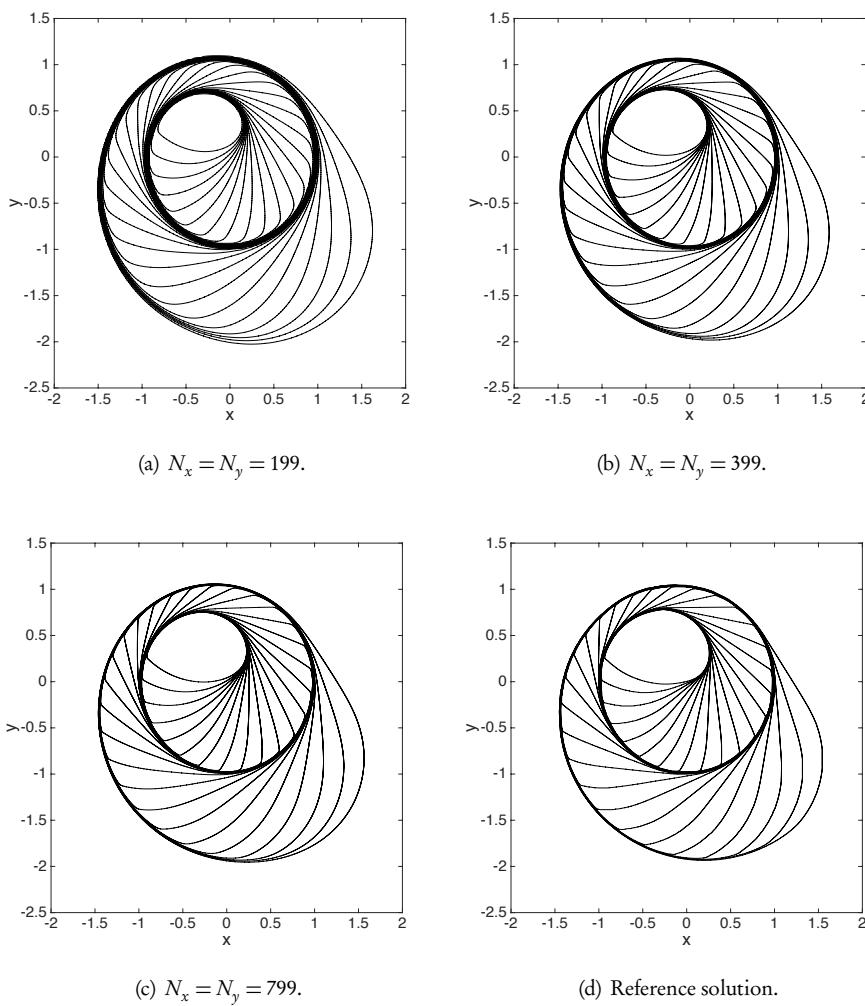


Figure 7.2. Convergence of solution to nonconvex KPP equation.

$Lx = 1$ ;  $Ly = 1$ ;  $Nx = 199$ ;  $Ny = 199$ ;  $hx = Lx/Nx$ ;  $hy = Ly/Ny$ ;  
 $RiemannProbCase = 4$ ;

```
% Define domain and initial conditions
xv = [0:hx:Lx]; yv = [0:hy:Ly]; [x,y] = meshgrid(xv,yv);
[r,ru,rv,E,FinalTime] = Riemann2D(x,y,gamma,RiemannProbCase);

% Solve Problem
q = zeros(Ny+1,Nx+1,4);
q(:,:,1) = r; q(:,:,2)=ru; q(:,:,3)=rv; q(:,:,4)=E;
[q] = EulerM2D(x,y,q,hx,hy,gamma,CFL,FinalTime);
```

EulerM2D.m performs the temporal integration by evaluating the right-hand side through EulerMrhs2D.m, where the boundary conditions are also imposed. In this case we restrict ourselves to constant solutions outside of the computational domain.

---

**Script 7.10. EulerM2D.m: Temporal integration routine for solving the two-dimensional Euler equation using a monotone scheme.**

---

```

function [q] = EulerM2D(x,y,q,hx,hy,gamma,CFL,FinalTime)
% function [q] = EulerMDriver2D(x,y,q,hx,hy,gamma,CFL,FinalTime)
% Purpose : Integrate 2D Euler equation until FinalTime using
% a monotone scheme.
time = 0; tstep = 0;

while (time<FinalTime)
    % Set timestep
    r = q(:,:,1); ru = q(:,:,2); rv = q(:,:,3); E = q(:,:,4);
    p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r); c = sqrt(gamma*p./r);
    maxvelu = max(max(c+abs(ru./r))); maxvelv = max(max(c+abs(rv./r)));
    k = CFL*min(hx,hy)/sqrt(2)/sqrt(maxvelu.^2+maxvelv.^2);
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    [dq] = EulerMrhs2D(x,y,q,gamma,hx,hy,k);
    q = q+k*dq;
    time = time+k; tstep = tstep+1;
end
return

```

---

**Script 7.11. EulerMrhs2D.m: Definition of right-hand side for the two-dimensional Euler equation.**

---

```

function [dq] = EulerMrhs2D(x,y,q,gamma,hx,hy,k);
% function [dq] = EulerMrhs2D(x,y,q,gamma,hx,hy,k);
% Purpose: Evaluate right-hand side for the two-dimensional
% Euler equations using monotone method
Nxy = size(x); Nx = Nxy(2); Ny = Nxy(1); dq = zeros(Ny,Nx,4);
qex = zeros(Nx+2,4); qey = zeros(Ny+2,4);

% Apply monotone scheme in the x-direction
for i=1:Ny
    % Extend data and assign boundary conditions in x-direction
    [xe,qex(:,1)] = extend(x(i,:),q(i,:,1),hx,1,'N',0.0,'N',0.0);
    [xe,qex(:,2)] = extend(x(i,:),q(i,:,2),hx,1,'N',0.0,'N',0.0);
    [xe,qex(:,3)] = extend(x(i,:),q(i,:,3),hx,1,'N',0.0,'N',0.0);
    [xe,qex(:,4)] = extend(x(i,:),q(i,:,4),hx,1,'N',0.0,'N',0.0);

    % Compute flux
    [dq1] = EulerLF2Dx(qex(2:Nx+1,:), qex(3:Nx+2,:),gamma);
    [dq2] = EulerLF2Dx(qex(1:Nx-1,:), qex(2:Nx+1,:),gamma);

    % Update residual
    dq(i,:,:,:) = -(dq1-dq2)/hx;
end

% Apply monotone scheme in the y-direction
for j=1:Nx
    % Extend data and assign boundary conditions in y-direction
    [ye,qey(:,1)] = extend(y(:,j),q(:,j,1),hy,1,'N',0.0,'N',0.0);
    [ye,qey(:,2)] = extend(y(:,j),q(:,j,2),hy,1,'N',0.0,'N',0.0);
    [ye,qey(:,3)] = extend(y(:,j),q(:,j,3),hy,1,'N',0.0,'N',0.0);
    [ye,qey(:,4)] = extend(y(:,j),q(:,j,4),hy,1,'N',0.0,'N',0.0);

    % Compute flux
    [dq1] = EulerLF2Dy(qey(2:Ny+1,:), qey(3:Ny+2,:),gamma);
    [dq2] = EulerLF2Dy(qey(1:Ny-1,:), qey(2:Ny+1,:),gamma);

```

```
% Update residual
dq(:,j,:) = dq(:,j,:)
             - reshape(dq1,Ny,1,4)
             - reshape(dq2,Ny,1,4))/hy;
end
return
```

The numerical flux, needed to complete the scheme, is slightly more complicated than in the simpler one-dimensional case. In `EulerLF2Dx.m` and `EulerLF2Dy.m` we illustrate the implementation of the Lax–Friedrichs flux along the  $x$ - and  $y$ -directions, respectively. The difference between using a local or a global Lax–Friedrichs flux is determined through the definition of the parameter  $\alpha$ .

**Script 7.12. EulerLF2Dx.m:** Definition of Lax-Friedrichs flux for the two-dimensional Euler equation along  $x$ .

```

function [dq] = EulerLF2Dx(ql,qr, gamma)
% function [dq] = EulerLF2Dx(ql,qr, gamma)
% Purpose: Evaluate Lax Friedrich numerical flux for Euler's equation along x

% Extract states
rl = ql(:,1); rul = ql(:,2); rvl = ql(:,3); El = ql(:,4);
rr = qr(:,1); rur = qr(:,2); rvr = qr(:,3); Er = qr(:,4);

% Compute fluxes for left and right states
ul = rul./rl; vl = rvl./rl;
pl = (gamma-1)*(El - 0.5*rl.* (ul.^2+vl.^2)); cl = sqrt(gamma*pl./rl);
F1l = rul; F2l = rul.*ul + pl; F3l = rul.*vl; F4l = (El + pl).*ul;

ur = rur./rr; vr = rvr./rr;
pr = (gamma-1)*(Er - 0.5*rr.* (ur.^2+vr.^2)); cr = sqrt(gamma*pr./rr);
F1r = rur; F2r = rur.*ur + pr; F3r = rur.*vr; F4r = (Er + pr).*ur;

% Compute dissipation for LF
maxvell = cl + abs(ul); maxvelr = cr + abs(ur);
alpha = max(max(maxvell, maxvelr))); % Global LF
% alpha = max(maxvell, maxvelr); % Local LF

% Compute flux
Nq = size(ql); N = Nq(1); dq = zeros(N,4);
dq(:,1)=(F1l+F1r)/2-alpha/2.*(rr-rl);
dq(:,2)=(F2l+F2r)/2-alpha/2.*(rur-rul);
dq(:,3)=(F3l+F3r)/2-alpha/2.* (rvr-rvl);
dq(:,4)=(F4l+F4r)/2-alpha/2.* (Er-El);
return

```

**Script 7.13. EulerLF2Dy.m:** Definition of Lax-Friedrichs flux for the two-dimensional Euler equation along  $y$ .

```

function [dq] = EulerLF2Dy(ql ,qr ,gamma)
% function [numflux] = EulerLF2Dy(ql ,qr ,gamma)
% Purpose: Evaluate Lax Friedrich numerical flux for Euler's equation along y

% Extract states
rl = ql(:,1); rul = ql(:,2); rvl = ql(:,3); El = ql(:,4);
rr = qr(:,1); rur = qr(:,2); rvr = qr(:,3); Er = qr(:,4);

```

```

% Compute fluxes for left and right states
ul = rul./rl; vl = rvl./rl;
pl = (gamma-1)*(El - 0.5*rl.* (ul.^2+vl.^2)); cl = sqrt(gamma*pl./rl);
F1l = rvl; F2l = rvl.*ul; F3l = rvl.*vl+pl; F4l = (El + pl).*vl;

ur = rur./rr; vr = rvr./rr;
pr = (gamma-1)*(Er - 0.5*rr.* (ur.^2+vr.^2)); cr = sqrt(gamma*pr./rr);
F1r = rvr; F2r = rvr.*ur; F3r = rvr.*vr+pr; F4r = (Er + pr).*vr;

% Compute dissipation for LF
maxvell = cl + abs(vl); maxvelr = cr + abs(vr);
alpha = max(max(max(maxvell, maxvelr))); % Global LF
% alpha = max(maxvell, maxvelr); % Local LF

% Compute flux
Nq = size(ql); N = Nq(1); dq = zeros(N,4);
dq(:,1)=(F1l+F1r)/2-alpha/2.* (rr-rl);
dq(:,2)=(F2l+F2r)/2-alpha/2.* (rur-rul);
dq(:,3)=(F3l+F3r)/2-alpha/2.* (rvr-rvl);
dq(:,4)=(F4l+F4r)/2-alpha/2.* (Er-El);
return

```

---

Similarly, we show in `EulerHLL2Dx.m` and `EulerHLL2Dy.m` the numerical HLL flux along the  $x$ - and  $y$ -directions, respectively.

**Script 7.14. *EulerHLL2Dx.m: Definition of HLL flux for the two-dimensional Euler equation along  $x$ .***

---

```

function [dq] = EulerHLL2Dx(ql,qr,gamma)
% function [dq] = EulerHLL2Dx(ql,qr,gamma)
% Purpose: Evaluate HLL numerical flux for Euler's equation along x

% Extract states
rl = ql(:,1); rul = ql(:,2); rvl = ql(:,3); El = ql(:,4);
rr = qr(:,1); rur = qr(:,2); rvr = qr(:,3); Er = qr(:,4);

% Compute fluxes for left and right states
ul = rul./rl; vl = rvl./rl;
pl = (gamma-1)*(El - 0.5*rl.* (ul.^2+vl.^2)); cl = sqrt(gamma*pl./rl);
F1l = rul; F2l = rul.*ul + pl; F3l = rul.*vl; F4l = (El + pl).*ul;
sl = min(ul,min(ul+cl,ul-cl));

ur = rur./rr; vr = rvr./rr;
pr = (gamma-1)*(Er - 0.5*rr.* (ur.^2+vr.^2)); cr = sqrt(gamma*pr./rr);
F1r = rur; F2r = rur.*ur + pr; F3r = rur.*vr; F4r = (Er + pr).*ur;
sr = max(ur,max(ur+cr,ur-cr));

% Compute Roe average along x and velocity bounds
rs = (sr.*rr - sl.*rl + F1l - F1r)./(sr-sl);
rus = (sr.*rur - sl.*rul + F2l - F2r)./(sr-sl);
rvs = (sr.*rvr - sl.*rvl + F3l - F3r)./(sr-sl);
Es = (sr.*Er - sl.*El + F4l - F4r)./(sr-sl);

us = rus./rs; vs = rvs./rs;
ps = (gamma-1)*(Es - 0.5*rs.* (us.^2 + vs.^2)); cs = sqrt(gamma*ps./rs);

ssmi = min(us,min(us+cs,us-cs)); ssma = max(us,max(us+cs,us-cs));
sm = min(sl,ssmi); sp = max(sr,ssma);

% Compute flux

```

```

Nq = size (ql) ; N = Nq(1) ; dq = zeros (N,4) ;
q1 = (sm>0); q2 = (sp>=0).* (sm<=0); q3 = (sp<0);
fs = (sp.*F1l - sm.*F1r + sm.*sp.* (rr-r1))./(sp-sm);
dq(:,1) = q1.*F1l + q2.*fs + q3.*F1r;
fs = (sp.*F2l - sm.*F2r + sm.*sp.* (rur-rul))./(sp-sm);
dq(:,2) = q1.*F2l + q2.*fs + q3.*F2r;
fs = (sp.*F3l - sm.*F3r + sm.*sp.* (rvr-rvl))./(sp-sm);
dq(:,3) = q1.*F3l + q2.*fs + q3.*F3r;
fs = (sp.*F4l - sm.*F4r + sm.*sp.* (Er-El))./(sp-sm);
dq(:,4) = q1.*F4l + q2.*fs + q3.*F4r;
return

```

---

**Script 7.15. EulerHLL2Dy.m: Definition of HLL flux for the two-dimensional Euler equation along  $y$ .**

---

```

function [dq] = EulerHLL2Dy(ql,qr,gamma)
% function [dq] = EulerHLL2Dy(ql,qr,gamma)
% Purpose: Evaluate HLL numerical flux for Euler's equation along y

% Extract states
rl = ql(:,1); rul = ql(:,2); rvl = ql(:,3); El = ql(:,4);
rr = qr(:,1); rur = qr(:,2); rvr = qr(:,3); Er = qr(:,4);

% Compute fluxes for left and right states
ul = rul./rl; vl = rvl./rl;
pl = (gamma-1)*(El - 0.5*rl.* (ul.^2+vl.^2)); cl = sqrt(gamma*pl./rl);
F1l = rvl; F2l = rvl.*ul; F3l = rvl.*vl+pl; F4l = (El + pl).*vl;
sl = min(vl,min(vl+cl, vl-cl));

ur = rur./rr; vr = rvr./rr;
pr = (gamma-1)*(Er - 0.5*rr.* (ur.^2+vr.^2)); cr = sqrt(gamma*pr./rr);
F1r = rvr; F2r = rvr.*ur; F3r = rvr.*vr+pr; F4r = (Er + pr).*vr;
sr = max(vr,max(vr+cr, vr-cr));

% Compute Roe average along x and velocity bounds
rs = (sr.*rr - sl.*rl + F1l - F1r)./(sr-sl);
rus = (sr.*rur - sl.*rul + F2l - F2r)./(sr-sl);
rvs = (sr.*rvr - sl.*rvl + F3l - F3r)./(sr-sl);
Es = (sr.*Er - sl.*El + F4l - F4r)./(sr-sl);

us = rus./rs; vs = rvs./rs;
ps = (gamma-1)*(Es - 0.5*rs.* (us.^2+vs.^2)); cs = sqrt(gamma*ps./rs);

ssmi = min(vs,min(vs+cs,vs-cs)); ssma = max(vs,max(vs+cs,vs-cs));
sm = min(sl,ssmi); sp = max(sr,ssma);

% Compute flux
Nq = size(ql); N = Nq(1); dq = zeros(N,4);
q1 = (sm>0); q2 = (sp>=0).* (sm<=0); q3 = (sp<0);
fs = (sp.*F1l - sm.*F1r + sm.*sp.* (rr-r1))./(sp-sm);
dq(:,1) = q1.*F1l + q2.*fs + q3.*F1r;
fs = (sp.*F2l - sm.*F2r + sm.*sp.* (rur-rul))./(sp-sm);
dq(:,2) = q1.*F2l + q2.*fs + q3.*F2r;
fs = (sp.*F3l - sm.*F3r + sm.*sp.* (rvr-rvl))./(sp-sm);
dq(:,3) = q1.*F3l + q2.*fs + q3.*F3r;
fs = (sp.*F4l - sm.*F4r + sm.*sp.* (Er-El))./(sp-sm);
dq(:,4) = q1.*F4l + q2.*fs + q3.*F4r;
return

```

---

As a first evaluation of the performance of the scheme, we consider an isentropic vortex, defined in subsection 1.4.2. We recall that this is an exact smooth solution to the compressible Euler equations [13], comprising a vortex, initially centered at  $(x_0, y_0)$ , and convected at the constant speed  $(u_0, v_0)$ . The implementation is shown in `IsentropicVortex2D.m`.

**Script 7.16.** *IsentropicVortex2D.m: Definition of the exact isentropic vortex problem for the Euler equations.*

---

```
function [r,ru,rv,E] = IsentropicVortex2D(x,x0,u0,y,y0,v0,gamma,beta,t)
% function [r,ru,rv,E] = IsentropicVortex2D(x,x0,u0,y,y0,v0,gamma,beta,t)
% Purpose: compute flow configuration given by
%          Y.C. Zhou, G.W. Wei / Journal of Computational Physics 189 (2003) 159

xs = x-u0*t-x0; ys = y-v0*t-y0; rr = sqrt(xs.^2 + ys.^2);
u = u0 - beta*exp(1-rr.^2).*ys/(2*pi); v = v0 + beta*exp(1-rr.^2).*xs/(2*pi);
r = (1 - ((gamma-1)*beta^2*exp(2*(1-rr.^2))/(16*gamma*pi.^2))).^(1/(gamma-1));
ru = r.*u; rv = r.*v; E = r.^gamma/(gamma-1)+0.5*r.* (u.^2+v.^2);
return
```

---

To evaluate the accuracy of the scheme, we take  $\beta = 5$ ,  $(x_0, y_0) = (0, 0)$ , and  $(u_0, v_0) = (1, 0)$ .

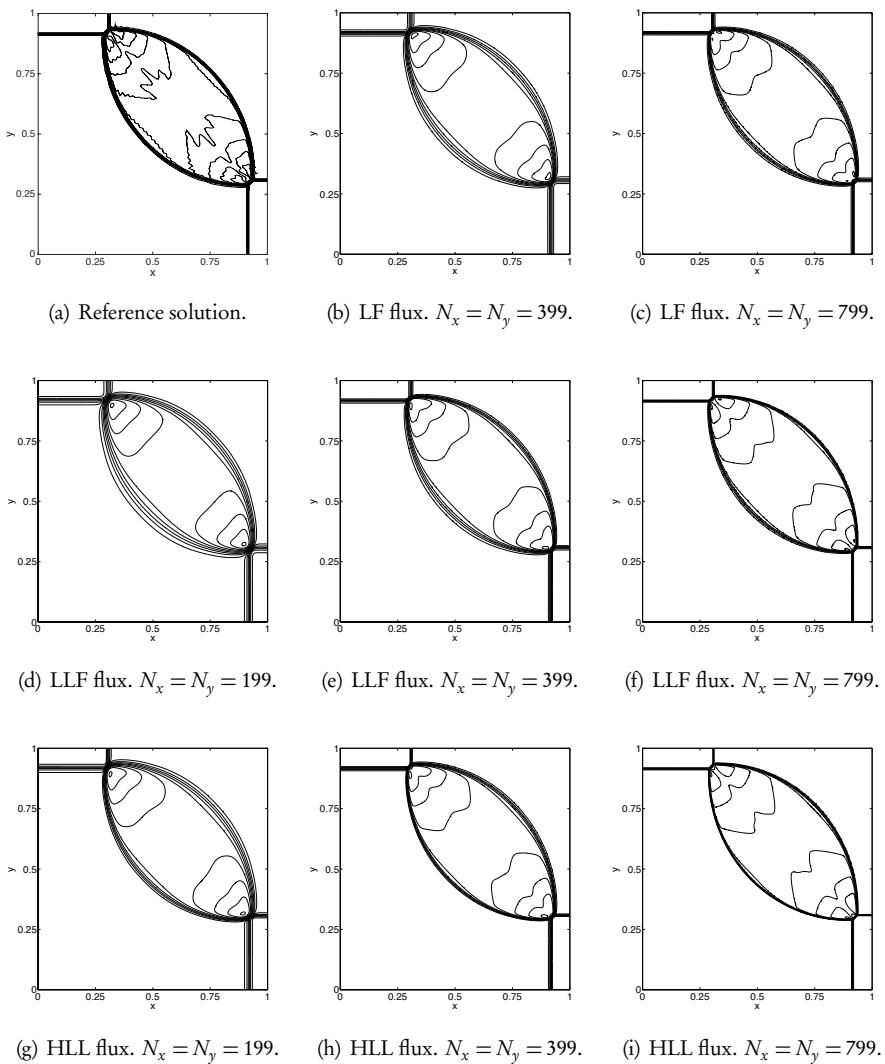
In Table 7.1 we list the errors, measured in  $\|\cdot\|_{h,1}$ , of some of the conserved variables at  $T = 1$ , computed with different fluxes. As expected for a smooth case, we observe  $\mathcal{O}(h)$  convergence for all three fluxes. Furthermore, we see a clear advantage of using the advanced HLL flux, which yields a more accurate result than what is obtained with the global Lax-Friedrichs flux. Indeed, the result obtained with the HLL flux with  $N_x = N_y = 399$  has comparable accuracy to the result obtained with the global Lax-Friedrichs flux with  $N_x = N_y = 799$ .

To further evaluate the quality of the methods and, in particular, the importance of the choice of the numerical flux, we consider the two-dimensional version of the shock tube problems discussed in subsection 1.4.2.

We first consider the fourth test case, comprising four interacting shocks. Figure 7.3 shows the computed density  $\rho$  at a final time  $T = 0.25$ , obtained with different resolutions and choice of fluxes. While the exact solution is unknown, the results indicate convergence to the reference solution. It is also clear that increasing resolution and/or choosing a more advanced flux results in improved details and sharpening of the shock fronts.

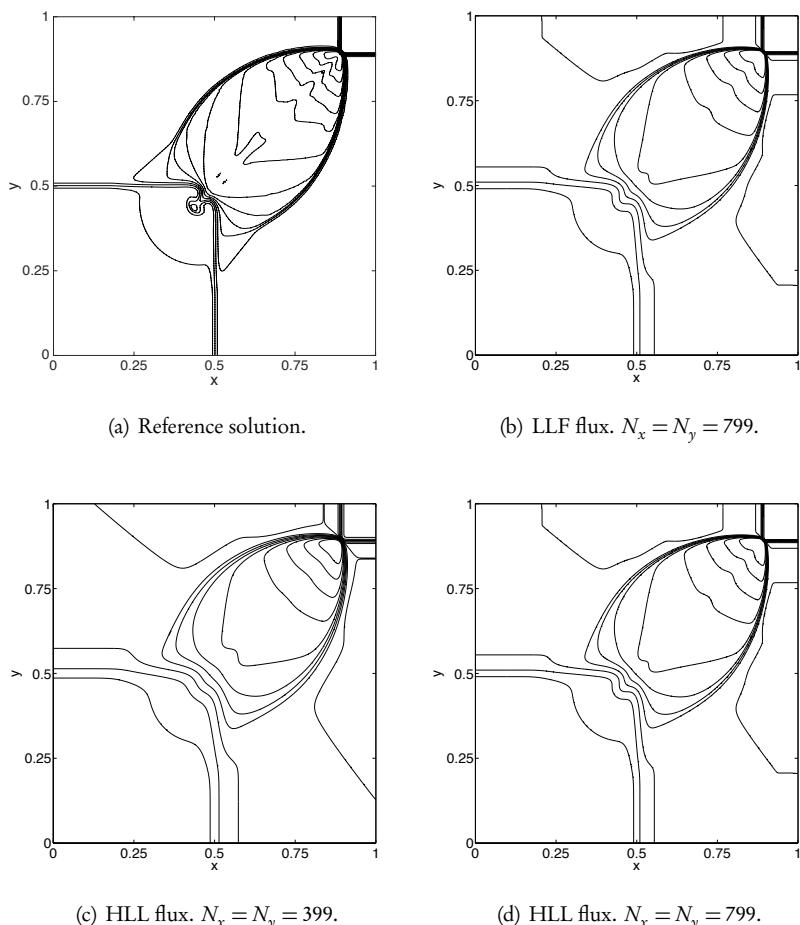
**Table 7.1.** *Errors for the isentropic vortex test of the two-dimensional Euler equations, solved with three different numerical fluxes. The columns show the error, measured in  $\|\cdot\|_{h,1}$ , of conserved variables at  $T = 1$  for the three fluxes.*

$N_x = N_y$	Global LF		Local LF		HLL	
	$\rho$	$\rho u$	$\rho$	$\rho u$	$\rho$	$\rho u$
199	4.08e-1	8.03e-1	3.21e-1	6.20e-1	2.27e-1	4.75e-1
399	2.31e-1	4.47e-1	1.75e-1	3.32e-1	1.20e-1	2.49e-1
799	1.24e-1	2.37e-1	9.19e-2	1.73e-1	6.21e-2	9.88e-2
1599	6.47e-2	1.23e-1	4.72e-2	8.82e-2	3.16e-2	6.45e-2



**Figure 7.3.** Computed density of Riemann problem number 4 (see [6, 4]) comprising four shocks. All figures show the density at  $T = 0.25$ . The upper-left figure represents a high resolution reference solution. The remaining figures in the first row show the density obtained with a global Lax-Friedrichs (LF) flux, the second row shows the density obtained with the local Lax-Friedrichs flux (LLF), and the last row shows the density obtained using an HLL flux.

As a more challenging second case, we consider test case 12, which features two shocks and two contact waves. We recall the results for Sod's problem that showed significant smearing of contact waves. The results in Fig. 7.4 show similar behavior, yielding a poor resolution of the contact waves, even for the HLL flux. As we have discussed previously, this is associated with the nature of the contact wave, behaving as a linear wave where the effect of dissipation accumulates. The only way to truly address this is to pursue the development of higher-order schemes.



**Figure 7.4.** Computed density for Riemann problem number 12 (see [6, 4]) comprising two shocks and two contact waves. All figures show the density at  $T = 0.2$ . The first row contains the reference solution and the result obtained with a local Lax-Friedrichs (LLF) flux, while the second row shows the results using an HLL flux.

## References

- [1] Timothy Barth and Mario Ohlberger. *Finite Volume Methods: Foundation and Analysis*. Encyclopedia of Computational Mechanics. John Wiley, 2004.
- [2] Phillip Colella. Multidimensional upwind methods for hyperbolic conservation laws. *Journal of Computational Physics*, 87(1):171–200, 1990.
- [3] Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. Finite volume methods. *Handbook of Numerical Analysis*, 7:713–1018, 2000.

- [4] Alexander Kurganov and Eitan Tadmor. Solution of two-dimensional Riemann problems for gas dynamics without Riemann problem solvers. *Numerical Methods for Partial Differential Equations*, 18(5):584–608, 2002.
- [5] Jan Olav Langseth and Randall J. LeVeque. A wave propagation method for three-dimensional hyperbolic conservation laws. *Journal of Computational Physics*, 165(1):126–166, 2000.
- [6] Peter D. Lax and Xu-Dong Liu. Solution of two-dimensional Riemann problems of gas dynamics by positive schemes. *SIAM Journal on Scientific Computing*, 19(2):319–340, 1998.
- [7] Randall J. LeVeque. Wave propagation algorithms for multidimensional hyperbolic systems. *Journal of Computational Physics*, 131(2):327–353, 1997.
- [8] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*, volume 31 of Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002.
- [9] David Sidilkover. *A Genuinely Multidimensional Upwind Scheme and an Efficient Multigrid for the Compressible Euler Equations*. ICASE Report, 1994.
- [10] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer Science+Business Media, 2009.
- [11] John A. Trangenstein. *Numerical Solution of Hyperbolic Partial Differential Equations*. Cambridge University Press, 2009.
- [12] Henk K. Versteeg and Weeratunge Malalasekera. *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education, 2007.
- [13] Yongcheng Zhou and Guo-Wei Wei. High resolution conjugate filters for the simulation of flows. *Journal of Computational Physics*, 189(1):159–179, 2003.

## Chapter 8

# High-order accuracy and its challenges

Whereas the development of monotone schemes has led us to a powerful, flexible, and robust family of methods, the discussion also revealed some clear limitations. Perhaps the most critical of these is the restriction to first order accuracy and the impact this has on the resolution of the solution, e.g., smeared contact waves even when a very fine spatial grid is used.

In this third and most extensive part of the text, we pursue the development of high-order accurate schemes for conservation laws. Ultimately, the motivation for this is to do more with less, i.e., to develop schemes that are more accurate than first order accurate schemes without substantially increasing the computational cost. As simple as this may appear, we have already experienced some of the challenges lying ahead. In Ex. 4.4 we observed that the second order accurate Lax–Wendroff scheme was superior in terms of the steepness of the shock but resulted in the introduction of artificial oscillations upstream of the shock. A similar observation was made in subsection 5.3.5 when solving the Euler equations.

The discussion in Chapter 4 highlighted that to achieve high-order accuracy we must abandon the notion of monotone schemes, and seek a different approach to control oscillations and ensure stability. Furthermore, Godunov’s theorem implies that we cannot hope to develop high-order accurate linear schemes for conservation laws. Hence, we must seek more advanced nonlinear, i.e., solution dependent, schemes.

In this chapter we discuss in more detail the advantages of high-order accurate methods when solving conservation laws, and we seek to shed some light on the reasons for some of the challenges we have already observed. This sets the stage for a more general development of high-order accurate schemes in the remainder of the text.

### 8.1 • The good

To quantify the value of high-order accurate schemes for conservation laws, let us focus on the linear problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0, \quad x \in [0, 2\pi],$$

with periodic boundary conditions and an initial condition  $u_0(x)$ . We recall that the exact solution is  $u(x, t) = u_0(x - at)$ , with phase speed  $a$  being real.

Let us introduce the grid  $x_j = jh$ ,  $j = 0 \dots N$ , and  $h = \frac{2\pi}{N+1}$ , and assume that the solution in the neighborhood of each grid point  $x_j$  can be represented by a polynomial  $u_b$  as

$$u(x, t) \simeq u_b(x, t) = \sum_{m=-p}^q u_{j+m}(t) \ell_{j+m}(x), \quad x \in [x_{j-p}, x_{j+q}].$$

Here  $\ell_{j+m}(x)$  is the interpolating Lagrange polynomial of order  $p+q$ , defined as

$$\ell_{j+k}(x) = \prod_{\substack{l=-p \\ l \neq k}}^q \frac{x - x_{j+l}}{x_{j+k} - x_{j+l}},$$

and  $u_{j+m}(t) = u(x_{j+m}, t)$ . We can now approximate derivatives of  $u$  at  $x_j$  by evaluating the derivatives of  $u_b$  at  $x_j$ . Depending on the choices of  $p$  and  $q$  we recover a number of well-known approximations:

$$\left. \frac{\partial u}{\partial x} \right|_{x_j} \simeq \begin{cases} D_b^- u_j, & p = 1, q = 0, \\ D_b^+ u_j, & p = 0, q = 1, \\ D_b^0 u_j, & q = p = 1, \\ D_b^0 (I - \frac{b^2}{6} D_b^+ D_b^-) u_j, & q = p = 2, \\ D_b^0 (I - \frac{b^2}{6} D_b^+ D_b^- + \frac{b^4}{30} (D_b^+)^2 (D_b^-)^2) u_j, & q = p = 3, \end{cases} \quad (8.1)$$

where we have borrowed the notation from Chapter 5 to represent the stencils and  $u_j = u_j(t) = u_b(x_j, t)$ . We recognize the first three stencil schemes, discussed in detail in Chapter 5, as standard finite difference stencils of first and second order accuracy. The last two schemes are central difference schemes of fourth and sixth order accuracy, respectively.

To gain insight into the accuracy of these approximations, we recall the error term associated with Lagrange interpolation [3]:

$$u(x, t) - u_b(x, t) = \frac{u^{(p+q+1)}(\xi, t)}{(p+q+1)!} \prod_{m=-p}^q (x - x_{j+m}), \quad \xi \in [x_{j-p}, x_{j+q}].$$

Differentiating this and evaluating it at  $x_j$  yields

$$\left. \left( \frac{\partial u}{\partial x} - \frac{\partial u_b}{\partial x} \right) \right|_{x_j} = (-1)^q \frac{p! q! u^{(p+q+1)}(\xi, t)}{(p+q+1)!} h^{p+q}, \quad (8.2)$$

which confirm our expectations in terms of accuracy.

Constructing the Lagrange polynomials and differentiating these directly can be utilized to derive schemes of any order. If we, for simplicity, consider central schemes with  $m = p = q$ , straightforward algebra shows

$$\left. \frac{\partial u_b}{\partial x} \right|_{x_j} = \sum_{n=1}^m \alpha_n^m \frac{E_n - E_{-n}}{2nh} u_j, \quad \alpha_n^m = -2(-1)^n \frac{(m!)^2}{(m-n)!(m+n)!}, \quad (8.3)$$

which results in schemes of  $\mathcal{O}(h^{2m})$  accuracy. Similarly, one can derive general one-sided stencils or stencils on nonuniform grids. An elegant algorithm for computing the corresponding stencil weights for any situation is given in [7, 8].

Before we develop a deeper understanding of the properties of these high-order schemes, let us consider an example to gain some intuition.

**Example 8.1.** We consider the linear wave problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0, \quad x \in [0, 1],$$

with periodic boundary conditions and the initial condition

$$u_0(x) = \exp(\sin(2\pi x)).$$

We solve the problem using a standard finite difference method in the form

$$\frac{\partial u_h}{\partial t} \Big|_{x_j} + a \frac{\partial u_h}{\partial x} \Big|_{x_j} = 0,$$

and consider the different approximations to the spatial derivative given in (8.1). A stable time integration method is chosen to ensure that all errors from the temporal integration can be neglected.

In Fig. 8.1 we show the solution for  $a = 1$  computed with  $N = 128$  grid points at three different times with three different schemes. We first of all observe that the upwind scheme quickly dissipates the variation of the solution and reaches a constant solution which reflects mass conservation

$$\int_0^1 \exp(\sin(2\pi x)) dx = 1.26606587\dots$$

A comparison with the results obtained with the second order central scheme reveals two distinct differences. A first one is that the solution, obtained with the second order accurate scheme, remains accurate for a substantially longer time. Furthermore, as the wave begins to disintegrate, it does so in a way that is qualitatively different from a dissipative scheme. Finally, for the fourth order scheme the accuracy is excellent, even after 100 periods of propagation, confirming improved accuracy. To confirm that there is nothing special about the fourth order scheme, we continue the computations to  $T = 1000$  and  $T = 10000$  in Fig. 8.2 and observe that, eventually, the behavior is qualitatively similar to that obtained with the second order scheme.

The qualitative differences in the long time behavior of the solution can be understood by considering the modified equation [20], i.e., we seek a smooth solution,  $v(x, t)$ , which satisfies the numerical scheme exactly. Neglecting temporal errors, we obtain

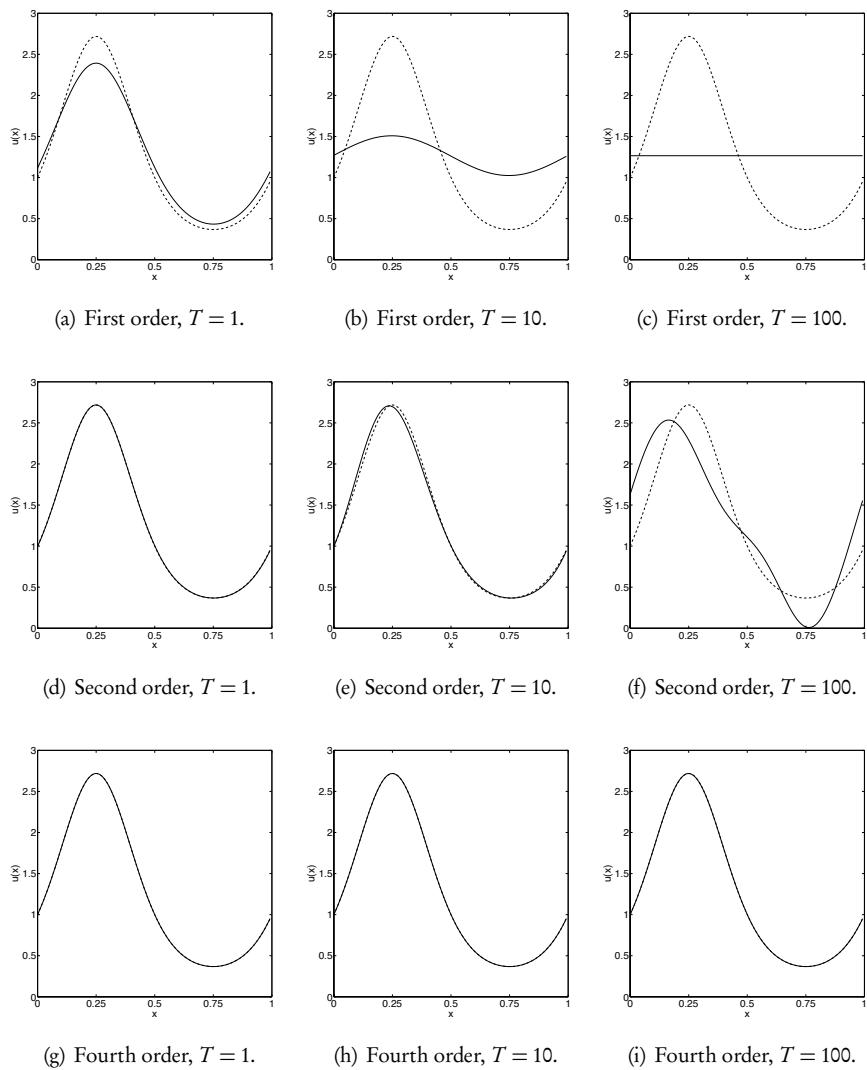
$$\frac{\partial v}{\partial t} + a \left( \frac{\partial v}{\partial x} - R(x, t) \right) = 0,$$

where

$$R(x, t) = \frac{1}{(p+q+1)!} \frac{\partial}{\partial x} \left( \prod_{m=-p}^q (x - x_{j+m}) v^{(p+q+1)}(\xi, t) \right)$$

results from the error term of the Lagrange interpolation. From (8.2), we obtain the remainder  $R(x, t)$  at the grid points

$$R(x_j, t) = \frac{(-1)^q p! q!}{(p+q+1)!} h^{p+q} v^{(p+q+1)}(\xi, t).$$

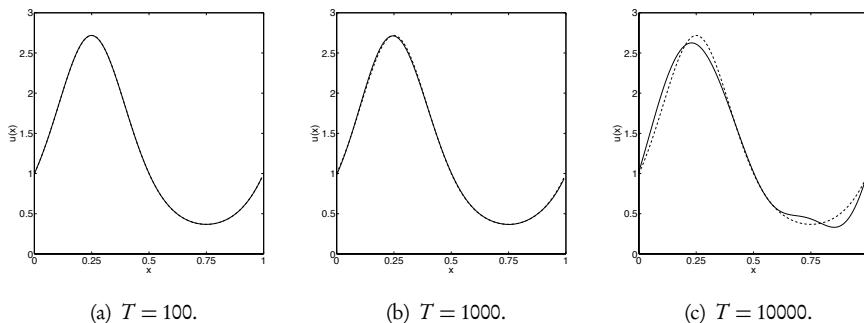


**Figure 8.1.** Solution of the linear wave equation using different finite difference schemes. The solid lines represent the computed solution and the dashed lines is the exact solution. All results are computed with  $N = 128$  points until final time  $T$ .

This allows us to recover the modified equation for different choices of  $p$  and  $q$ . For upwinding ( $p = 1, q = 0$ ) we obtain the modified equation to leading order (see also Ex. 5.1)

$$\frac{\partial v}{\partial t} + \alpha \frac{\partial v}{\partial x} = \alpha \frac{h}{2} v^{(2)}, \quad (8.4)$$

which highlights the dissipation inherent in the scheme, as well as its first order accuracy.



**Figure 8.2.** Solution of the linear wave equation using a fourth order finite difference scheme with  $N = 128$  points. The solid lines represent the computed solution and the dashed lines the exact solution.

For the downwinding scheme ( $p = 0, q = 1$ ) we obtain

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = -a \frac{h}{2} v^{(2)},$$

which reflects the unstable nature of the scheme, since the modified equation is ill-posed for  $a > 0$ .

For the second order accurate central scheme we recover

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = -a \frac{h^2}{6} v^{(3)}, \quad (8.5)$$

while the modified equation for the fourth order central scheme is

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = a \frac{h^4}{30} v^{(5)}. \quad (8.6)$$

These latter two modified equations are dispersive in nature, resulting in a wave speed that depends on the frequency of the wave [35]. This is exactly the type of behavior observed in Figs. 8.1 and 8.2.

While it is expected that the fourth order scheme is superior in accuracy when compared to the low-order schemes, it is only fair to include the cost of the schemes in the assessment since the fourth order scheme uses twice as many points in the evaluation of the derivative as the second order scheme. To assess the accuracy of the schemes at a given resolution, we show the errors of the schemes, measured in different norms, in Table 8.1.

An inspection of the results shows that comparable accuracy is achieved for  $N = 512$  with the first order scheme,  $N = 128$  for the second order central scheme, and  $N = 16$  when using the fourth order central scheme. This is clearly much more than a factor of two in cost reduction.

It is worth observing that if we consider the extreme case of  $p + q = N$ , i.e., we are using all available information to evaluate the derivative, we could, under appropriate assumptions on smoothness, expect an error

$$h^N \simeq N^{-N} \simeq \frac{1}{N!} e^{-N},$$

**Table 8.1.** Errors for the linear wave problem, using different finite difference schemes for increasing resolution  $N$ . The errors are measured at  $T = \sqrt{2}$  in  $\|\varepsilon\|_{h,1}$  and  $\|\varepsilon\|_{h,2}$ , where  $\varepsilon = u - u_b$ . The last row represents the approximate rate of convergence.

$N$	First order upwind		Second order central		Fourth order central	
	$\ \varepsilon\ _{h,1}$	$\ \varepsilon\ _{h,2}$	$\ \varepsilon\ _{h,1}$	$\ \varepsilon\ _{h,2}$	$\ \varepsilon\ _{h,1}$	$\ \varepsilon\ _{h,2}$
16	6.09e-1	6.89e-1	2.94e-1	3.48e-1	4.54e-2	5.82e-2
32	4.43e-1	5.02e-1	7.92e-2	1.07e-1	3.12e-3	4.51e-3
64	2.82e-1	3.25e-1	1.97e-2	2.76e-2	2.01e-4	2.90e-4
128	1.65e-1	1.94e-1	4.94e-3	6.93e-3	1.26e-5	1.83e-5
256	9.06e-2	1.09e-1	1.24e-3	1.73e-3	7.89e-7	1.15e-6
512	4.78e-2	5.80e-2	3.09e-4	4.33e-4	4.94e-8	7.16e-8
1024	2.46e-2	3.01e-2	7.72e-5	1.08e-4	3.08e-9	4.47e-9
2048	1.25e-2	1.53e-2	1.93e-5	2.71e-5	1.93e-10	2.82e-10
4096	6.28e-3	7.73e-3	4.83e-6	6.77e-6	1.94e-11	2.87e-11
	$\mathcal{O}(h)$	$\mathcal{O}(h)$	$\mathcal{O}(h^2)$	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^4)$

which suggests the potential for exponential accuracy. This limiting case of the finite difference schemes has been discussed extensively in [9], and we return to this in much more detail in Chapter 13.

As a final summary, we expect to find a preference for high-order methods when high accuracy is needed, when moderate accuracy is needed with a minimal number of grid points, or for problems which require long time integration. These are characteristics that are shared by many applications, described by conservation laws. ■

### 8.1.1 • Phase error analysis

With a qualitative understanding of the benefits of high-order accurate methods for solving wave problems, let us attempt to establish these observations in a more rigorous manner. To this, we shall explore phase error analysis, introduced in [16].

Let us employ the setting from the last section and consider the semidiscrete central finite difference approximation of order  $2m$  as

$$\frac{du_j(t)}{dt} + a \sum_{n=1}^m \alpha_n^m \frac{E_n - E_{-n}}{2nh} u_j(t) = 0, \quad (8.7)$$

where  $u_j(t) = u_b(x_j, t)$  is the grid point value at  $x_j$  and  $\alpha_n^m$  is given in (8.3).

To keep things simple, we choose the Fourier mode  $u(x, 0) = \exp(ilx)$  as the initial condition, resulting in the exact solution  $u(x, t) = \exp(il(x - at))$ , where  $l = 2\pi\tilde{\lambda}^{-1}$  is the wave number and  $\tilde{\lambda}$  is the associated wavelength. More complex initial conditions can be understood as a linear combination of Fourier modes.

We now conjecture that the semidiscrete solution to (8.7) takes a form similar to the continuous solution

$$u_b(x, t) = \exp(il(x - \alpha_{2m}(l)t)), \quad (8.8)$$

where we have introduced the numerical phase speed,  $a_{2m}(l)$ . As we shall see shortly, this depends on both  $m$  and  $l$ .

Let us define the phase error:

$$e_{2m}(l) = \left| \frac{u(x, t) - u_b(x, t)}{u(x, t)} \right| = |1 - \exp(il(a - a_{2m}(l))t)| \simeq |l(a - a_{2m}(l))t|.$$

For each order of approximation  $2m$ , the numerical phase speed is recovered by inserting (8.8) into the scheme. For the second order scheme, this yields

$$a_2(l) = a \frac{\sin(lh)}{lh} = a \left( 1 - \frac{(lh)^2}{6} + \dots \right),$$

while the expression for the fourth order scheme is

$$a_4(l) = a \frac{8 \sin(lh) - \sin(2lh)}{6lh} = a \left( 1 - \frac{(lh)^4}{30} + \dots \right).$$

We observe that the order of the scheme is reflected in the accuracy of the numerical phase speed and, by consulting Ex. 8.1, we recognize the terms in the Taylor expansions as the leading terms in the modified equation. For the sixth order central scheme we recover

$$a_6(l) = a \frac{45 \sin(lh) - 9 \sin(2lh) + \sin(3lh)}{30kh} = a \left( 1 - \frac{(lh)^6}{140} + \dots \right).$$

We recover the phase errors to leading order as

$$e_2(l) = lat \frac{(lh)^2}{6}, \quad e_4(l) = lat \frac{(lh)^4}{30}, \quad e_6(l) = lat \frac{(lh)^6}{140}.$$

To reach a measure of accuracy that is problem independent, let us follow [16] and introduce two dimensionless variables,  $p$  and  $\nu$ . Here  $p$  is the number of points per wavelength:

$$p = \frac{\tilde{\lambda}}{h} = \frac{2\pi}{lh} \Rightarrow lh = 2\pi p^{-1}.$$

When  $p$  is large, one expects a high accuracy but also a high cost. In other words, for a fixed value of  $m$ ,  $p$  is a direct measure of cost. We also define the number of periods over which we need to solve the problem:

$$\nu = \frac{at}{\tilde{\lambda}} = \frac{lat}{2\pi}.$$

This is a direct measure of the length of the evolution time. Using these variables, we recover the phase errors as

$$e_2(p, \nu) = \frac{\pi\nu}{3} (2\pi p^{-1})^2, \quad e_4(p, \nu) = \frac{\pi\nu}{15} (2\pi p^{-1})^4, \quad e_6(p, \nu) = \frac{\pi\nu}{70} (2\pi p^{-1})^6.$$

We are now in a position to seek the answer to the key question: given a desirable accuracy  $\varepsilon$ , how does one choose the order  $2m$  of the scheme to minimize  $p$  and, thus,

minimize the computational effort? Requiring that the phase error,  $e_{2m}$ , be less than the tolerance,  $\varepsilon$ , yields

$$p_2(\varepsilon, \nu) \geq 2\pi \sqrt[2]{\frac{\nu\pi}{3\varepsilon}}, \quad p_4(\varepsilon, \nu) \geq 2\pi \sqrt[4]{\frac{\nu\pi}{15\varepsilon}}, \quad p_6(\varepsilon, \nu) \geq 2\pi \sqrt[6]{\frac{\nu\pi}{70\varepsilon}}. \quad (8.9)$$

This reflects that the required number of points per wavelength depends on the tolerance,  $\varepsilon$ , and the required time of evolution,  $\nu$ .

**Example 8.2.** With the estimates for the number of points per wavelength in (8.9) it is illustrative to consider a few specific cases with different levels of required accuracy:

$\varepsilon = 0.1$ :

$$p_2(\nu) \geq 20\sqrt[2]{\nu}, \quad p_4(\nu) \geq 7\sqrt[4]{\nu}, \quad p_6(\nu) \geq 6\sqrt[6]{\nu},$$

$\varepsilon = 0.01$ :

$$p_2(\nu) \geq 64\sqrt[2]{\nu}, \quad p_4(\nu) \geq 13\sqrt[4]{\nu}, \quad p_6(\nu) \geq 8\sqrt[6]{\nu},$$

$\varepsilon = 0.00001$ :

$$p_2(\nu) \geq 643\sqrt[2]{\nu}, \quad p_4(\nu) \geq 43\sqrt[4]{\nu}, \quad p_6(\nu) \geq 26\sqrt[6]{\nu}.$$

As expected, we see that to ensure high accuracy, only a high-order scheme is practical. Recalling that the above expressions assume one spatial dimension, it is clear that the advantages of a higher-order scheme is expected to be even more profound for problems in multiple spatial dimensions. However, even for moderate accuracy, higher-order accurate methods may well be advantageous unless the time of integration is short.

To highlight the importance of high-order methods for long time integration, consider

$\nu = 1$ :

$$p_2(\varepsilon) \geq 6\sqrt[2]{\varepsilon^{-1}}, \quad p_4(\varepsilon) \geq 3\sqrt[4]{\varepsilon^{-1}}, \quad p_6(\varepsilon) \geq 1\sqrt[6]{\varepsilon^{-1}},$$

$\nu = 10$ :

$$p_2(\varepsilon) \geq 20\sqrt[2]{\varepsilon^{-1}}, \quad p_4(\varepsilon) \geq 5\sqrt[4]{\varepsilon^{-1}}, \quad p_6(\varepsilon) \geq 2\sqrt[6]{\varepsilon^{-1}},$$

$\nu = 100$ :

$$p_2(\varepsilon) \geq 64\sqrt[2]{\varepsilon^{-1}}, \quad p_4(\varepsilon) \geq 29\sqrt[4]{\varepsilon^{-1}}, \quad p_6(\varepsilon) \geq 13\sqrt[6]{\varepsilon^{-1}},$$

$\nu = 1000$ :

$$p_2(\varepsilon) \geq 202\sqrt[2]{\varepsilon^{-1}}, \quad p_4(\varepsilon) \geq 51\sqrt[4]{\varepsilon^{-1}}, \quad p_6(\varepsilon) \geq 19\sqrt[6]{\varepsilon^{-1}}.$$

This clearly indicates that for long time integration, high-order methods are the only practical approach, in agreement with the observations in Ex. 8.1.  $\blacksquare$

For a general order of the scheme, we recover an expression for the required points per wavelength as

$$p_{2m}(\varepsilon, \nu) \geq 2\pi \sqrt[2m]{\frac{2\pi(m!)^2}{(2m+1)!} \frac{\nu}{\varepsilon}}.$$

Using the Legendre's duplication formula for the Gamma function, one obtains

$$\lim_{m \rightarrow \infty} p_{2m}(\varepsilon, \nu) = \pi.$$

This has the interesting consequence that for very high order schemes, the accuracy requirement becomes independent of the integration time. This is exactly what happens for global Fourier spectral methods where a more detailed analysis shows that the limit of  $p_{2m}$  is two—the minimum number of points required to uniquely represent a harmonic wave. We refer to [15] for a discussion of this aspect and return to this case again in Chapter 13.

Let us finally consider how these results translate into estimates of work. We use as an estimate

$$\begin{aligned} W_{2m}(\nu, \varepsilon) &= 2m p_{2m} K = 2m p_{2m} \frac{\nu}{khCFL_{2m}} \\ &= 2m \frac{p_{2m}^2}{CFL_{2m}} \nu = \frac{2m\nu}{CFL_{2m}} \sqrt[m]{2\pi \frac{(m!)^2}{(2m+1)!} \frac{\nu}{\varepsilon}}, \end{aligned}$$

where  $K$  is the total number of time steps and  $CFL_{2m}$  is the CFL number used. One can assume this to be  $\mathcal{O}(1)$ . For the schemes we have discussed previously, the estimates become

$$W_2(\nu, \varepsilon) \simeq 2\nu \frac{\nu}{\varepsilon}, \quad W_4(\nu, \varepsilon) \simeq 2\nu \sqrt{\frac{\nu}{\varepsilon}}, \quad W_6(\nu, \varepsilon) \simeq 3\nu \sqrt[3]{\frac{\nu}{\varepsilon}}.$$

For large values of  $m$ , i.e., very high order schemes, we recover

$$W_{2m}(\varepsilon, \nu) \simeq mv,$$

which reflects a simple linear dependence on the order of the scheme and the length of the integration.

When one considers wave dominated problems, high-order accurate methods are desirable, and perhaps even essential, when high accuracy is required or for problems where the time of interest is large. Furthermore, even for moderate accuracy, the substantial reduction of the required total number of grid points, enabled by a high-order scheme, is often necessary to control memory usage, in particular for multidimensional problems. A final, and perhaps less obvious point, is that for problems with many scales, i.e., where the largest important wave number  $l$  can be very large, the benefits of high-order methods are obvious.

## 8.2 • The bad

The phase error analysis provides a powerful argument for the development of high-order accurate methods for conservation laws. However, the semidiscrete analysis of the linear problem only exposes certain elements of what is needed of a computational scheme, suitable to solve general systems of conservation laws. Most importantly, the discussion does not address the stability of the schemes, in particular in the case of nonlinear problems.

In Chapter 4 we developed monotone schemes which guaranteed stability under light conditions, i.e., the CFL condition. Unfortunately, we also concluded that monotone schemes are limited to first order accuracy. In the following we develop different notions of stability to enable the pursuit of genuinely high-order accurate stable schemes.

### 8.2.1 • Total variation stability

If we first restrict ourselves to the scalar conservation law endowed with a convex flux, we recall that Oleinik's entropy condition, (2.11), ensures that the entropy solution to the conservation law is total variation diminishing, i.e.,

$$TV(u(t)) \leq TV(u(0)), \quad TV(u(t)) = \sup_{\varepsilon} \int_{\Omega_x} \left| \frac{u(x + \varepsilon, t) - u(x, t)}{\varepsilon} \right| dx,$$

assuming that the solution is compactly supported on  $\Omega_x$ .

Let us define the total variation over  $[0, T]$  as

$$\begin{aligned} TV_T(u) &= \limsup_{\varepsilon \rightarrow 0} \int_0^T \int_{\Omega_x} \left| \frac{u(x + \varepsilon, t) - u(x, t)}{\varepsilon} \right| dx dt \\ &\quad + \limsup_{\varepsilon \rightarrow 0} \int_0^T \int_{\Omega_x} \left| \frac{u(x, t + \varepsilon) - u(x, t)}{\varepsilon} \right| dx dt. \end{aligned}$$

Turning to the discrete solution  $U_j^n$  of the conservation law we recover that

$$\begin{aligned} TV_T(U) &= \sum_j \sum_n \left[ k|U_{j+1}^n - U_j^n| + b|U_j^{n+1} - U_j^n| \right] \\ &= \sum_n \left[ k\|U^n\|_{TV} + \|U^{n+1} - U^n\|_{b,1} \right], \end{aligned}$$

where we have introduced the discrete total variation as

$$\|U^n\|_{TV} = \sum_j |U_{j+1}^n - U_j^n| = \sum_j |\Delta^+ U_j^n|.$$

We call the scheme total variation stable, or TV-stable, if  $TV_T(U) \leq R$  for some constant,  $R$ . We recall the following result [20].

**Theorem 8.3.** *Consider a conservative scheme with a Lipschitz-continuous numerical flux. Assume that the total variation is bounded, i.e., there exists  $k_0, R > 0$  such that*

$$\|U^n\|_{TV} \leq R,$$

*for all  $(n, k)$  with  $k \leq k_0$ . Then the method is TV-stable.*

**Proof.** We consider a conservative scheme

$$U_j^{n+1} = U_j^n - \frac{k}{b} \left[ F_{j+1/2}^n - F_{j-1/2}^n \right], \quad F_{j+1/2}^n = F(U_{j-p}^n, \dots, U_{j+q}^n).$$

Hence

$$\|U^{n+1} - U^n\|_{b,1} = k \sum_j |F_{j+1/2}^n - F_{j-1/2}^n|.$$

Since the numerical flux is Lipschitz continuous, we have

$$|F_{j+1/2}^n - F_{j-1/2}^n| \leq M \max_{-p \leq l \leq q} |\Delta^- U_{j+l}^n| \leq M \sum_{l=-p}^q |\Delta^- U_{j+l}^n|,$$

where  $M$  is the Lipschitz constant. Furthermore, since  $U^n$  is compactly supported,  $|U_j^n| \leq R/2$  by the assumed bound on  $\|U^n\|_{TV}$  and we recover

$$\|U^{n+1} - U^n\|_{b,1} \leq kM \sum_{l=-p}^q \sum_j |\Delta^- U_{j+l}^n| \leq kM \sum_{l=-p}^q \|U^n\|_{TV} \leq kM(p+q+1)R = \alpha k.$$

For the total variation, we recover

$$TV_T(U) = \sum_n [k\|U^n\|_{TV} + \|U^{n+1} - U^n\|_{b,1}] \leq \sum_n [kR + \alpha k] \leq (\alpha + R)T,$$

since  $n = 0, \dots, T/k$ .  $\square$

Hence,  $\|U^n\|_{TV} \leq R$  implies TV-stability, which itself suffices to guarantee convergence for a consistent scheme. Since we employ a conservative scheme, the solution must be a weak solution to the conservation law.

Theorem 4.5 states that a monotone scheme is also total variation diminishing (TVD). Furthermore, Theorem 4.8 shows that, if a scheme is TVD, it is also monotonicity-preserving, i.e., artificial oscillations cannot be generated. This also holds for the slightly weaker condition of total variation bounded.

On the other hand, we recall that for linear schemes, monotonicity-preserving schemes are also monotone, which limits the accuracy to first order. The development of high-order methods must therefore involve nonlinear, i.e., solution-dependent, schemes.

Returning to the scalar conservation law, we now seek schemes that are TVD. A seminal result is obtained in [14].

**Theorem 8.4.** *Consider a scheme for the periodic scalar conservation law in the form*

$$U_j^{n+1} = U_j^n - C_{j-1/2} \Delta^- U_j^n + D_{j+1/2} \Delta^+ U_j^n,$$

where  $C_{j-1/2}$  and  $D_{j+1/2}$  are coefficients that may depend on  $U_j^n$ .

If

$$C_{j+1/2} \geq 0, \quad D_{j+1/2} \geq 0, \quad C_{j+1/2} + D_{j+1/2} \leq 1,$$

the scheme is TVD.

**Proof.** Consider the scheme

$$U_j^{n+1} = U_j^n - C_{j-1/2} \Delta^+ U_{j-1}^n + D_{j+1/2} \Delta^+ U_j^n,$$

utilizing that  $\Delta^- U_j^n = \Delta^+ U_{j-1}^n$ . We likewise have the shifted scheme

$$U_{j+1}^{n+1} = U_{j+1}^n - C_{j+1/2} \Delta^+ U_j^n + D_{j+3/2} \Delta^+ U_{j+1}^n,$$

and subtract the former from the latter to recover

$$\Delta^+ U_j^{n+1} = C_{j-1/2} \Delta^+ U_{j-1}^n + (1 - C_{j+1/2} - D_{j+1/2}) \Delta^+ U_j^n + D_{j+3/2} \Delta^+ U_{j+1}^n.$$

Taking the absolute value, applying the triangle inequality, and summing over all grid points  $j$ , we obtain

$$\|U^{n+1}\|_{TV} \leq \sum_j |C_{j-1/2}| |\Delta^+ U_{j-1}^n| + |(1 - C_{j+1/2} - D_{j+1/2})| |\Delta^+ U_j^n| + |D_{j+3/2}| |\Delta^+ U_{j+1}^n|.$$

Now realize that

$$\sum_j |C_{j-1/2}| |\Delta^+ U_{j-1}^n| = \sum_j |C_{j+1/2}| |\Delta^+ U_j^n|,$$

since we are assuming periodic boundary conditions. A similar argument can be used for problems on  $\mathbb{R}$ .

We can therefore shift indexes to  $j + 1/2$  to obtain

$$\|U^{n+1}\|_{TV} \leq \sum_j (|C_{j+1/2}| + |(1 - C_{j+1/2} - D_{j+1/2})| + |D_{j+1/2}|) |\Delta^+ U_j^n|.$$

If we now assume that

$$C_{j+1/2} \geq 0, \quad (1 - C_{j+1/2} - D_{j+1/2}) \geq 0, \quad D_{j+1/2} \geq 0,$$

we can eliminate the absolute value and recover

$$\|U^{n+1}\|_{TV} \leq \|U^n\|_{TV},$$

hence completing the proof.  $\square$

This result indicates a path to higher-order stable schemes. However, to realize that this may well be a complicated path, let us resort to an example.

**Example 8.5.** Consider the general scheme

$$U_j^{n+1} = U_j^n - \lambda [F_{j+1/2}^n - F_{j-1/2}^n] = G(U_{j-1}^n, U_j^n, U_{j+1}^n),$$

where, for simplicity, we restrict our attention to three point schemes.

Let us rewrite the scheme as

$$U_j^{n+1} = U_j^n - \lambda \frac{f(U_j^n) - F_{j-1/2}^n}{\Delta - U_j^n} \Delta^- U_j^n + \lambda \frac{f(U_j^n) - F_{j+1/2}^n}{\Delta + U_j^n} \Delta^+ U_j^n.$$

Employing the notation introduced in Theorem 8.4, this yields

$$C_{j+1/2} = \lambda \frac{f(U_{j+1}^n) - F_{j+1/2}^n}{\Delta^+ U_j^n}, \quad D_{j+1/2} = \lambda \frac{f(U_j^n) - F_{j+1/2}^n}{\Delta^+ U_j^n}.$$

Hence, the scheme is TVD, provided  $k$  is sufficiently small and the numerical flux  $F_{j+1/2}^n$  satisfies

$$\forall u \in [U_j^n, U_{j+1}^n]: \Delta^+ U_j^n (f(u) - F_{j+1/2}^n) \geq 0.$$

Such a flux, called an E-flux, was first introduced in [22].

For a monotone flux we have that  $F(\uparrow, \downarrow)$ . Furthermore, consistency of the numerical flux requires  $f(u) = F(u, u)$ . Consequently, by a Taylor expansion we have

$$F_{j+1/2}^n = F(U_j^n, U_{j+1}^n) = F(U_{j+1}^n, U_{j+1}^n) - \Delta^+ U_j^n \partial F_1(\xi, U_{j+1}^n),$$

where  $\xi \in [U_j^n, U_{j+1}^n]$ . This immediately implies that

$$C_{j+1/2} = \lambda \frac{f(U_{j+1}^n) - F_{j+1/2}^n}{\Delta^+ U_j^n} = \lambda \partial F_1(\xi, U_{j+1}^n) \geq 0.$$

A similar argument can be made for  $D_{j+1/2}$ . Hence, a monotone flux is an E-flux. We note that this does not rule out the development of TVD-stable high-order schemes based on an E-flux.

To explore this further, consider the semidiscrete scheme [22]

$$\frac{dU_j}{dt} = -\frac{F_{j+1/2} - F_{j-1/2}}{h}, \quad (8.10)$$

which we assume is a consistent approximation to a scalar conservation law with an entropy pair  $(\eta, \psi)$ .

Let us proceed to multiply (8.10) with  $\eta'(U_j)$  and introduce the numerical entropy flux,  $\Psi$ , as

$$\Psi_{j+1/2} = \eta'(U_{j+1}) (F_{j+1/2} - f(U_{j+1})) + \psi(U_{j+1}).$$

Adding  $\Delta^- \Psi_{j+1/2}$  on both sides of (8.10) we recover

$$b \left( \frac{d\eta(U_j)}{dt} + \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{h} \right) = (\Delta^+ \eta'(U_j)) F_{j+1/2} - \Delta^+ (\eta'(U_j) f(U_j) - \psi(U_j)).$$

Now consider

$$\begin{aligned} \Delta^+ \psi(U_j) &= \int_{U_j}^{U_{j+1}} \psi'(w) dw = \int_{U_j}^{U_{j+1}} \eta'(w) f'(w) dw \\ &= \Delta^+ (\eta'(U_j) f(U_j)) - \int_{U_j}^{U_{j+1}} \eta''(w) f(w) dw. \end{aligned}$$

Likewise, we have

$$(\Delta^+ \eta'(U_j)) F_{j+1/2} = \int_{U_j}^{U_{j+1}} \eta''(w) F_{j+1/2} dw,$$

to recover the cell entropy condition

$$b \left( \frac{d\eta(U_j)}{dt} + \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{h} \right) = \int_{U_j}^{U_{j+1}} \eta''(w) [F_{j+1/2} - f(w)] dw. \quad (8.11)$$

For a convex entropy, one easily concludes that the E-flux guarantees entropy dissipation and, thus, convergence to a unique solution. This result, obtained under light

assumptions, establishes a scheme with an E-flux as being both TVD-stable and possessing a cell entropy condition. Thus, a theoretical foundation as strong as that of a monotone scheme is recovered.

Now assume that we have an E-flux such that

$$C_{j+1/2} = \lambda \frac{f(U_{j+1}^n) - F_{j+1/2}^n}{\Delta^+ U_j^n} \geq 0.$$

If we consider the case where  $U_{j+1}^n \rightarrow U_j^n$  one quickly realizes that this requires  $F(\cdot, \downarrow)$ . Similar considerations for  $D_{j+1/2} \geq 0$  imply that  $F(\uparrow, \cdot)$ . In other words, an E-flux requires that  $F(\uparrow, \downarrow)$  and is, thus, a monotone flux. As an unfortunate consequence, a method using an E-flux can be at most first order accurate. ■

**Example 8.6.** Recall the linear wave equation

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

subject to an appropriate initial condition and periodic boundary conditions. We consider a scheme in the form

$$U_j^{n+1} = U_j^n - \lambda \frac{\Delta^+ U_j^n + \Delta^- U_j^n}{2} + \frac{\lambda}{2} [\alpha_{i+1/2} \Delta^+ U_j^n - \alpha_{i-1/2} \Delta^- U_j^n]. \quad (8.12)$$

Constants  $\alpha_{i\pm 1/2}$  may, in general, depend on  $U^n$ . Returning to Ex. 5.6, we observe that if  $\alpha_{i+1/2} = \alpha_{i-1/2} = \lambda^{-1}$  we recover the monotone Lax-Friedrichs scheme. On the other hand, taking  $\alpha_{i+1/2} = \alpha_{i-1/2} = \lambda$  results in the nonmonotone Lax-Wendroff scheme. Since  $\lambda \leq 1$ , this confirms that the former scheme can be expected to be more dissipative than the latter.

We now rewrite (8.12) in the form in Theorem 8.4 as

$$U_j^{n+1} = U_j^n - \frac{\lambda}{2} (1 + \alpha_{i-1/2}) \Delta^- U_j^n - \frac{\lambda}{2} (1 - \alpha_{i+1/2}) \Delta^+ U_j^n \quad (8.13)$$

such that

$$C_{j-1/2} = \frac{\lambda}{2} (1 + \alpha_{i-1/2}), \quad D_{j+1/2} = -\frac{\lambda}{2} (1 - \alpha_{i+1/2}).$$

If we first consider the simplest case of  $\alpha = \alpha_{i+1/2} = \alpha_{i-1/2}$ , the discussion in Ex. 5.2 indicates that the local truncation error is

$$\tau_j^k = -\frac{\alpha}{2h} (\alpha - \lambda) \frac{\partial^2 u}{\partial x^2} + \mathcal{O}(h^2).$$

Hence, to enable higher than first order accuracy, one must have  $\alpha = \lambda \leq 1$ . On the other hand, the result of Theorem 8.4 shows that the scheme can be TVD only if  $\alpha \geq 1$  to ensure that  $D_{j+1/2} \geq 0$ . The only solution is to take  $\alpha = 1$ , which results in

$$U_j^{n+1} = U_j^n - \lambda \Delta^- U_j^n.$$

This we recognize as the first order upwind scheme.

To develop a higher-order scheme, we follow the discussion in [28] and define  $\phi_{j+1/2} = 1 - \alpha_{j+1/2}$  in (8.13) to achieve

$$U_j^{n+1} = U_j^n - \frac{\lambda}{2} (2 - \phi_{j-1/2}) \Delta^- U_j^n - \frac{\lambda}{2} \phi_{j+1/2} \Delta^+ U_j^n. \quad (8.14)$$

By defining  $r_j = \Delta^- U_j^n / \Delta^+ U_j^n$ , we can eliminate  $\Delta^+ U_j^n$  and obtain

$$U_j^{n+1} = U_j^n - \lambda \left[ 1 + \frac{1}{2} \left( \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \right) \right] \Delta^- U_j^n = U_j^n - C_{j-1/2} \Delta^- U_j^n.$$

Following Theorem 8.4, the scheme is TVD provided

$$0 \leq \lambda \left[ 1 + \frac{1}{2} \left( \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \right) \right] \leq 1$$

or, equivalently,

$$-2 \leq \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \leq \frac{2(1 - \lambda)}{\lambda}.$$

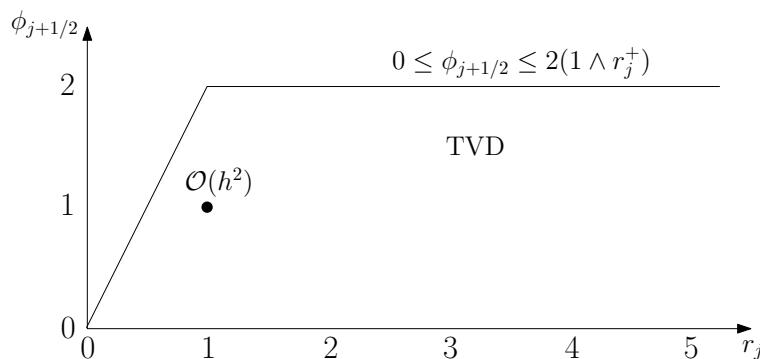
If we restrict our attention to the case of  $\lambda \leq \frac{1}{2}$ , this inequality further simplifies to

$$0 \leq \phi_{j+1/2} \leq 2(1 \wedge r_j^+),$$

where we have borrowed the notation from Theorem 4.5. When these conditions hold, the scheme is TVD, as illustrated in Fig. 8.3.

To understand the requirements on accuracy, consider the following approximation of the spatial derivative (8.14):

$$\frac{\lambda}{2} (2 - \phi_{j-1/2}) \Delta^- U_j^n + \frac{\lambda}{2} \phi_{j+1/2} \Delta^+ U_j^n.$$



**Figure 8.3.** The region of  $(r, \phi)$ -space where the scheme for the linear wave equation is total variation diminishing (TVD). Second order accuracy can be achieved provided  $\phi_{j+1/2}(1) = 1$  and  $\phi_{j+1/2}(r)$  is smooth.

Assuming that the exact solution is smooth, the semidiscrete local truncation error is

$$\tau_j(t) = -\frac{a}{2} (2 - \phi_{j-1/2} + \phi_{j+1/2}) \frac{\partial u}{\partial x} - a \frac{h}{4} (-2 + \phi_{j-1/2} + \phi_{j+1/2}) \frac{\partial^2 u}{\partial x^2} + \mathcal{O}(h^2).$$

To ensure second order accuracy, we must require that

$$\phi_{j+1/2} - \phi_{j-1/2} = \mathcal{O}(h^2), \quad 1 - \frac{1}{2} (\phi_{j+1/2} + \phi_{j-1/2}) = \mathcal{O}(h). \quad (8.15)$$

Assuming that  $\phi_{j+1/2} = \phi_{j+1/2}(r_j)$ , i.e., a function of  $r_j$ , we first observe that

$$r_j = \frac{\Delta^- U_j^n}{\Delta^+ U_j^n} \simeq 1 - h \frac{u_{xx}(x_j)}{u_x(x_j)} + \mathcal{O}(h^2).$$

This implies that as  $h$  vanishes,  $r_j$  approaches the value of one and, furthermore, we have

$$\begin{aligned} \phi_{j+1/2}(r_j) &\simeq \phi_{j+1/2}(1) - h \frac{u_{xx}(x_j)}{u_x(x_j)} \phi'_{j+1/2}(1) + \mathcal{O}(h^2), \\ \phi_{j-1/2}(r_{j-1}) &\simeq \phi_{j-1/2}(1) - h \frac{u_{xx}(x_j)}{u_x(x_j)} \phi'_{j-1/2}(1) + \mathcal{O}(h^2). \end{aligned}$$

Combining this with (8.15) we realize that sufficient conditions for second order accuracy is that  $\phi_{j+1/2}(1) = 1$  and that  $\phi_{j+1/2}(r_j)$  is a smooth function of  $r_j$ . ■

Unfortunately, there are limits to what can be achieved [23].

**Theorem 8.7.** *A TVD-stable scheme can be at most first order accurate at local extrema that are not sonic.*

**Proof.** Following Theorem 8.4, we express the scheme as

$$U_j^{n+1} = U_j^n - \frac{k}{h} (\tilde{C}_{j-1/2} \Delta^- U_j^n - \tilde{D}_{j+1/2} \Delta^+ U_j^n),$$

where

$$C_{j-1/2} = \frac{k}{h} \tilde{C}_{j-1/2}, \quad D_{j+1/2} = \frac{k}{h} \tilde{D}_{j+1/2},$$

to make the dependence on  $h$  explicit. Now consider the local truncation error:

$$\begin{aligned} &\frac{1}{h} (\tilde{C}_{j-1/2} \Delta^- U_j^n - \tilde{D}_{j+1/2} \Delta^+ U_j^n) \\ &= (\tilde{C}_{j-1/2}(u) - \tilde{D}_{j+1/2}(u)) \frac{\partial u}{\partial x} - \frac{h}{2} (\tilde{C}_{j-1/2}(u) + \tilde{D}_{j+1/2}(u)) \frac{\partial^2 u}{\partial x^2} \\ &\quad + (\tilde{C}_{j-1/2} - \tilde{C}_{j-1/2}(u) + \tilde{D}_{j+1/2}(u) - \tilde{D}_{j+1/2}) \frac{\partial u}{\partial x} + \mathcal{O}(h^2), \end{aligned}$$

where  $\tilde{C}_{j-1/2}(u) = \tilde{C}_{j-1/2}(u, u, u, \dots, u)$  and  $u$  is an exact, smooth solution.

For consistency, we require that

$$\tilde{C}_{j-1/2}(u) - \tilde{D}_{j+1/2}(u) = -f'(u).$$

Second order accuracy at an extrema ( $u_{xx} \neq 0$ ) requires

$$\tilde{C}_{j-1/2}(u) + \tilde{D}_{j+1/2}(u) = 0.$$

When combined, this yields the conditions

$$\tilde{C}_{j-1/2}(u) = -\frac{1}{2}f'(u), \quad \tilde{D}_{j+1/2}(u) = \frac{1}{2}f'(u).$$

However, this violates the result of Theorem 8.4, where both coefficients must be positive for the scheme to be TVD.  $\square$

This result limits the local accuracy of a TVD scheme to first order, resulting in a limit of second order accuracy in a global norm [13].

For problems beyond one spatial dimension, the situation is in fact worse.

**Theorem 8.8.** *A TVD scheme for a two-dimensional conservation law is at most first order accurate.*

This result is proved in [10], to which we refer the reader for the details.

This is a disappointing result, as it appears to limit the value of high-order accurate TVD schemes to the one-dimensional case, and restricts the error to  $\mathcal{O}(h^{1+1/p})$  when measured in  $\|\cdot\|_{h,p}$ , provided there is a finite number of extrema. However, in spite of this result, the application of one-dimensional TVD schemes to the solution of multidimensional conservation laws often yields excellent results.

If we slightly relax the condition of TVD and, instead, require that the scheme is total variation bounded (TVB) we recover an interesting result. Let us allow the total variation to increase slightly:

$$\|U^{n+1}\|_{TV} \leq \|U^n\|_{TV} + \alpha k \leq (1 + \alpha k) \|U^n\|_{TV},$$

where  $\alpha > 0$  is a constant. The total variation remains bounded since

$$\|U^n\|_{TV} \leq (1 + \alpha k)^n \|U^0\|_{TV} \leq e^{\alpha t^n} \|U^0\|_{TV},$$

since  $nk = t^n$ . As shown in [24], this slight modification allows the scheme to maintain accuracy at smooth extrema, thus renewing the hope of formulating schemes of uniformly high order.

Total variation boundedness suffices for convergence, as stated in the following result [17].

**Theorem 8.9.** *Assume that  $U_j^n$  is a compactly supported and uniformly bounded sequence of solutions for  $(h_l, k_l) \rightarrow 0$  as  $l \rightarrow \infty$ . If the numerical flux is consistent and the solution is total variation bounded, then there exists a subsequence that converges to a weak solution of the conservation law.*

To realize the potential of developing TVB schemes, consider the three point scheme

$$U_j^{n+1} = U_j^n - \lambda \left[ F_{j+1/2}^n - F_{j-1/2}^n \right], \quad (8.16)$$

and assume it is TVD. The first question to address is how much change is needed to overcome the accuracy restrictions of the TVD property. Let us assume that we can modify the numerical flux such that

$$\tilde{F}_{j+1/2}^n = F_{j+1/2}^n + c_{j+1/2},$$

for some suitable  $c_{j+1/2}$ . Now consider this scheme to advance the conservation law

$$\tilde{U}_j^{n+1} = \tilde{U}_j^n - \lambda \left[ \tilde{F}_{j+1/2}^n - \tilde{F}_{j-1/2}^n \right], \quad (8.17)$$

and express this scheme in TVD form as

$$U_j^{n+1} = U_j^n - C_{j-1/2} \Delta^- U_j^n + D_{j+1/2} \Delta^+ U_j^n,$$

and recall that

$$C_{j+1/2} = \lambda \frac{f(U_{j+1}^n) - \tilde{F}_{j+1/2}^n}{\Delta^+ U_j^n}, \quad D_{j+1/2} = \lambda \frac{f(U_j^n) - \tilde{F}_{j+1/2}^n}{\Delta^+ U_j^n}.$$

To understand the magnitude of changes we can make, consider

$$\tilde{F}_{j+1/2}^n - f(U_j^n),$$

which should be modified in a way that impacts accuracy at extrema without destroying stability. By consistency,  $f(U_j^n) = F(U_j^n, U_j^n)$  and we recover

$$\tilde{F}_{j+1/2}^n - f(U_j^n) = c_{j+1/2} + \Delta^+ U_j^n \partial_2 F(U_j^n, U_j^n) + \frac{1}{2} (\Delta^+ U_j^n)^2 \partial_{22} F(U_j^n, U_j^n) - \dots$$

If we assume that  $U_j^n$  is sufficiently smooth and  $h$  is sufficiently small, we recover

$$\tilde{F}_{j+1/2}^n - f(U_j^n) = c_{j+1/2} + C_1 h + C_2 h^2,$$

where the three coefficients  $C_i$  depend on  $u, u_x, u_{xx}$  but not on  $h$ . If we first consider a case in which  $C_1 > 0$ , no correction term  $c_{j+1/2}$  is needed. However, if  $C_1 = 0$ , as would be the case at a local extrema,  $c_{j+1/2}$  can be used to modify the accuracy close to the extrema. Hence, we can conclude that  $c_{j+1/2} = C h^2$  suffices to control a local reduction of the accuracy in the TVD scheme.

However, it remains unclear if such a modification yields a TVB-stable scheme. To understand this, consider the two schemes, (8.16) and (8.17). Assuming that  $U_j^n = \tilde{U}_j^n$  and subtracting one scheme from the other yields

$$\tilde{U}_j^{n+1} = U_j^{n+1} - \frac{k}{h} [c_{j+1/2} - c_{j-1/2}],$$

for which we have

$$\|\tilde{U}_j^{n+1}\| \leq \|U_j^{n+1}\|_{TV} + 2N \frac{k}{h} \max |c_{j+1/2}| \leq \|U_j^n\|_{TV} + 2L \frac{k}{h^2} \max |c_{j+1/2}|.$$

Here  $L$  is the length of the domain and  $N$  the number of grid points.

Provided  $\max|c_{j+1/2}| \propto h^2$ , TVB-stability follows from TVD-stability of the original scheme. Hence, it is both necessary and sufficient to seek an  $\mathcal{O}(h^2)$  modification of TVD schemes to overcome the reduction to first order at local extrema. As we shall later discuss, this is surprisingly easy and opens up the possibility of constructing schemes that are uniformly high-order accurate.

While this is encouraging news, we should not forget that convergence to a weak solution, as guaranteed by TVD/TVB-stability, does not guarantee convergence to the entropy solution. To illustrate this, consider an example, inspired by [17].

**Example 8.10.** Consider the three point scheme

$$U_j^{n+1} = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n],$$

and express the numerical flux as

$$F(u, v) = \frac{f(u) + f(v)}{2} - \frac{h}{2k} (v - u) Q(u, v),$$

with  $F(u, u) = f(u)$  for consistency;

$$Q(u, v) = \frac{k}{h} \frac{f(u) + f(v) - 2F(u, v)}{v - u}$$

simply reflects the tautology. Hence, we can write the numerical flux as

$$F(U_j^n, U_{j+1}^n) = F_{j+1/2}^n = \frac{f(U_j^n) + f(U_{j+1}^n)}{2} - \frac{h}{2k} \Delta^+ U_j^n Q(U_j^n, U_{j+1}^n).$$

This exposes that the role of  $Q(u, v)$  is to add dissipation and suggests that we can consider alternative formulations of this term to introduce dissipation.

Let us define

$$\alpha_{j+1/2} = \frac{k}{h} \frac{\Delta^+ f(U_j^n)}{\Delta^+ U_j^n}, \quad q_{j+1/2} = Q(U_j^n, U_{j+1}^n),$$

and

$$C_{j-1/2} = \frac{1}{2} (\alpha_{j-1/2} + q_{j-1/2}), \quad D_{j+1/2} = -\frac{1}{2} (\alpha_{j+1/2} - q_{j+1/2}).$$

The three point scheme becomes

$$U_j^{n+1} = U_j^n - C_{j-1/2} \Delta^- U_j^n + D_{j+1/2} \Delta^+ U_j^n.$$

If we now assume that

$$|\alpha_{j+1/2}| \leq q_{j+1/2} \leq 1, \quad (8.18)$$

it is straightforward to see that the scheme satisfies Theorem 8.4 and, thus, is TVD.

Let us now redefine the dissipative function  $Q(u, v)$  as

$$Q^*(u, v) = \frac{k}{h} \frac{f(u) - f(v)}{u - v}. \quad (8.19)$$

In this case,  $\alpha_{i+1/2} = q_{i+1/2}$  and, if we choose  $k$  sufficiently small, (8.18) guarantees TVD-stability.

Now consider the special case of Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to the initial condition

$$u_0(x) = \begin{cases} -1, & x < 0, \\ 1, & x \geq 0. \end{cases}$$

If we use the numerical flux

$$F_{j+1/2}^n = \frac{f(U_j^n) + f(U_{j+1}^n)}{2} - \frac{b}{2k} \Delta^+ U_j^n Q^*(U_j^n, U_{j+1}^n),$$

it is easily seen that  $F_{j+1/2}^n - F_{j-1/2}^n = 0$ . Hence, the scheme returns an entropy violating solution, confirming that TVD schemes are not guaranteed to compute the entropy solution.

The problem here is that the dissipation in  $Q^*$ , defined in (8.19), is not sufficient to establish a cell entropy inequality. If we return to the Lax–Friedrichs scheme, we see that

$$Q(u, v) = \frac{k}{b} \max_u |f'(u)|.$$

Since this is a monotone scheme, we know that it computes the entropy solution. It satisfies (8.18) provided

$$\frac{k}{b} \max_u |f'(u)| \leq 1,$$

which we recognize as the CFL condition. As expected, the scheme is TVD. ■

### 8.2.2 • Entropy stability

As Ex. 8.10 highlights, seeking total variation stability provides a necessary but not sufficient condition that the entropy solution is computed. To achieve this, we must identify methods that satisfy a cell entropy condition

$$\eta(U^{n+1})_j - \eta(U^n)_j + \frac{k}{b} (\Psi(U^n)_j - \Psi(U^n)_{j-1}) \leq 0,$$

where  $\eta$  is the entropy function and  $\Psi$  is the numerical entropy flux of the scheme, as discussed in detail in sections 2.3 and 3.3. Schemes which satisfy a cell entropy condition are called entropy stable.

**Example 8.11.** Let us again consider the linear wave equation

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0, \quad (8.20)$$

subject to an initial condition and periodic boundary conditions.

We explore the formulation [21]

$$U_j^{n+1} = U_j^n - \lambda(U_{j+1/2}^n - U_{j-1/2}^n), \quad (8.21)$$

where the details of the scheme are defined by specifying how  $U_{j\pm 1/2}^n$  is to be recovered.

If we choose the entropy pair as  $(\eta, \psi) = (u^2, \alpha u^2)$ , we can express the change in entropy—the entropy production—as

$$\Delta S(U^n)_j = (U_j^{n+1})^2 - (U_j^n)^2 + \lambda((U_{j+1/2}^n)^2 - (U_{j-1/2}^n)^2).$$

Eliminating  $U_j^{n+1}$  with (8.21) yields

$$\Delta S(U^n)_j = \lambda \left[ (U_{j+1/2}^n - U_j^n)^2 - (U_j^n - U_{j-1/2}^n)^2 \right] + \lambda^2 (U_{j+1/2}^n - U_{j-1/2}^n)^2. \quad (8.22)$$

To satisfy the cell entropy condition, we must require that the entropy decreases, i.e., that  $\Delta S(U^n)_j \leq 0$ . Consider the upwind scheme, recovered by taking  $U_{j+1/2}^n = U_j^n$ ,  $U_{j-1/2}^n = U_{j-1}^n$ , in which case

$$\Delta S(U^n)_j = -\lambda(1-\lambda)(\Delta^- U_j^n)^2.$$

This is clearly negative provided only that the CFL condition is fulfilled. Hence, enforcing the cell entropy condition ensures stability.

If we introduce

$$R_j = \frac{U_{j+1/2}^n - U_j^n}{U_j^n - U_{j-1/2}^n},$$

we can eliminate  $U_{j+1/2}^n$  in (8.22) and recover

$$\begin{aligned} \Delta S(U^n)_j &= \lambda (U_j^n - U_{j-1/2}^n)^2 \left[ (R_j - 1)^2 + \lambda(R_j + 1)^2 \right] \\ &= -\lambda (U_j^n - U_{j-1/2}^n)^2 \left[ (R_j + 1)((1-\lambda) - R_j(1+\lambda)) \right]. \end{aligned}$$

Taking  $\alpha > 0$  and  $0 \leq \lambda \leq 1$  for stability (a similar argument applied if  $\alpha < 0$ ), this requires

$$-1 \leq R_j \leq \frac{1-\lambda}{1+\lambda} \leq 1.$$

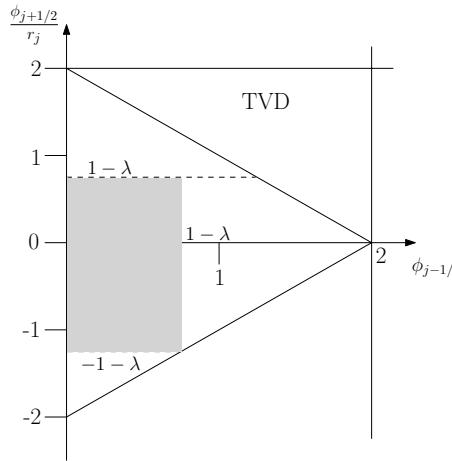
If we take  $\lambda = 1$ , only the upwind scheme  $U_{j+1/2}^n = U_j^n$  is an admissible solution.

Let us consider a scheme based on

$$U_{j+1/2}^n = U_j^n + \phi_{j+1/2} \Delta^+ U_j^n,$$

to obtain

$$-1 \leq \frac{\phi_{j+1/2}}{(1-\phi_{j-1/2})r_j} \leq \frac{1-\lambda}{1+\lambda}, \quad r_j = \frac{\Delta^- U_j^n}{\Delta^+ U_j^n}.$$



**Figure 8.4.** Illustration to facilitate the identification of the region in which the scheme for the linear wave equation satisfies the entropy condition, marked by the shaded region.

Recall that the requirement for the scheme to be TVD is

$$0 \leq \phi_{j+1/2} \leq 2(1 \wedge r_j^+),$$

and that second order accuracy requires that  $\phi_{j+1/2}(1) = 1$ . Guided by Fig. 8.4, we recover the conditions for the scheme to satisfy a cell entropy condition as

$$0 \leq \phi_{j-1/2} \leq 1 - \lambda, \quad -(1 - \lambda) \leq \frac{\phi_{j+1/2}}{r_j} \leq 1 - \lambda, \quad (8.23)$$

which can be expressed as

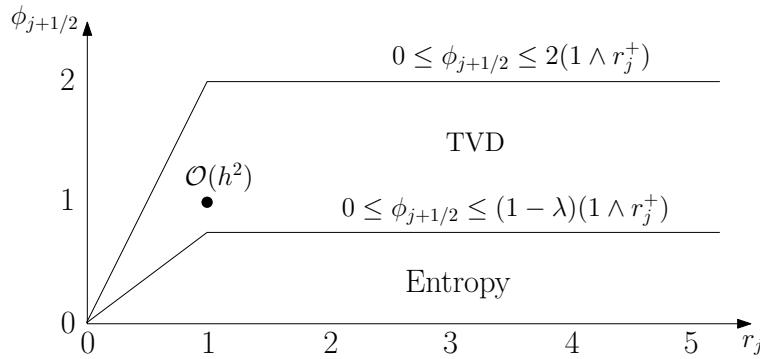
$$0 \leq \phi_{j+1/2} \leq (1 - \lambda)(1 \wedge r_j^+).$$

Comparing the two conditions, illustrated in Fig. 8.5, we see that the condition for a cell entropy condition is stronger than the one required for TVD-stability. Unfortunately, we also observe that the condition for entropy decay generally does not allow  $\phi_{j+1/2}(1) = 1$ , thus precluding higher-order accuracy for a scheme of this type. ■

It is clear that we should seek methods that are not only TVD-stable but also guarantee the computation of the entropy solution. As we experienced in Ex. 8.10, this requires that the scheme be sufficiently dissipative. Such schemes, derived by enforcing a local cell entropy condition, are termed entropy stable schemes, and have been developed in detail in [29, 30].

Let us first consider the semidiscrete setting in which the scalar conservation law,

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$



**Figure 8.5.** Region in  $(r, \phi)$  where the scheme for the linear wave equation is total variation diminishing (TVD) and satisfies a cell entropy condition. Second order accuracy can be achieved, provided  $\phi_{j+1/2}(1) = 1$  and  $\phi_{j+1/2}(r)$  is smooth.

is approximated as

$$\frac{dU_j}{dt} + \frac{F_{j+1/2} - F_{j-1/2}}{h} = 0,$$

where  $U_j(t) = u(x_j, t)$ . For simplicity we consider a three point scheme. We also assume that the conservation law has an entropy pair  $(\eta(u), \psi(u))$  which obeys

$$\frac{\partial \eta}{\partial t} + \frac{\partial \psi}{\partial x} \leq 0,$$

where  $u$  is the weak entropy solution. The corresponding semidiscrete form of this requirement is

$$\frac{d\eta_j}{dt} + \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{h} \leq 0, \quad (8.24)$$

where  $\eta_j(t) = \eta(U_j(t))$  and  $\Psi_{j+1/2} = \Psi(U_j, U_{j+1})$  is a consistent numerical entropy flux. If this inequality holds for each cell, the entropy condition is satisfied globally.

In the following, we focus on the scalar conservation laws for which an entropy as  $\eta(u) = u^2/2$  is suitable. Following [30] we call the scheme entropy conservative if

$$U_j(F_{j+1/2} - F_{j-1/2}) = \Psi_{j+1/2} - \Psi_{j-1/2}, \quad (8.25)$$

which ensures equality in (8.24). The correspondence with the continuous identity  $\eta' f' = \psi'$  is clear.

Rewriting (8.25) we recover

$$\begin{aligned} U_j(F_{j+1/2} - F_{j-1/2}) &= \frac{1}{2} (U_j + U_{j+1}) F_{j+1/2} - \frac{1}{2} (U_j + U_{j-1}) F_{j-1/2} \\ &\quad - \frac{1}{2} \Delta^+ U_j F_{j+1/2} - \frac{1}{2} \Delta^- U_j F_{j-1/2} \\ &= \Psi_{j+1/2} - \Psi_{j-1/2}, \end{aligned}$$

which defines the numerical entropy flux

$$\Psi_{j+1/2} = \frac{1}{2} (U_j + U_{j+1}) F_{j+1/2} - \frac{1}{2} \Delta^+ U_j F_{j+1/2}.$$

Summing over  $j$  and reordering the terms in (8.25) yields an alternative expression for entropy conservation

$$\Delta^+ U_j F_{j+1/2} = \Delta^+ \zeta_j.$$

To define the potential  $\zeta_j$ , let us consider

$$F_{j+1/2}^* = \int_0^1 f(U_j + \xi \Delta^+ U_j) d\xi,$$

and consider

$$\Delta^+ U_j F_{j+1/2}^* = \int_0^1 \Delta^+ U_j f(U_j + \xi \Delta^+ U_j) d\xi = \int_{U_j}^{U_{j+1}} f(s) ds = \Delta^+ \zeta_j.$$

Hence, the natural definition of the potential  $\zeta$  is

$$f = \frac{\partial \zeta}{\partial u} \Rightarrow \zeta = u f - \psi,$$

where the latter follows directly from  $\eta' f' = \psi'$  with  $\eta(u) = u^2/2$ . This allows us to recover the entropy conservative flux

$$\Psi_{j+1/2} = \frac{1}{2} (U_j + U_{j+1}) F_{j+1/2}^* - \frac{1}{2} (\zeta_{j+1} + \zeta_j).$$

Now express the numerical flux:

$$\begin{aligned} F_{j+1/2}^* &= \int_0^1 f(U_j + \xi \Delta^+ U_j) d\xi = \int_0^1 \frac{d}{d\xi} \left( \xi - \frac{1}{2} \right) f(U_j + \xi \Delta^+ U_j) d\xi \\ &= \frac{1}{2} (f(U_{j+1}) + f(U_j)) - \frac{\Delta^+ U_j}{2} \int_0^1 (2\xi - 1) f'(U_j + \xi \Delta^+ U_j) d\xi \\ &= \frac{1}{2} (f(U_{j+1}) + f(U_j)) - \frac{1}{2} Q_{j+1/2}^* \Delta^+ U_j, \end{aligned}$$

where

$$Q_{j+1/2}^* = Q^*(U_j, U_{j+1}) = \int_0^1 (2\xi - 1) f'(U_j + \xi \Delta^+ U_j) d\xi.$$

Returning to Ex. 8.10, we recognize this as the viscosity form in which  $Q^*$  controls the amount of dissipation. Also, we recall that its careful definition is essential to ensure recovery of the entropy solution. The following theorem offers an important guideline [29].

**Theorem 8.12.** *There exists a conservative scheme for which  $Q_{j+1/2} \geq Q_{j+1/2}^*$  is entropy stable.*

*Proof.* We begin by writing the conservative scheme in dissipative form as

$$\frac{dU_j}{dt} + \frac{F_{j+1/2}^* - F_{j-1/2}^*}{h} = \frac{1}{2b} (Q_{j+1/2} \Delta^+ U_j - Q_{j-1/2} \Delta^- U_j).$$

Multiply with  $U_j$  and recall the cell entropy condition to recover

$$U_j (F_{j+1/2}^* - F_{j-1/2}^*) = \Psi_{j+1/2} - \Psi_{j-1/2}.$$

Furthermore, we obtain

$$\begin{aligned} & U_j (Q_{j+1/2} \Delta^+ U_j - Q_{j-1/2} \Delta^- U_j) \\ &= -\frac{1}{2} [\Delta^+ U_j Q_{j+1/2} \Delta^+ U_j + \Delta^- U_j Q_{j-1/2} \Delta^- U_j] \\ & \quad + \frac{1}{2} [(U_j + U_{j+1}) Q_{j+1/2} \Delta^+ U_j - (U_j + U_{j-1}) Q_{j-1/2} \Delta^- U_j]. \end{aligned}$$

Hence the cell entropy equation becomes

$$\frac{d\eta_j}{dt} + \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{h} = -\frac{1}{4b} [\Delta^+ U_j Q_{j+1/2} \Delta^+ U_j + \Delta^- U_j Q_{j-1/2} \Delta^- U_j],$$

provided the numerical entropy flux is defined as

$$\Psi_{j+1/2} = U_j F_{j+1/2}^* - \frac{1}{4} (U_j + U_{j+1}) Q_{j+1/2} \Delta^+ U_j.$$

Now express  $Q_{j+1/2} = \alpha_{j+1/2} + Q_{j+1/2}^*$  and recall that  $Q_{j+1/2}^*$  is associated with the entropy conservative scheme. We obtain

$$\frac{d\eta_j}{dt} + \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{h} = -\frac{1}{4b} [\Delta^+ U_j \alpha_{j+1/2} \Delta^+ U_j + \Delta^- U_j \alpha_{j-1/2} \Delta^- U_j].$$

If  $Q_{j+1/2} > Q_{j+1/2}^*$ , i.e.,  $\alpha_{j+1/2} \geq 0$ , the cell entropy inequality is satisfied, completing the proof.  $\square$

Consider now

$$F_{j+1/2}^* = \int_0^1 f(U_j + \xi \Delta^+ U_j) d\xi = \frac{1}{2} (f(U_j) + f(U_{j+1})) + \mathcal{O}(\Delta^+ U_j),$$

such that

$$\frac{F_{j+1/2}^* - F_{j-1/2}^*}{h} = \frac{f(U_{j+1}) - f(U_{j-1})}{2h} + \mathcal{O}(h^2).$$

This shows that the entropy conservative scheme will be second order accurate in regions where the solution is smooth.

**Example 8.13.** Let us consider a few schemes and their entropy stability. First, we rewrite

$$\begin{aligned}
 Q_{j+1/2}^* &= \int_0^1 (2\xi - 1) f'(U_j + \xi \Delta^+ U_j) d\xi \\
 &= \int_{-1/2}^{1/2} 2\zeta f'((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta \\
 &= \int_{-1/2}^{1/2} \frac{d}{d\zeta} \left( \zeta^2 - \frac{1}{4} \right) f'((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta \\
 &= \Delta^+ U_j \int_{-1/2}^{1/2} \left( \frac{1}{4} - \zeta^2 \right) f''((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta.
 \end{aligned} \tag{8.26}$$

Using the first expression for  $Q_{j+1/2}^*$ , we recover a crude bound as

$$Q_{j+1/2}^* \leq \max_{w \in [U_j, U_{j+1}]} |f'(w)|,$$

which reflects the dissipation associated with the local Lax–Friedrichs flux. A slightly sharper bound is recovered from the last expression in (8.26) as

$$Q_{j+1/2}^* \leq \frac{1}{6} \max_{w \in [U_j, U_{j+1}]} |f''(w)| |\Delta^+ U_j|,$$

to which we shall return shortly. A final estimate can be obtained if we consider

$$\begin{aligned}
 Q_{j+1/2}^* &= \int_{-1/2}^{1/2} 2\zeta f'((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta \\
 &\leq \int_{-1/2}^{1/2} |f'((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j)| d\zeta = \frac{1}{\Delta^+ U_j} \int_{U_j}^{U_{j+1}} |f'(s)| ds,
 \end{aligned}$$

which results in the dissipation associated with the Engquist–Osher flux (6.9).

Consider the general form of the viscous term as

$$Q_{j+1} = \frac{f(U_j) + f(U_{j+1}) - 2F_{j+1/2}^*}{\Delta^+ U_j}.$$

We now seek the maximum dissipation across all increasing solutions, which results in

$$Q_{j+1/2}^* \leq \max_{u \in [U_j, U_{j+1}]} \left| \frac{f(U_j) + f(U_{j+1}) - 2f(u)}{\Delta^+ U_j} \right|.$$

This is the dissipation associated with the Godunov flux. This confirms the entropy stability of Godunov’s method, as discussed in Ex. 6.1.

Let us finally restrict our attention to the special case of a convex flux and consider

$$Q_{j+1/2}^* = \Delta^+ U_j \int_{-1/2}^{1/2} \left( \frac{1}{4} - \zeta^2 \right) f''((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta.$$

Since  $f(u)$  is convex,  $Q_{j+1/2}^* < 0$  for  $\Delta^+ U_j < 0$ , i.e., the shock case. For the rarefaction wave,  $\Delta^+ U_j \geq 0$  and dissipation is needed. A direct estimate is

$$Q_{j+1/2}^* = \Delta^+ U_j \frac{1}{4} \int_{-1/2}^{1/2} f''((U_j + U_{j+1})/2 + \zeta \Delta^+ U_j) d\zeta = \frac{1}{4} [f'(U_{j+1}) - f'(U_j)].$$

Hence, sufficient dissipation is

$$Q_{j+1/2}^* = \frac{1}{4} [f'(U_{j+1}) - f'(U_j)]^+,$$

which is closely related to that of the Lax–Wendroff flux discussed in section 5.2. ■

So far the discussion has focused on the semidiscrete formulation. However, the analysis carries over to the fully discrete case [21].

**Theorem 8.14.** *The fully discrete scheme*

$$U_j^{n+1} = U_j^n - \frac{k}{b} (F_{j+1/2}^n - F_{j-1/2}^n)$$

is entropy stable if the semidiscrete scheme is entropy stable.

**Proof.** We express the entropy at level  $n+1$  as

$$\eta(U^{n+1})_j = \eta(U^n)_j + \eta'(U^n)_j (U_j^{n+1} - U_j^n) + \frac{1}{2} \eta''(\xi)_j (U_j^{n+1} - U_j^n)^2,$$

where  $\xi \in [U_j, U_{j+1}]$ . Using the scheme itself we recover

$$\eta'(U^n)_j (U_j^{n+1} - U_j^n) = -\eta'(U^n)_j \frac{k}{b} (F_{j+1/2}^n - F_{j-1/2}^n) = -\frac{k}{b} (\Psi_{j+1/2}^n - \Psi_{j-1/2}^n)$$

such that

$$\frac{\eta(U^{n+1})_j - \eta(U^n)_j}{k} + \frac{\Psi_{j+1/2}^n - \Psi_{j-1/2}^n}{b} = -\frac{1}{2k} \eta''(\xi)_j (U_j^{n+1} - U_j^n)^2 \leq 0.$$

The last inequality follows from the convexity of the entropy. □

**Example 8.15.** We consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to an appropriate initial condition and periodic boundary conditions. As discussed in Ex. 2.11, we can choose  $(u^2, \frac{4}{3}u^3)$  as the entropy pair. A conservative semidiscrete scheme is given as

$$\frac{dU_j}{dt} + \frac{F_{j+1/2} - F_{j-1/2}}{b} = 0,$$

where  $F_{j+1/2} = f(U_{j+1/2}) = f_{j+1/2}$ . We follow the notation of Ex. 8.6 and assume

$$U_{j+1/2} = U_j + \frac{\phi_{j+1/2}}{2} \Delta^+ U_j.$$

The function  $\phi_{j+1/2}$  is unknown but we recall that  $0 \leq \phi_{j+1/2} \leq 2$  is required for the scheme to be TVD and rewrite this as

$$\phi_{j+1/2} \left( 1 - \frac{\phi_{j+1/2}}{2} \right) \geq 0,$$

which implies that

$$(\Delta^+ U_j)^2 \frac{\phi_{j+1/2}}{2} \left( 1 - \frac{\phi_{j+1/2}}{2} \right) = (U_{j+1/2} - U_j)(U_{j+1} - U_{j+1/2}) \geq 0.$$

Hence,  $U_{j+1/2}$  must take a value between  $U_j$  and  $U_{j+1}$ .

Now consider the entropy production

$$\begin{aligned} h \Delta S_j &= -2U_j(f_{j+1/2} - f_{j-1/2}) + \psi_{j+1/2} - \psi_{j-1/2} \\ &= \left[ -2U_j(f_{j+1/2} - f_j) + (\psi_{j+1/2} - \psi_j) \right] + \left[ -2U_j(f_j - f_{j-1/2}) + (\psi_j - \psi_{j-1/2}) \right] \\ &= \Delta S_j^+ + \Delta S_j^-, \end{aligned}$$

and require

$$\Delta S_j = \Delta S_j^+ + \Delta S_j^- \leq 0$$

to ensure entropy stability. To obtain a cell entropy condition, we must choose  $U_{j+1/2}$  such that

$$\Delta S_j^+ \leq 0, \quad \Delta S_{j+1}^- \leq 0.$$

If we first consider the case of  $U_j > 0, U_{j+1} > 0$ , a natural choice is  $U_{j+1/2} = U_j$  for upwinding, resulting in  $\phi_{j+1/2} = 0$ . In this case,  $\Delta S_j^+ = 0$  and

$$\Delta S_{j+1}^- = -\frac{2}{3} U_{j+1}^3 + 2U_{j+1} U_j^2 - \frac{4}{3} U_j^3.$$

Furthermore, the directional derivative of  $\Delta S_{j+1}^-$  along a direction perpendicular to  $U_j = U_{j+1}$  is always negative for  $U_j \geq U_{j+1}$ , proving that  $\Delta S_{j+1}^- \leq 0$ .

Provided  $U_j < 0, U_{j+1} < 0$ , the natural choice is  $U_{j+1/2} = U_{j+1}$  for downwinding, resulting in  $\phi_{j+1/2} = 2$ . In this case  $\Delta S_{j+1}^- = 0$  and  $\Delta S_j^+ \leq 0$ . If we have a situation with a shock with  $U_j \geq 0 \geq U_{j+1}$ , we can make a similar decision on upwinding or downwinding based on the sign of  $U_j + U_{j+1}$ .

For the case of the rarefaction wave, where  $U_j \leq 0 \leq U_{j+1}$ , we must address potential problems at the sonic point by enforcing

$$f'(U_{j+1/2}) = 2U_{j+1/2} = 0,$$

by requiring that

$$\phi_{j+1/2} = -\frac{2U_j}{\Delta^+ U_j}.$$

Hence, we have derived an entropy stable scheme [21] through the definition of  $\phi_{j+1/2}$  as

$$\phi_{j+1/2} = \begin{cases} 1 - \text{sign}(U_j), & U_j U_{j+1} > 0, \\ 1 - \text{sign}(U_j + U_{j+1}), & U_j \geq 0 \geq U_{j+1}, \\ -\frac{2U_j}{\Delta^+ U_j}, & U_j < 0 < U_{j+1}. \end{cases}$$

Since the scheme is based on upwinding, it is only first order accurate.

Let us evaluate the performance of the scheme by solving Burgers' equation on  $[0, 1]$  with the initial condition

$$u(x, 0) = 0.5 - \sin(2\pi x).$$

Figure 8.6 shows the computed solution along with the evolution of  $\phi$ , illustrating the expected switching between the two extreme values which results in first order accuracy of the scheme.

To construct a second order scheme, we recall the cell entropy condition

$$\Delta S_j^+ + \Delta S_j^- \leq 0.$$

Again restricting our attention to Burgers' equation, we obtain

$$\begin{aligned} \Delta S_j^+ &= -2U_j \left( (U_{j+1/2})^2 - U_j^2 \right) + \frac{4}{3} \left( (U_{j+1/2})^3 - U_j^3 \right) \\ &= 2 \left( U_{j+1/2} - U_j \right)^2 \left( U_j + \frac{4}{3} (U_{j+1/2} - U_j) \right). \end{aligned}$$

Similarly, we have

$$\Delta S_j^- = -2 \left( U_j - U_{j-1/2} \right)^2 \left( U_j - \frac{4}{3} (U_j - U_{j-1/2}) \right).$$

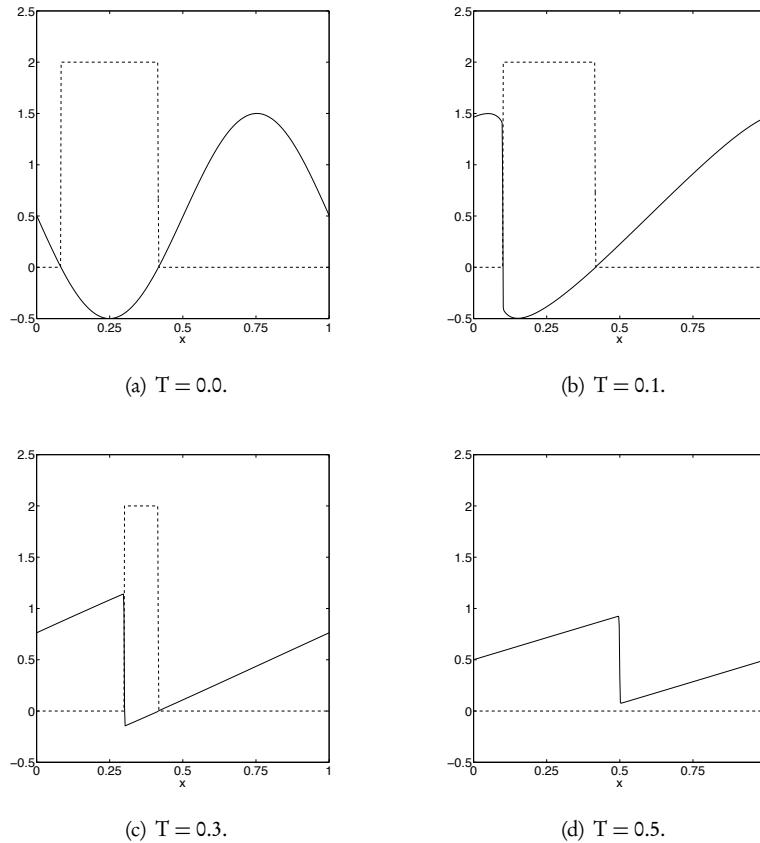
Finding the exact solution for  $U_{j+1/2}$  to guarantee entropy stability requires the solution of this nonlinear set of equations. However, if we assume that  $|U_{j+1/2} - U_j| \leq \frac{4}{3}|U_j|$  and  $|U_j - U_{j-1/2}| \leq \frac{4}{3}|U_j|$ , the situation simplifies as

$$\Delta S_j^+ = \frac{1}{2} \left( \phi_{j+1/2} \Delta^+ U_j \right)^2 U_j, \quad \Delta S_{j+1}^- = -\frac{1}{2} \left( 2 - \phi_{j+1/2} \right)^2 \left( \Delta^+ U_j \right)^2 U_j,$$

which yields the total entropy production

$$\Delta S_j = \frac{1}{2} \left[ \left( \phi_{j+1/2} \Delta^+ U_j \right)^2 - \left( 2 - \phi_{j+1/2} \right)^2 \left( \Delta^+ U_j \right)^2 \right] U_j.$$

To guarantee entropy stability, we must ensure that  $\Delta S_j \leq 0$ .



**Figure 8.6.** Solution to Burgers' equation using a first order accurate entropy stable scheme. The solid line represents the solution  $u_b$ , while the dashed line represents  $\phi$ . Both  $u_b$  and  $\phi$  are computed with  $N = 512$ .

If we first assume that  $U_j > 0$  and second order accuracy,  $\phi_{j+1/2}(1) = 1$ , we recover from (8.23) the condition

$$\phi_{j+1/2} \leq \left| \frac{\Delta^- U_j}{\Delta^+ U_j} \right| = |r_j|.$$

If, on the other hand,  $U_j < 0$ , we assume that  $\phi_{j+3/2} = 1$  and recover the condition

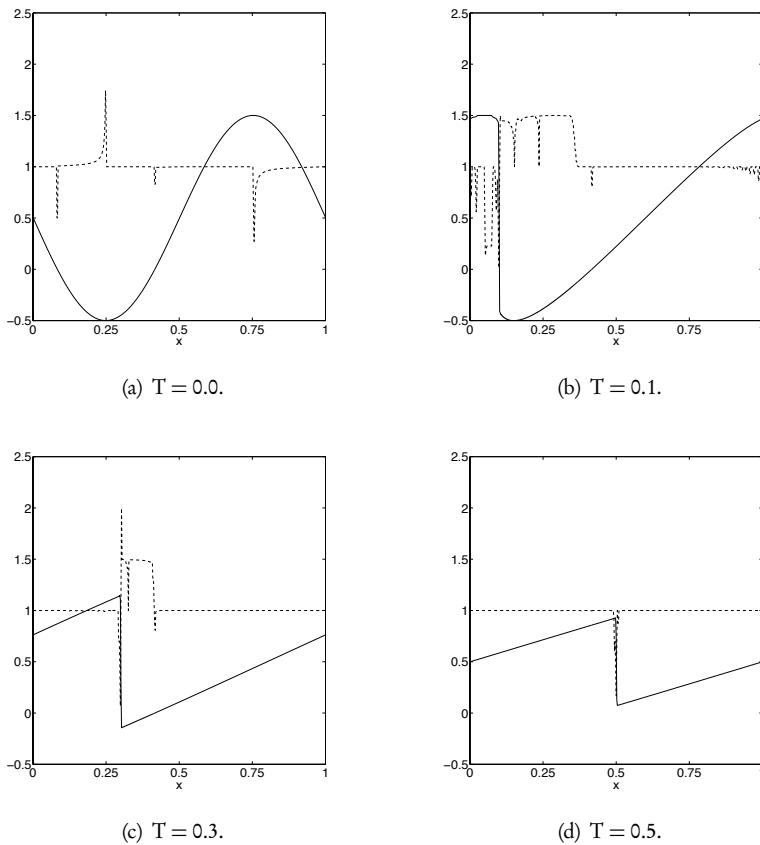
$$\phi_{j+1/2} \geq 2 - |r_j|^{-1}.$$

Recalling that  $\phi_{j+1/2} \leq 1$  for entropy stability, we obtain

$$\phi_{j+1/2} = \begin{cases} \min(1, |r_j|), & (U_j, U_{j+1}) > 0, \\ \max(1, 2 - |r_j|^{-1}), & (U_j, U_{j+1}) < 0. \end{cases}$$

When  $U_j$  and  $U_{j+1}$  have different signs, the scheme reverts to the first order scheme.

In Fig. 8.7 we show the computed solution and the value of  $\phi$  at different times for Burgers' equation. We observe that there are substantial regions of the solution where



**Figure 8.7.** Solution to Burgers' equation using a locally second order accurate entropy stable scheme. The solid line represents the solution,  $u_b$ , while the dashed line is  $\phi$ , both computed with  $N = 512$ .

$\phi = 1$ , indicating local second order accuracy. However, we also see the generation of oscillations in  $\phi$  around local extrema as well as at sonic points. ■

Before we conclude this discussion, let us investigate whether it is, at least formally, possible to derive entropy conservative schemes of arbitrary spatial order. Naturally, if shocks emerge in the solution, dissipation is needed. However, there are several ways to achieve this and satisfy the cell entropy inequality.

Since we focus on the spatial order of accuracy, we consider the semidiscrete form

$$\frac{dU_j}{dt} = -\frac{F_{j+1/2} - F_{j-1/2}}{h},$$

where

$$F_{j+1/2} = F(U_{j-p+1}, \dots, U_{j+p}),$$

i.e.,  $F_{j+1/2}$  depends on  $2p$  values. For consistency, we require that  $F(u, \dots, u) = f(u)$ .

Provided the solution is smooth, we also require that

$$\frac{F_{j+1/2} - F_{j-1/2}}{h} = \frac{\partial f}{\partial x} + \mathcal{O}(h^{2p}),$$

for a scheme of order  $2p$ .

Let us recall the optimal flux

$$F_{j+1/2}^* = \int_0^1 f(U_j + \xi \Delta^+ U_j) d\xi = F^*(U_j, U_{j+1}).$$

Following [19], we assume that we can express the general numerical flux as

$$F_{j+1/2} = \sum_{i=1}^p \alpha_i^p \left( F^*(U_j, U_{j+i}) + \dots + F^*(U_{j-i+1}, U_{j+1}) \right),$$

where  $\alpha_i^p$  are coefficients. It follows that

$$F_{j+1/2} - F_{j-1/2} = \sum_{i=1}^p \alpha_i^p \left( F^*(U_j, U_{j+i}) - F^*(U_{j-i}, U_j) \right).$$

We are now ready to state the answer to our question [19].

**Theorem 8.16.** *Consider a hyperbolic conservation law, solved using a semidiscrete scheme in conservation form. Provided*

$$2 \sum_{i=1}^p i \alpha_i^p = 1, \quad \sum_{i=1}^p i^{2s-1} \alpha_i^p = 0, \quad s \in [2, \dots, p],$$

*the numerical flux*

$$F_{j+1/2} = \sum_{i=1}^p \alpha_i^p \left( F^*(U_j, U_{j+i}) + \dots + F^*(U_{j-i+1}, U_{j+1}) \right),$$

*is consistent and the resulting scheme is of order  $\mathcal{O}(h^{2p})$ . Furthermore, if the numerical entropy flux is defined as*

$$\Psi_{j+1/2} = \sum_{i=1}^p \alpha_i^p \left( \Psi^*(U_j, U_{j+i}) + \dots + \Psi^*(U_{j-i+1}, U_{j+1}) \right),$$

*with  $\Psi^*(U_j, U_{j+i}) = U_j F^*(U_j, U_{j+i})$ , the scheme is entropy conservative.*

**Proof.** Let us first discuss the accuracy. If we assume that  $u = U_j$  is smooth, we have

$$F^*(U_j, U_{j+i}) - F^*(U_{j-i}, U_j) = 2 \sum_{l=0}^p \frac{(i h)^{2l+1}}{(2l+1)!} f^{(2l+1)}(U_j) + \mathcal{O}(h^{2l+1})$$

such that

$$\frac{F_{j+1/2} - F_{j-1/2}}{h} = \frac{2}{h} \sum_{i=1}^p \alpha_i^p \sum_{l=0}^p \frac{(i h)^{2l+1}}{(2l+1)!} f^{(2l+1)}(U_j) + \mathcal{O}(h^{2p}).$$

Consistency requires that

$$\frac{2}{b} \sum_{i=0}^p \alpha_i^p i b f^{(1)}(U_j) = \frac{\partial f}{\partial x}(U_j) \Rightarrow 2 \sum_{i=1}^p i \alpha_i^p = 1.$$

For the remaining terms to vanish we must require

$$\sum_{i=1}^p i^{2s-1} \alpha_i^p = 0, \quad s = 2, \dots, p.$$

Consider

$$\begin{aligned} U_j (F_{j+1/2} - F_{j-1/2}) &= \sum_{i=1}^p \alpha_i^p U_j (F^*(U_j, U_{j+i}) - F^*(U_{j-i}, U_j)) \\ &= \sum_{i=1}^p \alpha_i^p (\Psi^*(U_j, U_{j+i}) - \Psi^*(U_{j-i}, U_j)), \end{aligned}$$

we recover

$$\Psi_{j+1/2} = \sum_{i=1}^p \alpha_i^p (\Psi^*(U_j, U_{j+i}) + \dots + \Psi^*(U_{j-i+1}, U_{j+1})).$$

This completes the proof.  $\square$

Thus, there is no fundamental limitation to the formulation of high-order accurate entropy conservative schemes. However, additional steps are needed to make such schemes effective for solving problems with shocks.

**Example 8.17.** Consider again Burgers' equation and use a numerical flux defined as [31]

$$F_{j+1/2} = \frac{1}{3} \frac{U_{j+1}^3 - U_j^3}{\Delta^+ U_j}.$$

If we assume that the solution is smooth, a Taylor expansion yields

$$\frac{F_{j+1/2} - F_{j-1/2}}{b} = \frac{\partial u^2}{\partial x} + \frac{b^2}{6} \frac{\partial^3 u^2}{\partial x^3},$$

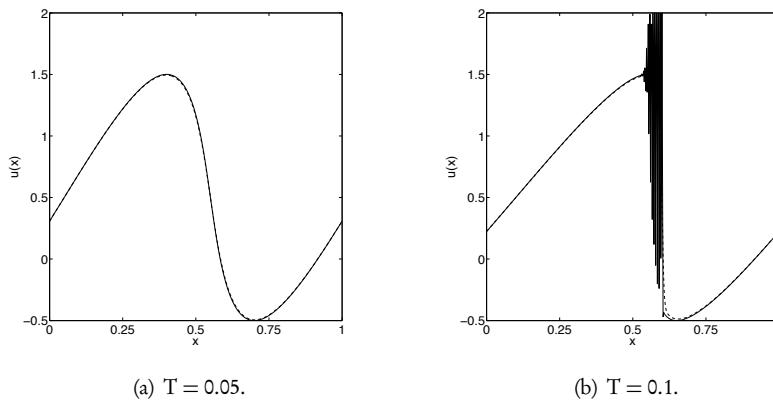
indicating consistency and second order accuracy. Now consider the entropy flux

$$\Psi_{j+1/2} = U_j F_{j+1/2} = \frac{U_j}{3} (U_{j+1}^2 + U_j U_{j+1} + U_j^2).$$

Assuming periodic boundary conditions, it is immediate that

$$\sum_j \frac{\Psi_{j+1/2} - \Psi_{j-1/2}}{b} = 0.$$

Hence, the scheme is entropy conservative.



**Figure 8.8.** Solution to Burgers' equation using an entropy conservative scheme. The solid line represents the solution,  $u$ , while the dashed line represents an approximate reference solution. Here  $N = 512$  grid points are used.

Let us employ this scheme to solve Burgers' equation with the initial condition

$$u(x, 0) = 0.5 + \sin(2\pi x)$$

and periodic boundary conditions. As shown in Fig. 8.8, the computation results in strong oscillations once the shock forms. This is a consequence of the scheme being entropy conservative, eliminating the dissipation in the neighborhood of the shock.

If we alter the scheme as

$$U_j^{n+1} = U_j^n - \frac{k}{h} (F_{j+1/2}^n - F_{j-1/2}^n) + \varepsilon k \frac{U_{j+1}^n - 2U_j^n + U_{j-1}^n}{h^2},$$

effectively considering a vanishing viscosity approach, the results in Fig. 8.9 show a dramatic improvement in the overall quality of the solution. However, we also observe that it comes at the cost of an increased smearing of the shock as time progresses. ■

### 8.3 ▪ The ugly

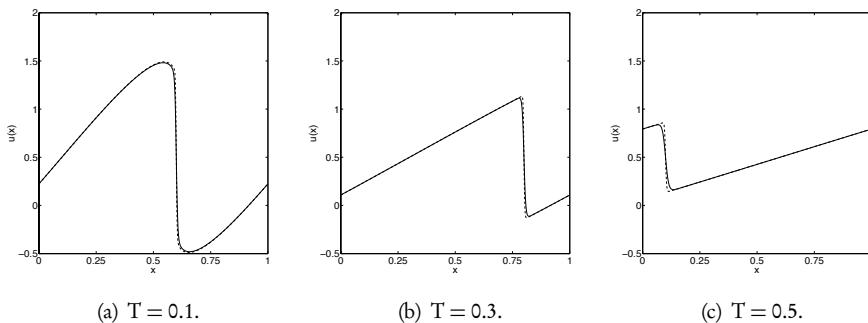
Let us consider the wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [0, 1],$$

subject to periodic boundary conditions and initial conditions

$$u_0(x) = \begin{cases} 1, & 0.25 < x < 0.75, \\ 0, & \text{otherwise.} \end{cases}$$

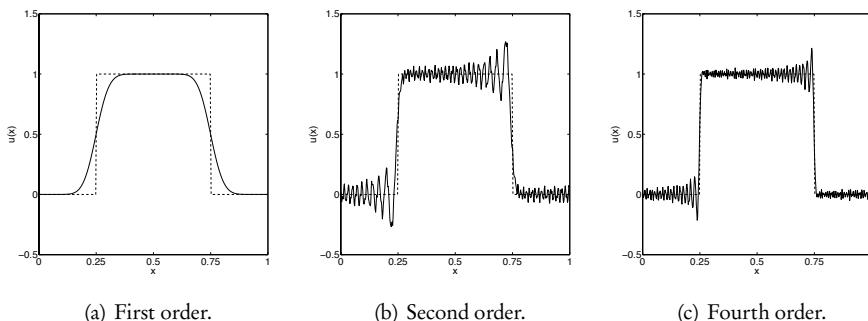
We illustrate the results of a computation, obtained with three different schemes in space, in Fig. 8.10. The temporal integration is carried out in a way that allows us to neglect this as a source of errors. The results allow us to identify a number of concerns,



**Figure 8.9.** Solution to Burgers' equation using the entropy conservative scheme, modified with a vanishing viscosity with  $\varepsilon = 0.005$ . The solid line represents the solution,  $u$ , while the dashed line represents an approximation to the exact solution. Here  $N = 512$  grid points are used.

most notably the strong oscillations emerging with methods of higher than first order. In the result obtained with the first order upwind scheme the oscillations are absent at the expense of severe smearing of the discontinuous solution. This is consistent with the insight gained through the modified equation, (8.4), showing that the error term is a second order operator. The maximum principle of the heat equation [6] guarantees that no oscillations emerge. For the second and fourth order schemes, the modified equations (8.5) and (8.6) are of a dispersive nature, and no maximum principle applies. The observation that the generated wave fronts appear to trail the discontinuity is consistent with the phase error analysis in subsection 8.1.1.

Revisiting Fig. 8.10 we also observe that, in spite of the strong oscillations which result in a visually unsatisfactory solution, the accuracy of the solution as measured by the propagation and steepness of the nonsmooth solution clearly improves as the order of the scheme increases. To gain some additional insight into the nature of the oscillations, let us consider an example [18].



**Figure 8.10.** Solution to the linear wave equation, subject to a discontinuous initial condition. The results were obtained by an upwind scheme (left), a central second order method (middle), and a fourth order central method (right). In all cases,  $N = 512$  and the solution is shown at  $T = 1$ . The dashed line illustrates the exact solution.

**Example 8.18.** Let us consider the linear wave equation

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

subject to appropriate boundary and initial conditions. We use a linear scheme of the form

$$U_j^{n+1} = \sum_{l=-p-1}^q c_l(\lambda) U_{j+l}^n.$$

For the scheme to be  $\mathcal{O}(h)$  accurate, we recall from Theorem 4.11 that we must require

$$\sum_{l=-p-1}^q c_l(\lambda) = 1, \quad \sum_{l=-p-1}^q l c_l(\lambda) = -\lambda.$$

A simple solution to this is

$$c_{-1}(\lambda) = \frac{1+\lambda}{2}, \quad c_1(\lambda) = \frac{1-\lambda}{2},$$

and  $c_l(\lambda) = 0$  otherwise.

Let us now consider a nonsmooth initial condition  $u(x, 0) = H(x)$ , where  $H(x)$  is the Heaviside function, and assume that  $x_0 = 0$  is a grid point. If we apply the scheme for one time step we recover

$$U_j^1 = \begin{cases} 0, & j \leq -1, \\ \frac{1}{2}(1-\lambda), & j = 0, 1, \\ 1, & j \geq 2. \end{cases}$$

Since  $|\lambda| \leq 1$  for stability it is clear that  $0 \leq U_j^n \leq 1$ . Hence, no oscillations are generated. Since this scheme is the monotone Lax-Friedrichs scheme, this is as expected.

Let us now consider a second order scheme for which the conditions in Theorem 4.11 are

$$\sum_{l=-p-1}^q c_l(\lambda) = 1, \quad \sum_{l=-p-1}^q l c_l(\lambda) = -\lambda, \quad \sum_{l=-p-1}^q l^2 c_l(\lambda) = \lambda^2.$$

A particular solution to this is

$$c_{-1}(\lambda) = \frac{\lambda(\lambda+1)}{2}, \quad c_0(\lambda) = 1 - \lambda^2, \quad c_1(\lambda) = \frac{\lambda(\lambda-1)}{2},$$

and  $c_l(\lambda) = 0$  otherwise. We recognize this as the Lax-Wendroff scheme. Considering the solution after just one time step, we recover

$$U_j^1 = \frac{1}{2} \begin{cases} 0, & j \leq -1, \\ \lambda(\lambda-1), & j = 0, \\ (-\lambda^2 - \lambda + 2), & j = 1, \\ 2, & j \geq 2. \end{cases}$$

If  $\lambda > 0$ , it is clear that  $U_0^1$  is negative. If, however,  $\lambda < 0$ , we see that  $U_1^1 > 1$ . Hence, an oscillation is created.

This is fully consistent with Godunov's theorem. Using an approach similar to the above, it is possible to prove that a nonoscillatory second order linear scheme cannot be constructed [18]. ■

In what follows we seek to identify the source of the oscillations and develop an understanding of whether the desirable properties of high-order methods for wave-dominated problems, discussed at length in section 8.1, are maintained for problems with discontinuous solutions. Finally, we discuss techniques to diminish or entirely remove the oscillations.

### 8.3.1 • The Gibbs phenomenon

To cast some light on the source of the oscillations, we need to develop an understanding of the behavior of high-order Lagrange interpolation when applied to discontinuous solutions.

Let  $u$  be a function defined on  $[0, 2\pi]$  and let  $u_b$  be an approximation in the form

$$u_b(x) = \sum_{l=0}^{2^n} u_l C_{l,b}(x),$$

where

$$C_{l,b}(x) = C\left(\frac{x - lh}{b}\right), \quad C(y) = \text{sinc}(\pi y) = \frac{\sin(\pi y)}{\pi y}, \quad b = \frac{2\pi}{2^n}.$$

One recognizes  $C(x)$  as the Whittaker or Shannon Cardinal function and  $\text{sinc}(x)$  as the Sinc-function [26, 27]. Among many other interesting properties, it is an  $L^2$ -complete basis.

We observe that  $C_{l,b}(x_l) = 1$  such that  $u_l = u(x_l)$ , i.e.,  $u_b$  is an interpolation. In fact, we can view it as the limit case of a high-order Lagrange interpolation—a connection that we shall make clear shortly.

The Whittaker Cardinal function plays a central role in signal analysis, as it allows for the exact representation of a band limited signal. To realize this, recall that  $b = \frac{2\pi}{2^n}$  and use the uniqueness of the interpolation to express the Cardinal function as

$$\begin{aligned} C_{l,b}(x) &= \frac{1}{2^n} \int_{-2^{n-1}}^{2^{n-1}} \exp(i(x_l - x)t) dt \\ &= \frac{b}{2\pi} \int_{-\pi/b}^{\pi/b} \exp(ilht) \exp(-ixt) dt = \frac{1}{2\pi} \int_{\mathbb{R}} F(t) \exp(-ixt) dt, \end{aligned} \tag{8.27}$$

where we have defined the band limited function

$$F(t) = \begin{cases} b \exp(ilht) & |t| \leq \frac{\pi}{b}, \\ 0 & |t| > \frac{\pi}{b}. \end{cases}$$

Hence,  $C_{l,b}$  is the Fourier transform of  $F$ . Unfortunately, this result is not of immediate value, since a discontinuous solution is, by its very nature, not band limited. However, by expressing the Cardinal function in this way, it is easy to prove that

$$\int_{\mathbb{R}} C_{i,b}(x) C_{j,b}(x) dx = h \delta_{ij}. \tag{8.28}$$

Hence, the Cardinal functions are all mutually orthogonal and we can express  $u_l = u(x_l)$  as

$$u_l = u(x_l) = \frac{1}{b} \int_{\mathbb{R}} u(x) C_{l,b}(x) dx.$$

We can rewrite the truncated expansion

$$u_b(x) = \sum_{l=0}^{2^n} u_l C_{l,b}(x) = \frac{1}{b} \int_{\mathbb{R}} u(t) \sum_{l=0}^{2^n} C_{l,b}(t) C_{l,b}(x) dt = \frac{1}{b} \int_{\mathbb{R}} u(t) K_b(x, t) dt,$$

where we introduce the kernel

$$K_b(x, t) = \sum_{l=0}^{2^n} C_{l,b}(t) C_{l,b}(x).$$

The kernel function is illustrated in Fig. 8.11 and we observe that it is essentially translation invariant. Hence, we can approximate the truncated expansion as

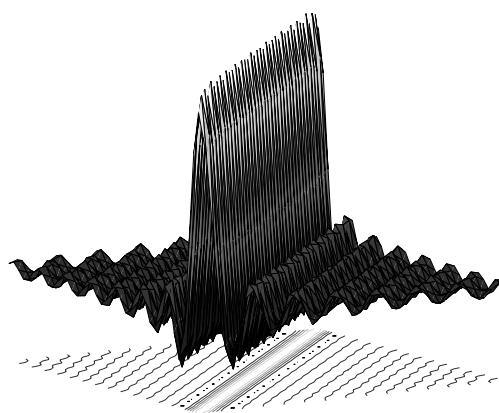
$$u_b(x) = \frac{1}{b} \int_{\mathbb{R}} u(x-t) K_b^*(t) dt, \quad K^*(t) = K_b(t, t). \quad (8.29)$$

Since substantial contributions to the integral originate only in the neighborhood of  $t \simeq 0$ , see Fig. 8.11, we consider

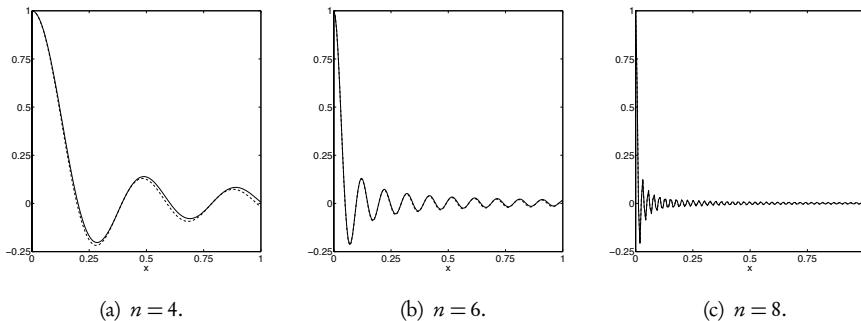
$$u_b(x) \simeq \frac{1}{b} \int_{-\varepsilon}^{\varepsilon} u(x-t) K_b^*(t) dt.$$

In the neighborhood of  $t = 0$ , the kernel is well-approximated by the periodic Dirichlet kernel defined as

$$D_b(t) = \frac{\sin(2^n t)}{2^{n+1} \sin(t/2)},$$



**Figure 8.11.** Illustration of the kernel  $K_b$  associated with the Whittaker Cardinal function for  $n = 4$ .



**Figure 8.12.** Kernel (solid line) associated with the Whittaker Cardinal function for increasing values of  $n$  compared with the periodic Dirichlet kernel (dashed line) in the neighborhood of  $t = 0$ .

as illustrated in Fig. 8.12. In fact, as  $n$  increases, the two kernels become indistinguishable.

Since  $D_h(t)$  is even, we recover

$$u_h(x) \simeq \frac{1}{h} \int_{-\varepsilon}^{\varepsilon} u(x-t) D_h(t) dt = \frac{2}{h} [u(x^+) + u(x^-)] \int_0^{\varepsilon} \frac{\sin(2^n t)}{2^{n+1} \sin(t/2)} dt,$$

where we use the notation  $u(x^\pm) = \lim_{\varepsilon \rightarrow 0^+} u(x \pm \varepsilon)$ . Realizing that

$$\frac{2}{h} \int_0^{\varepsilon} \frac{\sin(2^n t)}{2^{n+1} \sin(t/2)} dt = \frac{1}{\pi} \int_0^{2^n \varepsilon} \frac{\sin s}{s} ds \underset{n \rightarrow \infty}{\simeq} \int_0^\infty \frac{\sin \pi s}{\pi s} ds = \frac{1}{2},$$

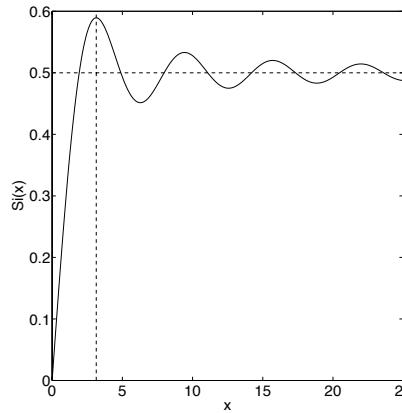
we recover

$$\lim_{n \rightarrow \infty} u_h(x) = \frac{1}{2} [u(x^+) + u(x^-)].$$

Hence, the expansion converges pointwise to the average of the left and the right limits of the function.

To understand the nature of the oscillations, we need to take this analysis one step further and consider (8.29) as

$$\begin{aligned} u_h(x + 2^{-n} z) &\simeq \frac{u(x^+)}{\pi} \int_{-\infty}^{2^{-n} z} \frac{\sin(2^n t)}{t} dt + \frac{u(x^-)}{\pi} \int_{2^{-n} z}^{\infty} \frac{\sin(2^n t)}{t} dt \\ &= u(x^+) \int_{-\infty}^{z/\pi} \frac{\sin(\pi t)}{\pi t} dt + u(x^-) \int_{z/\pi}^{\infty} \frac{\sin(\pi t)}{\pi t} dt \\ &= u(x^+) \left( \int_{-\infty}^0 \frac{\sin(\pi t)}{\pi t} dt + \int_0^{z/\pi} \frac{\sin(\pi t)}{\pi t} dt \right) \\ &\quad + u(x^-) \left( \int_0^{\infty} \frac{\sin(\pi t)}{\pi t} dt - \int_0^{z/\pi} \frac{\sin(\pi t)}{\pi t} dt \right) \\ &= \frac{1}{2} [u(x^+) + u(x^-)] + [u(x^+) - u(x^-)] \int_0^{z/\pi} \frac{\sin(\pi t)}{\pi t} dt. \end{aligned}$$



**Figure 8.13.** Illustration of the Sine integral, which plays an essential role in the understanding of the oscillations in high-order methods.

If  $z$  approaches  $x$  like  $\mathcal{O}(h)$ , the second term will not vanish and we have nonuniform convergence. To measure the error, consider

$$Si(z) = \int_0^{z/\pi} \frac{\sin(\pi t)}{\pi t} dt,$$

which is the scaled Sine integral, illustrated in Fig. 8.13.

The Sine integral takes the maximum value for  $z = \pi$ , with  $Si(\pi) = 0.58949\dots$  If we measure the size of the overshoot, already observed in Fig. 8.10, we find that

$$u_b(x^-) - u(x^-) \simeq 1.08949(u(x^+) - u(x^-)),$$

i.e., an overshoot of about 9% of the jump of the function. This value will be the result at  $x \pm \frac{h}{2}$ , with the point of discontinuity at  $x$ . Since the Sinc function is the generalization of the Fourier series to an unbounded domain, it comes as no surprise that we recover the classic results from Fourier analysis in this case. For more details we refer to [11, 12].

While this discussion helps understand the source of the oscillations when approximating a discontinuous function with the Whittaker Cardinal function, we still need to tie this discussion to the observations in Fig. 8.10, where the approximation is based on a finite difference scheme.

To accomplish this, recall the interpolating Lagrange polynomial

$$\ell_j(x) = \prod_{\substack{i=-p \\ i \neq j}}^q \frac{x - x_i}{x_j - x_i}, \quad (8.30)$$

where  $x_j$  represents the  $q + p + 1$  nodes. The polynomial is defined by  $\ell_j(x_i) = \delta_{ij}$  such that

$$u_b(x) = \sum_{j=-p}^q u(x_j) \ell_j(x)$$

is an interpolation. As discussed previously, by differentiating the Lagrange polynomial and evaluating it at grid points, the standard stencils are recovered.

To establish a connection between the Lagrange polynomial and the Whittaker Cardinal function, we take  $y = x/b$ , assume  $p = q$ , i.e., a centered stencil, and express the Lagrange polynomial as [32]

$$\ell_j(x) = (-1)^{q-j} \binom{y+q}{j+q} \binom{y-j-1}{q-j}.$$

This follows directly by rewriting (8.30) using binomials. We continue as

$$\ell_j(x) = (-1)^{q-j} \binom{y+q}{j+q} \binom{y-j}{q-j} \frac{1}{y-j} = (-1)^{q-j} \binom{y+q}{2q+1} \binom{2q}{q-j} \frac{2n+1}{y-j}.$$

With a bit of algebraic manipulations, one recovers

$$\ell_j(x) = C(q, j, y) \operatorname{sinc}\left(\frac{x - x_j}{b}\right), \quad C(q, j, y) = \frac{(q+y)!(q-y)!}{(q+j)!(q-j)!}. \quad (8.31)$$

We realize that  $C(q, j, x_i) = 1$  so that  $\ell_j(x_i) = \delta_{ij}$  as expected. We also recognize that  $C(n, j, y) = 0$  when  $|y|, |j| > n$ , naturally truncating the stencil width at  $2q+1$ . Using the asymptotic expansion [2]

$$C(q, j, y) \simeq \exp\left(-\frac{j^2 - y^2}{q}\right),$$

it is clear that the Lagrange basis is directly related to the Whittaker Cardinal function through a Gaussian modification with a variance that scales with  $\sqrt{q}$ .

As a final validation of the close connection between the Lagrange basis and the Whittaker Cardinal function, consider

$$u_b(x) = \sum_{j=-q}^q u(x_j) \ell_j(x).$$

The derivative at the grid points of  $u_b(x)$  is given as

$$\left. \frac{du_b}{dx} \right|_{x_i} = \sum_{j=-q}^q u(x_j) \ell'_j(x_i).$$

We now exploit relation (8.31) to obtain

$$\ell'_j(x_i) = \frac{(-1)^{(i-j)}}{b(i-j)} \frac{(q+i)!(q-i)!}{(q+j)!(q-j)!},$$

provided  $i \neq j$  and  $\ell'_j(x_j) = 0$ . These are exactly the weights for the finite difference formulas of order  $2q$  and, if  $i = 0$ , yield the central difference formulas in (8.3).

This closes the loop and allows us to interpret the Gibbs oscillations, exposed clearly in the Whittaker Cardinal function but also arising in finite difference methods based on Lagrange interpolations, as being driven by the same mechanics, namely an inability to accurately approximate a discontinuous function with a high-order global function. The consequences are the artificial oscillations, a reduction of the accuracy globally, and a loss of uniform convergence, all characteristics of the Gibbs phenomenon.

### 8.3.2 • Does it matter?

While these results appear discouraging, the results in Fig. 8.10 suggest a more complex picture. Although the pointwise accuracy is poor, even as the order increases, the accuracy in the phase of the scheme appears to be intact. This suggests that high-order accurate information is hidden in the computed solution.

To cast some light on this, consider solutions expressed as

$$u_b(x) = \sum_l u(x_l) C_{l,b}(x).$$

Our first goal is to understand the accuracy of this expansion as it relates to the smoothness of the solution,  $u(x)$ , being approximated. The result is stated in [27] in the following theorem.

**Theorem 8.19.** *Assume that  $u \in C^q(\mathbb{R})$  has decay*

$$\forall l = 0, \dots, q : |u^{(l-1)}(x)| \rightarrow 0 \quad \text{for } |x| \rightarrow \infty.$$

*Then*

$$\|u - u_b\|_2 \leq C b^q \|u^{(q)}\|_2.$$

*Proof.* Express  $u$  through its Fourier series

$$u(x) = \int_{\mathbb{R}} \hat{u}(t) e^{ixt} dt, \quad \hat{u}(t) = \frac{1}{2\pi} \int_{\mathbb{R}} u(x) e^{-ixt} dx,$$

and recall (8.27) to recover

$$\begin{aligned} u(x) - u_b(x) &= \int_{\mathbb{R}} \hat{u}(t) e^{ixt} dt - \sum_l \left( \int_{\mathbb{R}} \hat{u}(t) e^{ilbt} dt \right) C_{l,b}(x) \\ &= \int_{\mathbb{R}} \hat{u}(t) \left[ e^{ixt} - \sum_l e^{ilbt} C_{l,b}(x) \right] dt. \end{aligned}$$

Since  $g(x) = e^{ixt}$  is periodic, we express it as

$$g(x) = \sum_r \hat{g}_r e^{irbx}, \quad x \in [-\pi/b, \pi/b],$$

where

$$\hat{g}_r = \frac{b}{2\pi} \int_{-\pi/b}^{\pi/b} e^{ixt} e^{-irbx} dx = C_{r,b}(t),$$

by (8.27). By periodicity of  $g$  we recover

$$\sum_r C_{r,b}(t) e^{irbx} = \begin{cases} \exp(it(x - 2m\pi/b)), & (2m-1)\frac{\pi}{b} < x < (2m+1)\frac{\pi}{b}, \\ \cos(\pi t/b), & x = (2m \pm 1)\frac{\pi}{b}. \end{cases} \quad (8.32)$$

The second term results from  $C_{r,b}(t)$  being nonperiodic and the Fourier expansion converges to the average value.

With this, we recover

$$u(x) - u_b(x) = \int_{|x| > \pi/b} \hat{u}(t) \left[ e^{ixt} - \sum_l e^{ilbt} C_{l,b}(x) \right] dt.$$

Since both terms in the sum are bounded by one, we recover

$$\|u(x) - u_b(x)\|_2^2 \leq 4 \int_{x > \pi/b} |\hat{u}(t)|^2 dt. \quad (8.33)$$

Estimation of the Fourier coefficients can be done directly as

$$|\hat{u}(t)| = \frac{1}{2\pi} \left| \int_R u(x) e^{-ixt} dx \right| \leq \frac{1}{2\pi} |t|^{-q} \left| \int_R u^{(q)}(x) e^{-ixt} dx \right|,$$

where the boundary terms vanish by the decay assumption. Inserting this into (8.33) yields

$$\|u(x) - u_b(x)\|_2^2 \leq 4 \int_{x > \pi/b} |\hat{u}(t)|^2 dt \leq C \left( \frac{\pi}{b} \right)^{-2q} \|u^{(q)}\|_2^2,$$

and the result follows.  $\square$

Now consider the linear conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial a(x)u}{\partial x} = \frac{\partial u}{\partial t} + \mathcal{L}u = 0,$$

with a suitable initial condition  $u_0(x)$ . As for the situation in Fig. 8.10, we allow the initial condition to be nonsmooth, but the variable coefficient  $a(x)$  is assumed to be smooth.

We seek solutions of the form

$$u_b(x, t) = \sum_l u(x_l, t) C_{l,b}(x),$$

and require that the equation be satisfied in a Galerkin sense, i.e.,

$$\left( \frac{\partial u_b}{\partial t} + \mathcal{L}u_b, C_{i,b}(x) \right)_R = 0, |i| \leq N,$$

where we have defined the inner product

$$(u, v)_R = \int_R u \bar{v} dx.$$

At first glance, it could appear that this scheme is different from what we have considered previously. However, recalling the orthogonality (8.28), one quickly realizes that the schemes are indeed identical.

Let us also consider the adjoint problem

$$\frac{\partial v}{\partial t} - a(x) \frac{\partial u}{\partial x} = \frac{\partial v}{\partial t} - \mathcal{L}^*v = 0,$$

subject to a smooth initial condition,  $v_0$ . We also require that this be satisfied in a Galerkin sense.

By construction of the two problems we have that

$$\frac{d}{dt} (u(x, t), v(x, t))_R = 0$$

such that

$$(u(x, t), v(x, t))_R = (u_0(x), v_0(x))_R.$$

Since the scheme is a Galerkin scheme, we have that

$$\left( \frac{\partial u_b}{\partial t} + \mathcal{L}u_b, v_b \right)_R = 0,$$

and likewise for the adjoint problem to recover

$$\frac{d}{dt} (u_b(x, t), v_b(x, t))_R = 0.$$

Since the initial condition  $u_0$  satisfies

$$(u_0(x) - u_b(x, 0), v_b(x, 0))_R = 0,$$

by Galerkin orthogonality, we have

$$|(u_0(x), v_b(x, 0) - v_0(x))_R| \leq C \|u_0\|_2 \|v_b(x, 0) - v_0(x)\|_2 \leq C \|u_0\|_2 b^q \|v_0^{(q)}\|_2,$$

by Theorem 8.19. We recover

$$\begin{aligned} & |(u_b(x, 0), v_b(x, 0))_R - (u_0(x), v_0(x))_R| \\ &= |(u_b(x, 0) - u_0(x), v_b(x, 0))_R + (u_0(x), v_b(x, 0) - v_0(x))_R| \leq C \|u_0\|_2 b^q \|v_0^{(q)}\|_2. \end{aligned}$$

Since  $u$  and  $v_0(x)$  by assumption are smooth, we can replace  $v_b$  by the exact solution  $v$  to recover

$$\begin{aligned} & |(u_b(x, t), v_b(x, t))_R - (u(x, t), v(x, t))_R| = |(u_b(x, 0), v_b(x, 0))_R - (u_0(x), v_0(x))_R| \\ & \leq C b^q \|u_0\|_2 \|v_0^{(q)}\|_2. \end{aligned}$$

This result, first obtained in [1] in the context of a Fourier spectral method, shows that even though the initial conditions are discontinuous, the high-order accurate information is not lost. However, the accuracy is not obtained in a pointwise sense but, rather in the sense of moments. This helps us understand the results in Fig. 8.10 where the visual accuracy, measured by the location of the shock, clearly improves with the order of the scheme, even as the pointwise accuracy remains poor.

These results can be extended to include accuracy of moments for problems with nonsmooth forcing functions, and nonsmooth coefficients [36]. We return to this discussion in more detail in subsection 12.2.1.

The extension of such analysis to general nonlinear conservation laws remains an open problem. In [5] it is shown that only first order pointwise error is possible after a shock has passed. However, the accuracy of the moments is not known and many examples demonstrate that high-order accurate information is retained [4, 25].

Hence, the Gibbs oscillations, while clearly destroying the pointwise accuracy, may not destroy the high-order accuracy of the solution itself. A question that remains open for the general case is how to extract the accurate solution in a simple and robust manner. We return to this question in section 13.4.

### 8.3.3 • What to do if it does matter

The discussion in subsections 8.3.1 and 8.3.2 yields confidence that the Gibbs oscillations may not destroy the accuracy of the solution. However, there are numerous situations where such oscillations are unacceptable. For instance, if the oscillations arise in the approximation of a density or a pressure, causing them to become negative and, thus unphysical, the progression of the simulation is in question. Another example is a situation where the solution being approximated is a temperature that is slightly subcritical, but an artificial oscillation may cause an ignition as a purely numerical effect. In cases like these, the oscillations must be eliminated.

The seemingly simple question of how to formulate and analyze high-order accurate methods that do not cause artificial numerical oscillations is the key challenge ahead of us and is perhaps the single most important motivation for the continued development of numerical methods for conservation laws. Before entering into this discussion which, as a consequence of Godunov's theorem, will take us into nonlinear schemes, let us briefly discuss the use of filtering as a means to add dissipation to an existing scheme.

The classic idea of filtering originates in signal processing and is associated with the specific modification of Fourier components of a signal. For instance, by selectively damping high-frequency Fourier components in the signal, the signal becomes less noisy. This is known as a low pass filter.

In the current context, we are not restricted to periodic problems but we shall use a similar approach and seek operators of the form [34]

$$\Omega = \sum_{l=-L}^L \omega_l E_l,$$

where  $\omega_l$  are the filter coefficients that need to be determined.

To restrict ourselves to filters that modify the amplitude, the filter must be symmetric, i.e.,  $\omega_l = \omega_{-l}$ . Hence, we seek a filter of the form

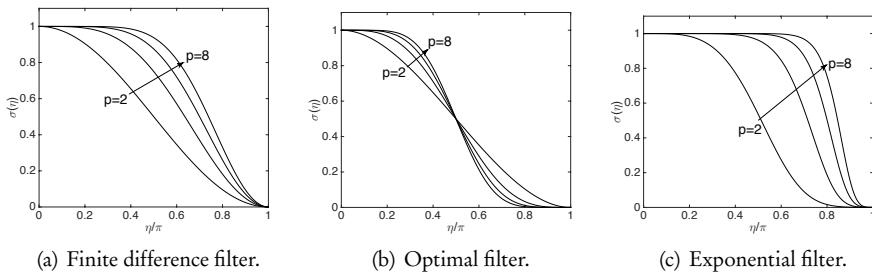
$$\Omega e^{ilx_j} = \left( \omega_0 + 2 \sum_{m=1}^M \omega_m \cos(lmh) \right) e^{ilx_j} = \sigma(\eta) e^{ilx_j}.$$

We recall that  $\eta = lh = 2\pi p^{-1}$ , where  $p$  is the number of points per wavelength, i.e.,  $0 \leq \eta \leq \pi$  to ensure at least two points per wavelength.

The design task is to specify a filter function,  $\sigma(\eta)$ , that achieves the desired damping. The properties of the filter function are specified in the definition of a filter of order  $p$  in [33].

**Definition 8.20.** *The  $p$ th-order filter ( $p > 1$ ) is defined by the filter function,  $\sigma(\eta) \in C^p : R^+ \rightarrow [0, 1]$ , with the following properties:*

$$\sigma(\eta) : \begin{cases} \sigma^{(s)}(0) = 0, & s = 0, \dots, p-1, \\ \sigma(\eta) = 0, & \eta \geq \pi, \\ \sigma^{(s)}(\pi) = 0, & s = 1, \dots, p-1. \end{cases}$$



**Figure 8.14.** Filter functions for various types of filters,  $\sigma(\eta)$ , of order  $p$ .

It should be noted that the last condition for the filter is a technical condition, needed in the analysis of the accuracy of the filter in [33] and is generally believed not to be essential. We shall revisit this discussion in more detail in subsection 13.4.1.

We now have the tools to design filters with the desired properties in two different ways. If we first consider the filter function  $\sigma(\eta)$  and use the definition of the filter, we obtain a set of algebraic equations for the unknowns  $\omega_l$ , i.e.,

$$\sigma(0) = 1 : \sum_{m=0}^M \omega_m = 1, \quad \sigma(\pi) = 0 : \sum_{m=0}^M (-1)^m \omega_m = 0.$$

The remaining  $M - 1$  conditions can be found by requiring additional smoothness at  $lh = 0$  as

$$\sigma^{(s)}(0) = 0, s = 1, \dots, M-1 : \sum_{m=0}^M l^{2m} \omega_m = 0.$$

Solving this algebraic system yields a filter of order  $2L$ . For the first few orders, the corresponding coefficients are given in Table 8.2. Figure 8.14 illustrates filter functions of different types and orders, highlighting the decay as  $\eta$  approaches  $\pi$ .

A different approach is to realize that  $\sigma$  is expressed as a cosine series. Hence, the filter coefficients can be obtained directly by

$$\omega_l = \frac{1}{\pi} \int_0^\pi \sigma(\eta) \cos(l\eta) d\eta,$$

where  $\sigma$  satisfies the conditions in Definition 8.20.

**Table 8.2.** Filter coefficients for filters of order  $2M$ .

	$M = 1$	$M = 2$	$M = 3$	$M = 4$	$M = 5$	$M = 6$
$\omega_0$	0.500000	0.625000	0.687499	0.726562	0.753906	0.774414
$\omega_1$	0.250000	0.250000	0.234375	0.218750	0.205078	0.193359
$\omega_2$		-0.062500	-0.093750	-0.109375	-0.117188	-0.120850
$\omega_3$			0.015625	0.031250	0.043945	0.053711
$\omega_4$				-0.003906	-0.009766	-0.016113
$\omega_5$					0.000977	0.002930
$\omega_6$						-0.000244

In FilterFD.m we show how to compute the filter coefficients for a filter of order  $p$  using a stencil of  $2M + 1$  grid points.

**Script 8.1.** *FilterFD.m: Routine to compute filter coefficients for a filter of order  $p$  using a stencil of  $2M + 1$  points. When computing the filter function, one should take  $p < M$ .*

---

```
function [fc] = filter(M,p)
% function [f] = filter(M,p)
% Purpose: Compute filter coefficients 0:M for filter of order p
% and width 2M+1.
fc = zeros(M+1,1);

% Exponential filter (p even)
alpha = 10;
f = @(x,n) exp(-alpha*(x/pi).^(p)).*cos(n*x);

% Optimal filter of order p
% f = @(x,n) (1-betainc(x/pi,p,p)).*cos(n*x);

for m=0:M
    fc(m+1) = integral(@(x) f(x,m),0,pi)/pi;
end
return
```

---

Designing a polynomial filter by strictly following the definition of the filter of order  $p$  in Definition 8.20 yields the optimal filter function [33].

$$\sigma(\eta) = 1 - \frac{\Gamma(2p)}{\Gamma(p)^2} \int_0^\eta [t(1-t)]^{p-1} dt,$$

where  $\Gamma(x)$  is the Gamma function. The filter function, illustrated in Fig. 8.14 for different orders  $p$  is very smooth but also eliminates a substantial amount of the energy for  $\eta > \pi/2$ .

An alternative and widely used filter, in particular for very high order methods, is the exponential filter of order  $p$ , given as

$$\sigma(\eta) = \exp\left(-\alpha\left(\frac{\eta}{\pi}\right)^p\right).$$

It is clear that  $\sigma(\pi)$  is not zero. However, by choosing  $\alpha$  sufficiently large,  $\sigma(\pi)$  can be made arbitrarily small, e.g., if we require that  $\sigma(\pi) = \varepsilon$ , then  $\alpha = -\ln \varepsilon$ . The illustration of the filter in Fig. 8.14 shows the strong damping of the values for  $\eta$  approaching  $\pi$ , whereas the amplitude is left essentially unchanged for small values of  $\eta$ .

To understand the impact of the filter, write the action of the filter as

$$\Omega u_j = \sum_{m=-M}^M \omega_m u_{j+m} \simeq \int_{-M}^M \omega(m) u(x_j + hm) dm,$$

where

$$\omega(m) = \frac{1}{c_m \pi} \int_0^\pi \sigma(\eta) \cos\left(\eta \frac{m}{M}\right) d\eta.$$

Here  $c_0 = 2$ , and  $c_m = 1$  otherwise. Hence, the action of the filter is simply a convolution with the cosine transform of the filter function.

One way to understand the impact of the filter is to consider the exponential filter,  $\sigma(\eta) = \exp(-\alpha\eta^2)$ . In this case, we recover

$$\begin{aligned}\omega(m) &= \frac{1}{c_m \pi} \int_0^\pi \exp(-\alpha\eta^2) \cos\left(\eta \frac{m}{M}\right) d\eta \simeq \frac{1}{c_m \pi} \int_0^\infty \exp(-\alpha\eta^2) \cos\left(\eta \frac{m}{M}\right) d\eta \\ &= \frac{1}{2c_m \sqrt{\pi\alpha}} \exp\left(-\frac{m^2}{4\alpha M^2}\right).\end{aligned}$$

Hence, the solution is being convolved with a kernel, similar to that of the heat equation [6]. As a consequence we expect the action of the filter to be similar to the dissipation associated with the second order operator.

Another way to realize this is to consider the equation

$$\frac{\partial u}{\partial t} = \beta(-1)^{p/2+1} \frac{\partial^p u}{\partial x^p}, \quad (8.34)$$

where  $\beta$  is a positive constant. For  $p = 2$  we recover the standard diffusive problem. To keep things simple, assume that  $u(x, t)$  is  $2\pi$ -periodic in space, which allows the representation of the solution as a Fourier series:

$$u(x, t) = \sum_{|l| \leq N} \hat{u}_l(t) \exp(ilx), \quad \hat{u}_l(t) = \frac{1}{2\pi} \int_0^{2\pi} u(x, t) \exp(-ilx) dx.$$

If we insert this into (8.34) and construct a Galerkin scheme, we recover the equations

$$\forall |l| \leq N : \frac{d\hat{u}_l}{dt} = -\beta l^p \hat{u}_l.$$

Integrating over  $[t, t+k]$  yields

$$\forall |l| \leq N : \hat{u}_l(t+k) = \exp\left(-\gamma \left(\frac{l}{N}\right)^p\right) \hat{u}_l(t), \quad \gamma = \beta N^p k.$$

Hence, the action of the filter is equivalent to the dissipation associated with a high-order differential equation of order  $p$ , applied over one time step. This clarifies why  $p$  should be even to ensure a dissipative effect.

To finally connect this to the traditional way that dissipation is added, consider

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = \beta(-1)^{p/2+1} \frac{\partial^p u}{\partial x^p}.$$

We recognize that filtering is basically equivalent to solving the modified conservation law in a two step fashion as

$$U_j^* = U_j^n - \frac{k}{h} (F_{j+1/2}^n - F_{j-1/2}^n), \quad U_j^{n+1} = \Omega U_j^*.$$

Due to the maximum principle of the heat equation, applying the second order filter is guaranteed to eliminate oscillations. Unfortunately, this is not true for higher-order filters.

**Example 8.21.** Consider the wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [0, 1],$$

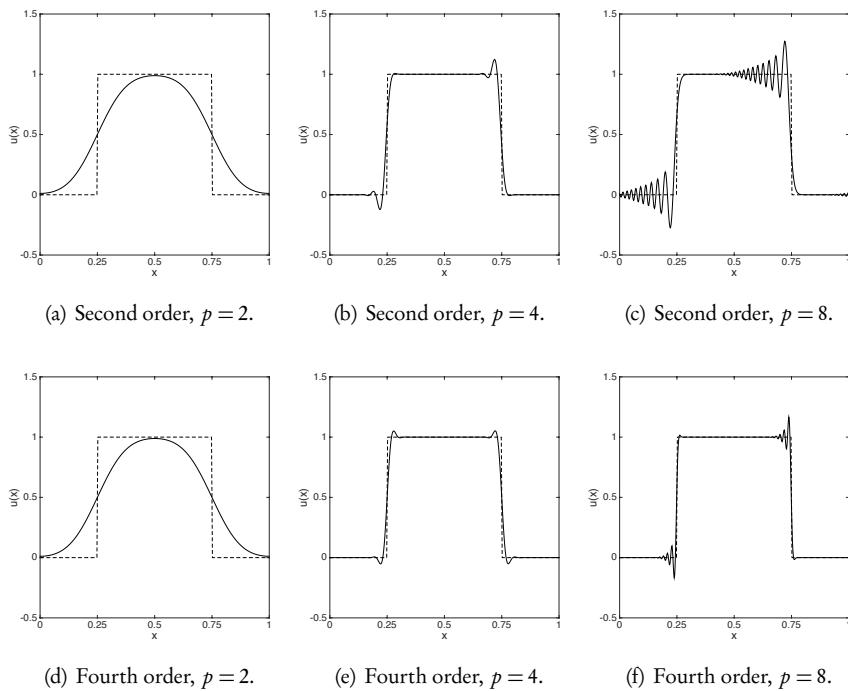
subject to periodic boundary conditions and the initial condition

$$u_0(x) = \begin{cases} 1, & 0.25 < x < 0.75, \\ 0, & \text{otherwise.} \end{cases}$$

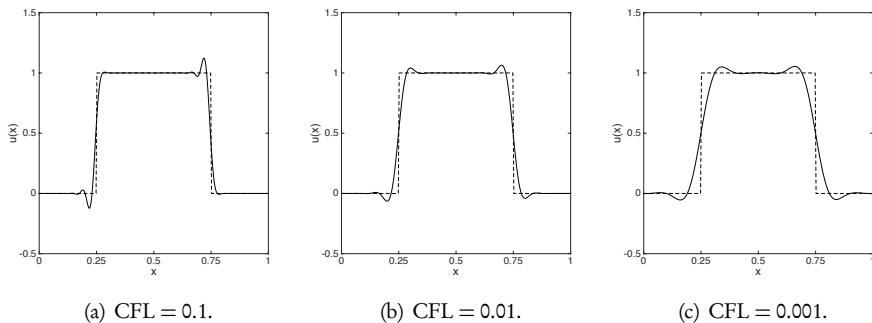
This is equivalent to the problem illustrated in Fig. 8.10. We solve the problem with a second and a fourth order central finite difference method, and apply filters of different orders.

The results are illustrated in Fig. 8.15. We use the filter defined in Table 8.2 but other filters yield qualitatively similar results. As expected, the use of a second order filter eliminates the oscillations entirely. It is also noteworthy that the dissipative effect is so strong that the difference between the second and fourth order scheme is essentially eliminated.

Increasing the order of the filter visually improves the accuracy at the expense of reintroduction of the oscillations. Furthermore, when the filter order is sufficiently high the accuracy of the fourth order scheme is again superior. Even though oscillations persist, the improvements as compared to the results in Fig. 8.10 are dramatic, in particular in terms of the spatial localization of the oscillations.



**Figure 8.15.** The effect of filtering on second- (top row) and fourth- (bottom row) order accurate schemes, filtered with a  $p$ th-order filter.



**Figure 8.16.** The effect of a fourth order filter as a function of the frequency by which the filter is applied.

A more careful inspection of the results in Fig. 8.15 highlights the interaction between the natural dispersion of the numerical scheme and the dissipation added by the filter. If we focus on the results for the fourth order scheme, we observe that the second and fourth order filters yield a symmetric solution, while with the eight order filter, the one-sided characteristics of the dispersive effects of the fourth order numerical method re-emerges.

We show in Fig. 8.16 the results obtained with a second order finite difference scheme and a fourth order filter. The only parameter that is varied in this case is the CFL number, translating into a variation of how often the filter is applied. We observe clearly that as the use of the filter increases, the solution transitions from being dominated by dispersive effects to a solution that is symmetric but with signs of increasing smearing. Similar observations can be made in Fig. 8.15, where the variation of the strength of the filter induces a similar transition.

To gain some insight into this, consider the linear wave problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0, \quad x \in [0, 2\pi],$$

subject to periodic boundary conditions. We assume that temporal errors can be neglected and use a central finite difference scheme of order  $2m$  in space. We also assume that we apply an exponential filter of order  $q$  to the solution at every time step. The modified equation for the semidiscrete scheme becomes

$$\frac{\partial v}{\partial t} + a \frac{\partial v}{\partial x} = aR(x, t) + \beta(-1)^{q/2+1}v^{(q)}(x, t),$$

where we have the remainder

$$R(x, t) \simeq \frac{(-1)^m (m!)^2}{(2m+1)!} h^{2m} v^{(2m+1)}(x, t) = (-1)^m C_{2m} h^{2m} v^{(2m+1)}(x, t),$$

and the second term reflects the effect of the exponential filter. Now assume that the solution can be represented by a Fourier series and consider a single wave,  $v(x, t) =$

$\hat{v}_l(t)e^{ilx}$ , to obtain

$$\frac{d\hat{v}_l}{dt} = (-ial + ialC_{2m}(bl)^{2m} - \beta l^q) \hat{v}_l(t).$$

This yields the solution at time  $t$  as

$$\hat{v}_l(t) = \exp(-ialt(1 - C_{2m}(bl)^{2m})) \exp(-\beta l^q t) \hat{v}_l(0).$$

We now reach back to subsection 8.1.1 to recall the dimensional variables  $v$  and  $p$ , designating periods and points per wavelength, respectively, and also recall that  $\gamma = \beta N^p k$  in the exponential filter, to recover

$$\hat{v}_l(t) = \exp(-i2\pi\nu(1 - C_{2m}(2\pi p^{-1})^{2m})) \exp(-\gamma K p^{-q}) \hat{v}_l(0), \quad K = \frac{t}{k}.$$

This captures the balance between the two effects of dispersion, measured by the first term, and dissipation, reflected in the second term. If the dispersive effects dominate, as in cases where  $2m < q$ , we should expect to see the wave trains associated with the dispersive effects. However, in the case where the dissipative effects dominate, symmetry should be maintained as all modes are simply damped, i.e., symmetries are maintained by the dissipative filter. Finally we observe that by increasing  $K$ , the dissipative effects should increase as well and, thus, help to retain symmetries as is evident in Fig. 8.16. ■

Filtering has two effects on the performance of the scheme. On one hand, it improves the stability of the scheme. To appreciate this, consider a linear scheme on fully discrete form as

$$U^{n+1} = G(k, h)U^n,$$

with the filtered version being

$$U^{n+1} = \Omega G(k, h)U^n.$$

We then have

$$\|U^{n+1}\| \leq \|\Omega G(k, h)\| \|U^n\| \leq \|\Omega\|^n \|G(k, h)\|^n \|U^0\|.$$

However,

$$\|U^*\|_2^2 = \|\Omega U^n\|_2^2 = \sum_l |\sigma(l) \hat{u}_l^n|^2 \leq \|U^n\|_2^2,$$

since  $\sigma(\eta) \leq 1$ . This implies that  $\|\Omega\|_2 \leq 1$ , hence improving stability. Indeed, even if  $\|G(k, h)\| > 1$ , indicating an unstable scheme, filtering can restore stability provided  $\|\Omega G(k, h)\| \leq 1$ . The stabilization of the central finite difference schemes, leading to the Lax-Friedrichs and Lax-Wendroff schemes discussed in Ex. 5.6, is an example of this.

While the enhanced stability of the filter is a positive effect, the impact of filtering on the overall accuracy of the computation is a more delicate issue. We know that the Lax-Friedrichs method is only first order accurate even though it is based on a second order central difference scheme, i.e., the order of accuracy is clearly impacted

by the dissipation. On the other hand, the results in Fig. 8.15 highlight the potential for substantial improvements of the pointwise accuracy. It is also clear, however, that the impact of filtering should not overwhelm the accuracy of the underlying scheme.

A complete error analysis of the impact of filtering on high-order finite difference methods is still lacking. However, since we have established a direct connection between finite difference methods and methods based on the Whittaker Cardinal function and, furthermore, explored the close connection between Whittaker Cardinal functions and Fourier series, it is relevant to look at the effect of filtering on functions represented by Fourier series. While the quantitative details will be different, the qualitative behaviors can be expected to be the same.

For a function, possibly with one or several points of discontinuity, the analysis of filtering on the pointwise accuracy of Fourier expansion is discussed in detail in [33], and we return to this in subsection 13.4.1. However, to summarize, let us first consider the impact of filtering on a smooth function  $u \in C^q(\Omega_x)$  when using a filter of order  $p$  with  $q > p$ . In this case

$$|u(x) - \Omega u(x)| \leq C h^{p-1/2} \|u^{(p)}\|_2,$$

where  $\Omega u(x)$  is the filtered approximation. This confirms that the order of the filter has a direct impact on the accuracy of the approximation, consistent with the results in Fig. 8.15, where the low-order filter substantially modifies the accuracy of the solution.

For a function which is only piecewise smooth, i.e.,  $u \in C^q(\Omega_x)$ , except at a finite number of points,  $\xi$ , one easily proves that the accuracy of the approximation does not improve at the point of discontinuity, i.e., filtering does not eliminate the Gibbs phenomenon. The pointwise accuracy of the filtered approximation is

$$|u(x) - \Omega u(x)| \leq C h^{p-1} \frac{1}{d(x)^{1-p}} \|u^{(p)}\|_2,$$

where  $d(x)$  is the distance to the nearest discontinuity. This suggests that high-order accuracy can indeed be recovered away from the points of discontinuity provided a high-order filter is used. This is consistent with the results in Fig. 8.15. This gives confidence that filtering can be used as an effective and accurate approach for the stabilization and improvement of the pointwise accuracy when using high-order methods for solving problems with discontinuous solutions.

## References

- [1] Saul Abarbanel, David Gottlieb, and Eitan Tadmor. *Spectral Methods for Discontinuous Problems*, ICASE Report NASA-CA-177974. Langley Research Center, NASA, 1985.
- [2] John P. Boyd. Sum-accelerated pseudospectral methods: Finite differences and sech-weighted differences. *Computer Methods in Applied Mechanics and Engineering*, 116(1):1–11, 1994.
- [3] Philip J. Davis. *Interpolation and Approximation*. Courier Corporation, 1975.
- [4] Wai Sun Don and Carl B. Quillen. Numerical simulation of shock-cylinder interactions: I. Resolution. *Journal of Computational Physics*, 122(2):244–265, 1995.

- [5] Bjorn Engquist and Björn Sjögren. The convergence rate of finite difference schemes in the presence of shocks. *SIAM Journal on Numerical Analysis*, 35(6):2464–2485, 1998.
- [6] Lawrence C. Evans. *Partial Differential Equations*, 2nd ed., American Mathematical Society, 1998.
- [7] Bengt Fornberg. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of Computation*, 51(184):699–706, 1988.
- [8] Bengt Fornberg. Classroom note: Calculation of weights in finite difference formulas. *SIAM Review*, 40(3):685–691, 1998.
- [9] Bengt Fornberg. *A Practical Guide to Pseudospectral Methods*. Vol. 1. Cambridge University Press, 1998.
- [10] Jonathan B. Goodman and Randall J. LeVeque. On the accuracy of stable schemes for 2d scalar conservation laws. *Mathematics of Computation*, 45:15–21, 1985.
- [11] David Gottlieb and Steven A. Orszag. *Numerical analysis of spectral methods: Theory and applications*, volume 26 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1977.
- [12] David Gottlieb and Chi-Wang Shu. On the Gibbs phenomenon and its resolution. *SIAM Review*, 39(4):644–668, 1997.
- [13] Bertil Gustafsson. The convergence rate for difference approximations to mixed initial boundary value problems. *Mathematics of Computation*, 29(130):396–406, 1975.
- [14] Ami Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49(3):357–393, 1983.
- [15] Jan S. Hesthaven, Sigal Gottlieb, and David Gottlieb. *Spectral Methods for Time-Dependent Problems*, volume 21 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2007.
- [16] Heinz-Otto Kreiss and Joseph Oliger. Comparison of accurate methods for the integration of hyperbolic equations. *Tellus*, 24(3):199–215, 1972.
- [17] Dietmar Kröner. *Numerical Schemes for Conservation Laws*. Wiley, 1997.
- [18] Peter D. Lax. Gibbs phenomena. *Journal of Scientific Computing*, 28(2-3):445–449, 2006.
- [19] P. G. Lefloch, J.-M. Mercier, and C. Rohde. Fully discrete, entropy conservative schemes of arbitrary order. *SIAM Journal on Numerical Analysis*, 40(5):1968–1992, 2002.
- [20] Randall J. LeVeque. *Numerical Methods for Conservation Laws*, Lectures in Mathematics, ETH Zürich. Birkhäuser-Verlag, 1992.
- [21] Marshal L. Merriam. *An Entropy-Based Approach to Nonlinear Stability*. NASA Technical Memorandum NASA-TM-101086, p. 64, 1989.

- 
- [22] Stanley Osher. Riemann solvers, the entropy condition, and difference. *SIAM Journal on Numerical Analysis*, 21(2):217–235, 1984.
  - [23] Stanley Osher and Sukumar Chakravarthy. High resolution schemes and the entropy condition. *SIAM Journal on Numerical Analysis*, 21(5):955–984, 1984.
  - [24] Chi-Wang Shu. TVB uniformly high-order schemes for conservation laws. *Mathematics of Computation*, 49(179):105–121, 1987.
  - [25] Chi-Wang Shu and Peter S. Wong. A note on the accuracy of spectral method applied to nonlinear conservation laws. *Journal of Scientific Computing*, 10(3):357–369, 1995.
  - [26] Frank Stenger. Numerical methods based on Whittaker cardinal, or sinc functions. *SIAM Review*, 23(2):165–224, 1981.
  - [27] Frank Stenger. *Numerical Methods Based on Sinc and Analytic Functions*, volume 20 of Springer Series in Computational Mathematics. Springer Science+Business Media, 2012.
  - [28] Peter K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 21(5):995–1011, 1984.
  - [29] Eitan Tadmor. The numerical viscosity of entropy stable schemes for systems of conservation laws. I. *Mathematics of Computation*, 49(179):91–103, 1987.
  - [30] Eitan Tadmor. Entropy stability theory for difference approximations of nonlinear conservation laws and related time-dependent problems. *Acta Numerica*, 12:451–512, 2003.
  - [31] Eitan Tadmor and Weigang Zhong. Novel entropy stable schemes for 1D and 2D fluid equations. In: *Hyperbolic Problems: Theory, Numerics, Applications*, pages 1111–1119. Springer, 2008.
  - [32] Vesa Välimäki. *Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters*. Helsinki University of Technology, 1995.
  - [33] Hervé Vandeven. Family of spectral filters for discontinuous problems. *Journal of Scientific Computing*, 6(2):159–192, 1991.
  - [34] Robert Vichnevetsky and John B. Bowles. *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*, volume 5 of Studies in Applied and Numerical Mathematics. SIAM, 1982.
  - [35] Gerald Beresford Whitham. *Linear and Nonlinear waves*, volume 42 of Pure and Applied Mathematics. John Wiley & Sons, 2011.
  - [36] Jens Zudrop and Jan S. Hesthaven. Accuracy of high order and spectral methods for hyperbolic conservation laws with discontinuous solutions. *SIAM Journal of Numerical Analysis*, 53(4):1857–1875, 2015.

## Chapter 9

# Strong stability preserving time integration

Before proceeding with the development of high-order schemes for the accurate spatial representation of solutions to conservation laws, we need to take a step back and consider the approximation in the temporal dimension. If we recall the generic conservation form

$$U_j^{n+1} = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n],$$

one quickly realizes that this is  $\mathcal{O}(k)$  accurate, except for special choices of the numerical flux, e.g., the Lax-Wendroff scheme. As we seek to develop schemes of arbitrarily high order, this is clearly not satisfactory.

In this chapter we describe the development of high-order schemes for the integration of the semidiscrete form of the conservation law, by employing a methods-of-lines approach. In other words, we assume that the spatial dimension is treated in some appropriate fashion and consider the semidiscrete problem as

$$\frac{dU_j(t)}{dt} = -\frac{F_{j+1/2}(t) - F_{j-1/2}(t)}{h} = L_j(U(t), t),$$

where  $L_j(U(t), t) = L_j(U_{j-p-1}(t), \dots, U_{j+q}(t), t)$ . We identify this as a system of ordinary differential equations in the form

$$\frac{dU(t)}{dt} = L(U(t), t),$$

where  $U(t) = [U_0(t), \dots, U_N(t)]^T$  is the grid function and  $L(U(t)) = [L_0(U(t)), \dots, L_N(U(t))]^T$  is the right-hand side. Using Picard's formula we can, under light assumptions to be discussed shortly, express the exact solution of the ordinary differential equation as

$$U_j^{n+1} = U_j^n + k \int_0^1 L_j(U(t+sk), t+sk) ds.$$

This expression does not yield a practical scheme since the argument of the integral remains unknown. The simplest scheme arises naturally if we assume that  $L_j(U(t), t)$

is constant,

$$U_j^{n+1} = U_j^n + k L_j(U(t^n), t^n) = U_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n].$$

Hence, we recover the familiar explicit scheme, based on forward Euler integration in time. Without resorting to an implicit scheme the extension of this approach to derive higher-order accurate time integration schemes is not entirely straightforward. For instance, if we apply the trapezoidal rule, we recover

$$U_j^{n+1} = U_j^n + k \frac{L_j(U(t^n), t^n) + L_j(U(t^{n+1}), t^{n+1})}{2},$$

which is a second order accurate implicit scheme. Using higher-order quadratures, this situation persists. This path to higher-order accurate scheme is possible, but explicit schemes are often desirable due to their simplicity and efficiency. Furthermore, since the orders of the temporal and the spatial operator are the same for conservation laws, explicit methods are often superior when measured in terms of computational efficiency.

While it is clear that the development of higher-order accurate explicit methods requires some thought, the nature of conservation laws imposes additional constraints. In particular, we seek methods which are TVD, i.e.,

$$\|U^{n+1}\|_{TV} \leq \|U^n\|_{TV},$$

to ensure the stability of the scheme.

This presents an additional challenge since all prior analysis has focused on the basic conservation form which has a limited temporal order of accuracy. It could appear that we will now need to develop a similar theoretical foundation for higher-order integration schemes, possibly for each scheme individually. This is clearly not feasible, and we need to approach this problem in a different way. Before doing so, let us consider an example to illustrate the importance of enforcing the TVD property for the fully discrete approximation.

**Example 9.1.** Let us again consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

subject to periodic boundary conditions and the smooth initial condition

$$u(x, 0) = 1 + \frac{1}{2} \cos(2\pi x).$$

Since the initial condition is strictly positive, the wave moves to the right, while a shock forms. For the spatial approximation we use a high-order accurate, weighted essentially nonoscillatory scheme (see Chapter 11) that guarantees that no oscillations are generated by the spatial scheme.

We consider Burgers' equation on the semidiscrete form

$$\frac{d\bar{u}(t)}{dt} = L(\bar{u}(t)),$$

where  $L(\bar{u})$  is a nonlinear operator that encodes the flux associated with Burgers' equation and  $\bar{u}$  is the vector of cell averages.

To integrate the semidiscrete form, we consider a standard explicit two-stage Runge–Kutta method:

$$\bar{u}^{(1)} = \bar{u}^n + k a_{21} L(\bar{u}^n), \quad \bar{u}^{n+1} = \bar{u}^n + k (b_1 L(\bar{u}^n) + b_2 L(\bar{u}^{(1)})).$$

Using Taylor expansions, one realizes that this scheme is locally third order accurate, resulting in a second order global accuracy, provided the coefficients satisfy the order conditions

$$b_1 + b_2 = 1, \quad a_{21} b_2 = \frac{1}{2}.$$

As an alternative, let us consider a scheme of the form

$$\bar{u}^{(1)} = \bar{u}^n + k a_1 L(\bar{u}^n), \quad \bar{u}^{n+1} = b_1 \bar{u}^n + b_2 (\bar{u}^{(1)} + k a_2 L(\bar{u}^{(1)})).$$

Utilizing Taylor expansions, we again recover order conditions

$$b_1 + b_2 = 1, \quad b_2 a_1 + b_2 a_2 = 1, \quad b_2 a_1 a_2 = \frac{1}{2},$$

which suffice to guarantee a local truncation error of third order. Comparing the two schemes, we observe that the first is a standard Runge–Kutta method, while the latter comprises a linear combination of two forward Euler steps, combined to ensure that second order global accuracy is maintained.

To makes things concrete, let us consider the two schemes [5]

$$\bar{u}^{(1)} = \bar{u}^n - 20k L(\bar{u}^n), \quad \bar{u}^{n+1} = \bar{u}^n + \frac{k}{40} (41L(\bar{u}^n) - L(\bar{u}^{(1)})), \quad (9.1)$$

and

$$\bar{u}^{(1)} = \bar{u}^n + k L(\bar{u}^n), \quad \bar{u}^{n+1} = \frac{1}{2} \bar{u}^n + \frac{1}{2} (\bar{u}^{(1)} + k L(\bar{u}^{(1)})), \quad (9.2)$$

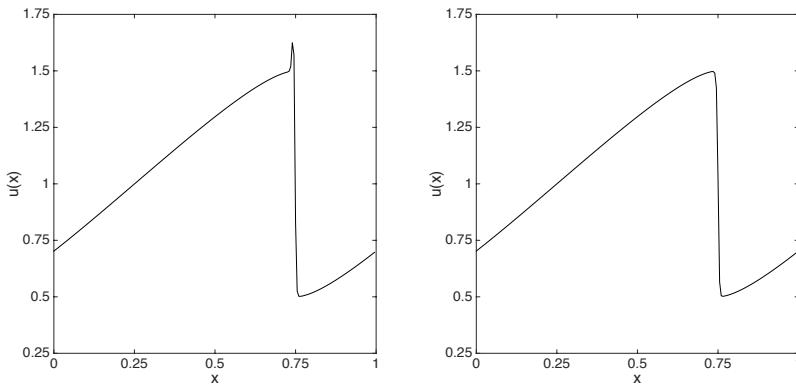
both of which are second order accurate. It is also worth observing that (9.2) can be written as a convex combination of forward Euler steps, while this is not the case for (9.1). As we shall see shortly, this is a key property.

Combined with a spatially stable scheme, we show in Fig. 9.1 the results obtained with these two schemes with  $N = 200$  control volumes. The most striking difference is that the simple change of the time-stepping methods has a substantial impact on the quality of the solution. Indeed, we observe that using (9.1) results in the introduction of oscillations. Since the spatial scheme is left unchanged, these oscillations emerge from the temporal integration scheme. ■

The development of computational methods to solve systems of ordinary differential equations is often based on Picard's theorem.

**Theorem 9.2.** *Assume a system of ordinary differential equations,*

$$\frac{dU}{dt} = L(U(t), t), \quad U(0) = U_0, \quad t \in [0, T], \quad (9.3)$$



**Figure 9.1.** Solution of Burgers' equation using a fifth order weighted essentially nonoscillatory (WENO) scheme in space and scheme (9.1) (left) and (9.2) (right) in time with  $N = 200$  control volumes. We show the solution at  $T = 0.25$ , computed with  $CFL = 0.3$  in both cases.

where  $L(U, t)$  is continuous in  $t$  and Lipschitz continuous in  $U$ . Then a unique solution of the form

$$U(t) = U_0 + \int_0^t L(U(s), s) ds,$$

exists for  $t \in [0, T]$ .

The proof is classic and can be found in, e.g., [2].

If we express this as a time-stepping scheme, we recover

$$U^{n+1} = U^n + k \int_0^1 L(U(t^n + ks), t^n + ks) ds,$$

which reduces the development of time integration schemes to the pursuit of efficient and accurate ways to evaluate the integral in the Picard solution. The details of how to approach this question typically separate schemes into two families of methods, known as multistage schemes, often associated with Runge–Kutta methods, and multistep schemes, often identified as Adams' methods.

## 9.1 • Runge–Kutta methods

Multistage schemes are based on a direct approximation of the Picard integral through fractional time steps within the integration interval. An  $s$ -stage Runge–Kutta method is based on the construction

$$\begin{cases} U^{(i)} = U^n + k \sum_{j=1}^s a_{ij} L(U^{(j)}, t^n + c_j k), & i = 1 \dots s, \\ U^{n+1} = U^n + k \sum_{i=1}^s b_i L(U^{(i)}, t^n + c_i k), \end{cases} \quad (9.4)$$

where  $U^{(1)} \dots U^{(s)}$  refer to the  $s$  intermediate stage values. Going forward, we assume for simplicity that the problem is autonomous,  $L(U(t))$ . For a nonautonomous

problem we can consider

$$\frac{d}{dt} \begin{bmatrix} U \\ t \end{bmatrix} = \begin{bmatrix} L(U(t)) \\ 1 \end{bmatrix}, \quad \begin{bmatrix} U(0) \\ t \end{bmatrix} = \begin{bmatrix} U_0 \\ 0 \end{bmatrix}.$$

Hence, any system with an explicit dependence on  $t$  can be written as an augmented autonomous system.

The coefficients of the classic Runge–Kutta method, expressed in the form of the Butcher tableau, are

$$\begin{array}{c|cc} c & A \\ \hline & b^T \end{array}, \quad A_{ij} = a_{ij}, \quad c_i = c_i, \quad b_i = b_i, \quad i, j = 1, \dots, s,$$

and are found by enforcing the order conditions. The details are classic and can be found in, e.g., [3]. The Runge–Kutta method is explicit if  $A$  is strictly lower triangular, and diagonally implicit if  $A$  is lower triangular. Otherwise one recovers a fully implicit Runge–Kutta method. For the explicit case, it is worth recalling that an  $s$ -stage explicit method can be at most of order  $s$ , while the order of an  $s$ -stage diagonally implicit Runge–Kutta method cannot exceed  $s + 1$ ; see [3].

A Runge–Kutta method in its classic form is defined uniquely through the Butcher tableau unless the method is reducible. In the current context, a method is reducible if  $a_{ij} = b_i = 0$  for some  $i$  and all  $j$ . In such a case, this stage can be eliminated without impacting the scheme. In what remains, we assume that all methods are irreducible.

Although the classic Runge–Kutta methods have proven themselves of substantial value and power for a broad range of problems, they present a challenge in the current context. Looking back at what we have discussed so far, all schemes are given as one-step schemes:

$$U^{n+1} = G(U^n).$$

For this case, we have a solid theoretical understanding of stability and convergence. However, the classic Runge–Kutta methods are not in this form, suggesting that we need to revisit questions of stability and convergence.

The solution to this problem is to return to the basic approximation in (9.4) and the prediction of the intermediate stage values. The classic approach is powerful, yet it is clearly not unique and we can seek alternatives. In the current context, it is of particular interest to pursue the development of schemes of the form [11]

$$\begin{cases} U^{(0)} = U^n, \\ U^{(i)} = \sum_{j=1}^s (\alpha_{ij} U^{(j)} + k \beta_{ij} L(U^{(j)})), \quad i = 1 \dots s+1; \\ U^{n+1} = U^{(s+1)}, \end{cases} \quad (9.5)$$

which is known as the Shu–Osher form of the Runge–Kutta method. In the following, we shall separate the discussion into explicit and implicit scheme.

### 9.1.1 • Explicit strong stability preserving (SSP) Runge–Kutta schemes

For explicit schemes, the central result is as follows [11].

**Theorem 9.3.** Assume that the Euler scheme is TVD-stable when solving (9.3) with time step  $k_{FE}$ . Provided  $\alpha_{ij}, \beta_{ij}$  are all semipositive, then the solution obtained with (9.5) with  $k \leq Ck_{FE}$  is TVD-stable. Here  $C \geq 0$  is a scheme-specific constant.

**Proof.** First of all we notice that

$$i = 1, \dots, s+1: \sum_{j=1}^i \alpha_{ij} = 1, \quad (9.6)$$

to ensure consistency. Next, consider

$$\begin{aligned} \|\mathbf{U}^{(i)}\|_{TV} &= \left\| \sum_{j=1}^i \left( \alpha_{ij} \mathbf{U}^{(j)} + k \beta_{ij} \mathbf{L}(\mathbf{U}^{(j)}) \right) \right\|_{TV} = \sum_{j=1}^i \left\| \alpha_{ij} \left( \mathbf{U}^{(j)} + k \frac{\beta_{ij}}{\alpha_{ij}} \mathbf{L}(\mathbf{U}^{(j)}) \right) \right\|_{TV} \\ &\leq \sum_{j=1}^i \alpha_{ij} \left\| \left( \mathbf{U}^{(j)} + k \frac{\beta_{ij}}{\alpha_{ij}} \mathbf{L}(\mathbf{U}^{(j)}) \right) \right\|_{TV}, \end{aligned}$$

if we assume that  $\alpha_{ij}$  is positive. Since

$$\left\| \left( \mathbf{U}^{(j)} + k \frac{\beta_{ij}}{\alpha_{ij}} \mathbf{L}(\mathbf{U}^{(j)}) \right) \right\|_{TV} \leq \|\mathbf{U}^{(j)}\|_{TV},$$

by TVD-stability of the forward Euler step provided  $k \frac{\beta_{ij}}{\alpha_{ij}} \leq k_{FE}$ , we immediately recover that  $\|\mathbf{U}^{(i)}\|_{TV} \leq \|\mathbf{U}^n\|_{TV}$  by (9.6). This further implies

$$\|\mathbf{U}^{n+1}\|_{TV} \leq \sum_{j=1}^{s+1} \alpha_{ij} \|\mathbf{U}^{(j)}\|_{TV} \leq \sum_{j=1}^{s+1} \alpha_{ij} \|\mathbf{U}^n\|_{TV} \leq \|\mathbf{U}^n\|_{TV},$$

hence completing the proof.  $\square$

Thus, if we can identify methods in which  $\alpha_{ij}$  and  $\beta_{ij}$  are semipositive, i.e., the scheme is constructed as a convex combination of forward Euler schemes, and chosen to retain high-order accuracy of the scheme, we overcome the need for additional analysis as stability follows from the stability of the forward Euler scheme. Schemes with such a property are called strong stability preserving (SSP) schemes. We refer to [6, 4] for comprehensive reviews of such methods beyond what is discussed here.

Searching for  $\alpha_{ij}$  and  $\beta_{ij}$ , we recall the requirement that

$$k \frac{\beta_{ij}}{\alpha_{ij}} \leq k_{FE} \Rightarrow k \leq \frac{\alpha_{ij}}{\beta_{ij}} k_{FE} = C(\alpha, \beta) k_{FE}.$$

Clearly, we should seek coefficients that maximize the SSP coefficient  $C(\alpha, \beta)$  defined as

$$C(\alpha, \beta) = \min_{ij} \frac{\alpha_{ij}}{\beta_{ij}}.$$

Some of the complications associated with this optimization are illustrated in the following example [4].

**Example 9.4.** Consider the standard, second order accurate, explicit 2-stage Runge–Kutta scheme

$$U^{(1)} = U^n + kL(U^n), \quad U^{n+1} = U^n + \frac{k}{2} (L(U^n) + L(U^{(1)})),$$

where, for simplicity of the notation, we consider a scalar problem.

Expressing this in the Shu–Osher form yields

$$U^{n+1} = \left( U^n + \frac{k}{2} L(U^n) \right) + \left( 0U^{(1)} + \frac{k}{2} L(U^{(1)}) \right),$$

which shows that  $\alpha_{22} = 0$  and  $\beta_{22} = 1/2$ . Since,  $C(\alpha_{22}, \beta_{22}) = 0$ , stability does not follow from the forward Euler scheme.

If we use the expression for  $U^{(1)}$  we recover a different scheme,

$$U^{n+1} = \left( \frac{3}{4}U^n + \frac{k}{4}L(U^n) \right) + \left( \frac{1}{4}U^{(1)} + \frac{k}{2}L(U^{(1)}) \right),$$

with  $C(\alpha, \beta) = C(\alpha_{22}, \beta_{22}) = 1/2$ . In this case, stability follows from stability of the Euler scheme but at the cost of twice the number of time steps.

This can be improved further by rewriting the scheme as

$$U^{n+1} = \frac{1}{2}U^n + \left( \frac{1}{2}U^{(1)} + \frac{k}{2}L(U^{(1)}) \right), \quad (9.7)$$

in which case  $C(\alpha, \beta) = C(\alpha_{22}, \beta_{22}) = 1$ .

To appreciate the optimality of this scheme, consider the general 2-stage scheme

$$U^{(1)} = \alpha_{11}U^n + k\beta_{11}L(U^n),$$

$$U^{n+1} = (\alpha_{21}U^n + k\beta_{21}L(U^n)) + (\alpha_{22}U^{(1)} + k\beta_{22}L(U^{(1)})).$$

By consistency, we have  $\alpha_{11} = 1$  and  $\alpha_{21} + \alpha_{22} = 1$ . A Taylor expansion yields the order conditions for second order as

$$\alpha_{11}\beta_{22} + \alpha_{22}\beta_{11} + \beta_{21} = 1, \quad \alpha_{22}\beta_{22}\beta_{11} = \frac{1}{2}.$$

Seeking schemes with the SSP coefficient exceeding one,  $\alpha_{11} = 1$  implies that  $\beta_{11} < 1$ . Combining this with  $\alpha_{22} \leq 1$  implies that  $\beta_{22} > 1/2$ , which again yields  $1 \geq \alpha_{22} > \beta_{22} > 1/2$ . This results in  $1 > \alpha_{22}\beta_{11} > 1/2$  and, thus,  $\beta_{21} < 0$ , hence violating the requirement that  $\beta_{ij}$  be positive. As a consequence, the SSP coefficient cannot exceed one for the general 2-stage scheme and (9.7) is an optimal second order 2-stage SSP Runge–Kutta scheme. ■

Going forward we name explicit SSP schemes of  $p$ th order with  $s$  stages as SSPERK( $s, p$ ), i.e., the optimal scheme (9.7) is SSPERK(2,2).

To recover schemes of higher order, we may pursue the optimization outlined in the example above. An optimal third order 3-stage SSP Runge–Kutta scheme—

SSPERK(3,3)—is given as

$$\begin{aligned} U^{(1)} &= U^n + kL(U^n), \\ U^{(2)} &= \frac{3}{4}U^n + \frac{1}{4}U^{(1)} + \frac{k}{4}L(U^{(1)}), \\ U^{n+1} &= \frac{1}{3}U^n + \frac{2}{3}U^{(2)} + \frac{2k}{3}L(U^{(2)}). \end{aligned} \quad (9.8)$$

The proof of optimality of SSPERK(3,3) with  $C(\alpha, \beta) = 1$  is given in [4], using the approach outlined in Ex. 9.4, i.e., by deriving the order conditions and optimizing the SSP coefficients under these constraints. It is established in [4] that this procedure cannot be continued and no fourth order 4-stage SSP Runge–Kutta (SSPERK) method exists if one requires that  $\beta_{ij}$  is semipositive.

While SSPERK(2,2) and SSPERK(3,3) are optimal schemes with the minimal number of stages, one can ask whether additional stages improves the overall efficiency of the scheme. To properly account for the computational cost associated with increasing the number of stages, let us define the effective SSP coefficient as

$$C_{eff} = \frac{1}{s} \max_{\alpha, \beta} C(\alpha, \beta).$$

Note that for SSPERK(2,2) we recover  $C_{eff} = 1/2$  and  $C_{eff} = 1/3$  for SSPERK(3,3). By increasing the number of stages,  $C_{eff}$  can be improved dramatically. An interesting example is the optimal SSPERK(10,4) [4]:

$$\begin{aligned} U^{(1)} &= U^n + \frac{k}{6}L(U^n), \\ U^{(2)} &= U^{(1)} + \frac{k}{6}L(U^{(1)}), \\ U^{(3)} &= U^{(2)} + \frac{k}{6}L(U^{(2)}), \\ U^{(4)} &= U^{(3)} + \frac{k}{6}L(U^{(3)}), \\ U^{(5)} &= \frac{3}{5}U^n + \frac{2}{5}U^{(4)} + \frac{k}{15}L(U^{(4)}), \\ U^{(6)} &= U^{(5)} + \frac{k}{6}L(U^{(5)}), \\ U^{(7)} &= U^{(6)} + \frac{k}{6}L(U^{(6)}), \\ U^{(8)} &= U^{(7)} + \frac{k}{6}L(U^{(7)}), \\ U^{(9)} &= U^{(8)} + \frac{k}{6}L(U^{(8)}), \\ U^{(10)} &= \frac{1}{25}U^n + \frac{9}{25}U^{(4)} + \frac{3}{5}U^{(9)} + \frac{3k}{50}L(U^{(4)}) + \frac{k}{10}L(U^{(9)}). \end{aligned}$$

This scheme is optimal among all explicit fourth order 10-stage schemes and has an effective SSP coefficient of  $C_{eff} = 3/5$ , i.e, it is about twice as efficient as SSPERK(3,3)

while being fourth order accurate. It is also worth observing that this 10-stage scheme can be implemented with only two additional variables, providing an example of a low-storage scheme. Other low-storage SSP methods are discussed in [4].

It can be shown that for a general explicit scheme, the optimal SSP coefficient cannot exceed the number of stages [4], i.e.,  $C_{eff} \leq 1$ .

### 9.1.2 • Implicit SSP Runge–Kutta schemes

To understand the extension of Theorem 9.3 to the implicit case, consider the backward Euler scheme

$$\mathbf{U}^{n+1} = \mathbf{U}^n + k\mathbf{L}(\mathbf{U}^{n+1}).$$

Take  $\gamma > 0$  and consider [4]

$$(1 + \gamma)\mathbf{U}^{n+1} = \mathbf{U}^n + \gamma\mathbf{U}^{n+1} + k\mathbf{L}(\mathbf{U}^{n+1}),$$

and rewrite it as

$$\mathbf{U}^{n+1} = \frac{1}{1 + \gamma} \left[ \mathbf{U}^n + \gamma \left( \mathbf{U}^{n+1} + \frac{k}{\gamma} \mathbf{L}(\mathbf{U}^{n+1}) \right) \right].$$

Provided  $k \leq \gamma k_{FE}$ , we have  $\|\mathbf{U}^{n+1} + \frac{k}{\gamma} \mathbf{L}(\mathbf{U}^{n+1})\|_{TV} \leq \|\mathbf{U}^{n+1}\|_{TV}$  by TVD-stability of the forward Euler scheme. Hence, we have

$$\|\mathbf{U}^{n+1}\|_{TV} \leq \frac{1}{1 + \gamma} [\|\mathbf{U}^n\|_{TV} + \gamma \|\mathbf{U}^{n+1}\|_{TV}],$$

which ensures  $\|\mathbf{U}^{n+1}\|_{TV} \leq \|\mathbf{U}^n\|_{TV}$  for any value of  $\gamma$ . Thus, the backward Euler method is strong stability preserving for  $k \leq \gamma k_{FE}$ . Since  $\gamma$  can take any value, this holds for arbitrarily large time steps,  $k$ .

Unfortunately, this result does not extend beyond methods of first order [12].

**Theorem 9.5.** *Consider an implicit multistage scheme of order  $p$ . For  $p > 1$ , the SSP coefficient is bounded.*

This negative result leaves open only the question of how large  $C_{eff}$  can be for an implicit scheme. While there currently is no rigorous result on this, extensive searches [4] suggest that all Runge–Kutta methods have a barrier of

$$C_{eff} \leq 2.$$

Hence, when compared to explicit methods, implicit SSP Runge–Kutta methods are inefficient due to the additional work associated with linear or nonlinear solvers.

In spite of this negative result, implicit SSP Runge–Kutta methods can be derived, given in the form (9.5). An optimal second order implicit Runge–Kutta method with  $s$  stages, SSPIRK(2,2), is given as

$$i = 1, \dots, s+1, j = 1, \dots, s : \quad \begin{aligned} \alpha_{ij} &= \begin{cases} 1, & i = j+1, \\ 0, & \text{otherwise,} \end{cases} \\ \beta_{ij} &= \frac{1}{2s} \begin{cases} 1, & i = j, i = j+1, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

It is easily seen that  $C_{eff} = 2$ . This optimal scheme is diagonally implicit.

An optimal third order implicit Runge–Kutta scheme with  $s$  stages, SSPRK( $s, 3$ ), is given as

$$i = 1, \dots, s+1, j = 1, \dots, s : \quad \alpha_{ij} = \begin{cases} 1, & i = j+1, \\ \gamma_4, & i = s+1, j = s, \\ 0, & \text{otherwise,} \end{cases} \quad \beta_{ij} = \begin{cases} \gamma_1, & i = j, \\ \gamma_2, & i = j+1, \\ \gamma_3, & i = s+1, j = s, \\ 0, & \text{otherwise} \end{cases}$$

where

$$\gamma_1 = \frac{1}{2} \left( 1 - \sqrt{\frac{s-1}{s+1}} \right), \quad \gamma_2 = \sqrt{\frac{s+1}{s-1}} \gamma_1, \\ \gamma_3 = \frac{(s+1)}{s(s+1 + \sqrt{s^2-1})}, \quad \gamma_4 = (s-1 + \sqrt{s^2-1}) \gamma_3.$$

The SSP coefficient is  $C = s - 1 + \sqrt{s^2 - 1} < 2s$ , confirming that  $C_{eff} < 2$ .

Implicit SSP Runge–Kutta methods of higher-order and stage counts can be found in [4], including up to 6th order. These schemes are generally diagonally implicit but since  $C_{eff} < 2$  they are advantageous only in very special situations.

### 9.1.3 • Order barriers

With the development of SSP Runge–Kutta methods, it is natural to ask what the connection to more standard Runge–Kutta methods is and if there are any fundamental limitations to the order of the SSP schemes.

Before doing so, let us consider the development of necessary, but not sufficient, order conditions for the classic Runge–Kutta scheme [2].

**Example 9.6.** Consider the scalar test problem

$$\frac{du}{dt} = u + t^{l+1}, \quad u(0) = 0, \quad (9.9)$$

where  $l \geq 1$  is a parameter.

Using the classic Runge–Kutta method (9.4) the scheme becomes

$$U^{(i)} = k \sum_{j=1}^s a_{ij} (U^{(j)} + (c_j k)^{l-1}).$$

If we introduce  $U = [U^{(1)}, \dots, U^{(s)}]^T$ ,  $C_{ii} = c_i$ , and  $e = [1, \dots, 1]^T$  is an  $s$ -long one-vector, the stage values are

$$(I - kA)U = k^l AC^{l-1}e.$$

Furthermore, we have

$$U^1 = k \sum_{i=1}^s b_i (U^{(i)} + (c_i k)^{l-1}) = k b^T U + k^l b^T C^{l-1} e.$$

Combining these expressions, we recover

$$U^1 = k^l \mathbf{b}^T [(\mathbf{I} - k\mathbf{A})^{-1} k\mathbf{A} + \mathbf{I}] \mathbf{C}^{l-1} \mathbf{e} \simeq k^l \mathbf{b}^T \left[ \sum_{n=0}^{\infty} (k\mathbf{A})^n \right] \mathbf{C}^{l-1} \mathbf{e},$$

assuming  $\|k\mathbf{A}\| \ll 1$ . The exact solution to (9.9) can be expressed as

$$u(k) = \sum_{n=0}^{\infty} \frac{1}{n!} k^n \frac{d^n u(0)}{dt^n}.$$

Repeatedly differentiating (9.9) yields

$$\frac{d^n u(0)}{dt^n} = \begin{cases} 0, & n < l, \\ (l-1)!, & n \geq l, \end{cases}$$

such that

$$u(k) = \sum_{n=0}^{\infty} \frac{(l-1)!}{(l+n)!} k^{l+n}.$$

Matching the coefficients yields the order conditions

$$\mathbf{b}^T \mathbf{A}^n \mathbf{C}^{l-1} \mathbf{e} = \frac{(l-1)!}{(l+n)!}, \quad 1 \leq l+n \leq p,$$

for a scheme with a local truncation error of  $k^{p+1}$ .

The conditions derived in this way are both necessary and sufficient for  $p \leq 3$  but only necessary for  $p > 3$ . To derive the complete set of order conditions for Runge–Kutta methods of arbitrary order, one must use the theory of rooted trees [3].

Let us also consider the stage order of the scheme, defined as the order of accuracy of the solution at the individual stage levels at which the local solution is an approximation to  $u(c_i k)$ . Utilizing Taylor expansions and matching terms as above, we recover the condition for stage order  $\tilde{p}$  as

$$\mathbf{A} \mathbf{C}^{l-1} \mathbf{e} = \frac{1}{l} \mathbf{C}^l \mathbf{e}, \quad l = 1, \dots, \tilde{p}. \quad (9.10)$$

Note that for  $l = 1$ , we recover the basis consistency requirement

$$\mathbf{A} \mathbf{e} = \mathbf{c}. \quad (9.11)$$

It is worth observing that since the second stage of an explicit Runge–Kutta method is a forward Euler step, the stage order cannot exceed one for an explicit scheme. Similarly, for a diagonally implicit Runge–Kutta method, the first stage is a backward Euler step, again restricting the stage order to one. Only by considering fully implicit Runge–Kutta methods or specially designed explicit schemes, avoiding the second stage, can this restriction be overcome. ■

As we have already seen in Ex. 9.4, the development of SSP schemes is not unique, which makes a simple connection to classic Runge–Kutta methods complicated. Let us begin by expressing the classic Runge–Kutta method (9.4) for the scalar problem in the form

$$U = \mathbf{e} U^n + k \mathbf{Q} \mathbf{L}, \quad (9.12)$$

where  $\mathbf{U} = [U^{(1)}, \dots, U^{(s)}, U^{n+1}]^T$ ,  $\mathbf{L} = [L(U^{(1)}), \dots, L(U^{(s)}), L(U^{n+1})]^T$ ,  $\mathbf{e}$  is an  $s+1$  one-vector, and

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{b}^T & 0 \end{bmatrix}$$

contains the Runge–Kutta coefficients on Butcher form.

To relate these to the SSP coefficients, let us consider the modified Shu–Osher form

$$\begin{cases} U^{(i)} = v_i U^n + \sum_{j=1}^s (\alpha_{ij} U^{(j)} + k \beta_{ij} L(U^{(j)})), & i = 1 \dots s+1, \\ U^{n+1} = U^{(s+1)}. \end{cases}$$

Consistency requires

$$i = 1 \dots s+1 : v_i + \sum_{j=1}^s \alpha_{ij} = 1 \Rightarrow \mathbf{v} + \alpha \mathbf{e} = \mathbf{e}, \quad (9.13)$$

where  $\alpha$  is  $(s+1) \times s$  matrix of coefficients. Expressing the modified scheme in compact form yields

$$\mathbf{U} = \mathbf{v} \mathbf{U}^n + \alpha \mathbf{U} + k \beta \mathbf{L}, \quad (9.14)$$

where the  $(s+1) \times (s+1)$  matrices  $\alpha$  and  $\beta$  are given as

$$\alpha = [\alpha \ 0], \quad \beta = [\beta \ 0],$$

and  $\alpha$  and  $\beta$  are  $(s+1) \times s$  matrices of SSP coefficients.

Solving for  $\mathbf{U}$  in (9.14) and using (9.13) yields

$$\mathbf{U} = \mathbf{e} \mathbf{U}^n + k(\mathbf{I} - \alpha)^{-1} \beta \mathbf{L},$$

from which, by comparing with (9.12), we recover the connection between the two schemes as

$$(\mathbf{I} - \alpha)^{-1} \beta = \mathbf{Q}, \quad (9.15)$$

provided that  $(\mathbf{I} - \alpha)$  is invertible. However, by considering the simple problem of  $u' = 0$  in (9.14) it is easily realized that this is a natural condition for all practical methods. Hence, any SSP method can be expressed in terms of its Butcher tableau by (9.15).

When trying to establish the reverse connection we should expect to encounter problems based on Ex. 9.4. Let us therefore restrict attention to SSP schemes with a given SSP coefficient  $r$ , and assume that all coefficients have the same SSP coefficient, i.e.,  $\alpha = r\beta$ . In this case (9.15) yields a canonical SSP scheme with

$$\beta = \mathbf{Q}(\mathbf{I} + r\mathbf{Q})^{-1}, \quad \alpha = r\beta, \quad \mathbf{v} = (\mathbf{I} - \alpha)\mathbf{e}.$$

Clearly, we must choose  $r$  such that  $\alpha_{ij}$  and  $\beta_{ij}$  are all semipositive. This has a number of important consequences. First of all, by choosing  $r = 0$  we immediately see that for a Runge–Kutta method to be SSP, the Butcher coefficients  $A_{ij}$  and  $b_i$  must all be nonnegative, since  $\mathbf{Q} \geq 0$  in an elementwise manner.

If we now assume that  $0 < r \ll 1$ , we can express

$$\beta = \sum_{k=0}^{\infty} (-1)^k r^k Q^{k+1} = Q - rQ^2 + r^2Q^3 + \dots \geq 0.$$

If  $b_j = 0$ , then  $Q_{s+1,j} = 0$  which, for small  $r$ , requires that  $(Q^2)_{s+1,j} = 0$  to ensure positivity. However,

$$(Q^2)_{s+1,j} = \sum_{i=1}^{s+1} b_i a_{ij} = 0.$$

Since  $a_{ij}$  and  $b_i$  are all semipositive, this requires that either  $b_i$  are all zero or  $a_{ij}$  are all zero. In either case, the Runge–Kutta method is reducible. Hence, we have established the following result [8].

**Theorem 9.7.** *Any irreducible Runge–Kutta method with a positive SSP coefficient has  $A \geq 0$  and  $b > 0$ .*

This result has a number of significant implications [8].

**Theorem 9.8.** *Any irreducible Runge–Kutta method with  $A \geq 0$  must have stage order  $\tilde{p} \leq 2$ .*

*Proof.* Recalling the stage order conditions (9.10) and (9.11), we have

$$i = 1 \dots s+1: \sum_{j=1}^s a_{ij} c_j = \frac{c_i^2}{2}, \quad \sum_{j=1}^s a_{ij} = c_i,$$

respectively. Let us assume that  $c_i$  are ordered such that  $0 \leq c_1 \leq \dots \leq c_s$ . Since  $A$  is semipositive,  $c_i$  are all semipositive. Multiply the second order condition with  $c_i$  and subtract the first to recover

$$\sum_{j=1}^s a_{ij} (c_i - c_j) = \frac{c_i^2}{2}.$$

If we consider  $i = 1$ , it is clear that the only solution is  $a_{1j} = c_1 = 0$ , as in an explicit method. In this case, the local stage error is 2.

If we now seek an additional order of accuracy, we have the additional constraint

$$\sum_{j=1}^s a_{ij} c_j^2 = \frac{c_i^3}{3}.$$

Proceeding as above, we select the smallest  $c_i > 0$  and recover

$$\sum_{j=1}^s a_{ij} (c_i^2 - c_j^2) = \frac{2c_i^3}{3}.$$

Since  $0 < c_i \leq c_j$  this is not possible with  $a_{ij} \geq 0$ . Hence, the stage satisfying this condition requires that  $a_{ij} = 0$ , in which case the method is reducible. This implies that the stage order of the Runge–Kutta method cannot exceed 2.  $\square$

This has a further consequence [8].

**Theorem 9.9.** *For any Runge–Kutta method of order  $p$  with  $b > 0$ , it holds that*

$$\tilde{p} \geq \left\lfloor \frac{p-1}{2} \right\rfloor,$$

where  $\tilde{p}$  is the stage order.

**Proof.** If we define the stage error as

$$\gamma_l = AC^{l-1}e - \frac{1}{l}C^l e,$$

it is shown in [1] that, for  $p \geq 5$ , one of the order conditions for any Runge–Kutta method is given as

$$b^T(\gamma_l)^2 = 0, \quad l = 1, \dots, \left\lfloor \frac{p-1}{2} \right\rfloor.$$

For  $b > 0$ , this is clearly not possible, unless the stage error vanishes, and the result follows.  $\square$

These two results imply that we cannot hope to develop explicit or diagonally implicit SSP Runge–Kutta methods of order higher than four since the stage order is one. Even the fully implicit SSP schemes cannot exceed sixth order, as the stage order cannot exceed two.

Finally, to close the loop between the classic Runge–Kutta schemes and the SSP schemes, let us quote the following result [4].

**Theorem 9.10.** *Consider a Runge–Kutta method with Butcher coefficients  $Q$ . The SSP coefficient of the method is*

$$C = \max\{r \geq 0 \mid (I + rQ)^{-1} \text{ exists and } \alpha \geq 0, v \geq 0\},$$

where the inequalities should be understood componentwise.

In other words, one can search for the maximum value of  $r$  that satisfies the positivity condition to derive optimal SSP schemes.

In Table 9.1, taken from [4], we summarize the best-known effective SSP coefficients for both  $s$ -stage explicit and implicit  $p$ th order Runge–Kutta schemes.

Before proceeding, it is worth remarking that one can relax the conditions on  $\alpha_{ij}, \beta_{ij}$  being semipositive and recover schemes with negative coefficients. This allows for relaxing some of the order conditions and develop higher-order SSP Runge–Kutta methods. However, the negative coefficients require one to solve problems backwards in time. This may introduce complications depending on the type of problem. For hyperbolic conservation laws, this is nevertheless possible. We refer to [4] for further discussions on this subject.

## 9.2 • Multistep methods

Let us now turn our attention to  $s$ -step multistep methods, expressed in the form

$$U^{n+1} = \sum_{i=0}^s \alpha_i U^{n+1-i} + \beta_i k L(U^{n+1-i}) = \sum_{i=0}^s \alpha_i \left( U^{n+1-i} + \frac{\beta_i}{\alpha_i} k L(U^{n+1-i}) \right),$$

**Table 9.1.** Effective SSP coefficients ( $C_{eff}$ ) for best known  $s$ -stage explicit and implicit  $p$ th order Runge–Kutta schemes [4].

s/p	SSPERK(s,p)			SSPIRK(s,p)				
	2	3	4	2	3	4	5	6
2	0.50	-	-	2	1.37	-	-	-
3	0.67	0.33	-	2	0.68	-	-	-
4	0.75	0.50	-	2	1.72	1.11	0.29	-
5	0.80	0.53	0.30	2	1.78	1.21	0.64	-
6	0.83	0.59	0.38	2	1.82	1.30	0.83	0.03
7	0.86	0.61	0.47	2	1.85	1.31	0.89	0.04
8	0.88	0.64	0.52	2	1.87	1.33	0.94	0.28
9	0.89	0.67	0.54	2	1.89	1.34	0.99	0.63
10	0.90	0.68	0.60	2	1.90	1.36	1.01	0.81
11	0.91	0.69	0.59	2	1.91	1.38	1.03	0.80

where  $\mathbf{U}^{n+1-i}$  refers to past values of the solution  $\mathbf{U}^n$ . If  $\alpha_0 = \beta_0 = 0$  the scheme is explicit; otherwise it is implicit.

Clearly, if  $\alpha_i, \beta_i$  are semipositive and

$$k \leq \min_i \frac{\alpha_i}{\beta_i} k_{FE},$$

the scheme is SSP in the sense that

$$\|\mathbf{U}^{n+1}\|_{TV} \leq \max_{i=1,\dots,s} \|\mathbf{U}^{n+1-i}\|_{TV}.$$

Accuracy of multistep schemes is recovered through Taylor expansions in the form

$$\begin{aligned} u(t^{n+1}) - \sum_{i=0}^s \alpha_i u(t^{n+1-i}) + \beta_i k L(u(t^{n+1-i})) \\ = - \sum_{j=1}^p k^j \sum_{i=0}^s \left( \alpha_i \frac{(-i)^n}{j!} + \beta_i \frac{(-i)^{j-1}}{(j-1)!} \right) u^{(j)}(t^{n+1}). \end{aligned}$$

The conditions for order  $p$  are

$$\sum_{i=0}^s i^j \alpha_i = j \sum_{i=0}^s i^{j-1} \beta_i, \quad j = 1, \dots, p.$$

This is a classic result that can be found in [7].

In contrast to the multistage schemes, there are no order barriers for the multistep schemes. However, the number of past steps can be very high for an efficient scheme, as indicated in the following result [9].

**Theorem 9.11.** *The SSP coefficient of an  $s$ -step explicit multistep method of order  $p > 1$  satisfies*

$$C \leq \frac{s-p}{s-1}.$$

An immediate consequence is that  $s > p$  and that  $C = 1$  is a maximum value of the SSP coefficient.

The situation does not improve significantly for implicit methods [10].

**Theorem 9.12.** *For  $p > 1$ , any implicit SSP multistep scheme has an SSP coefficient less than 2.*

A family of explicit second order  $s$ -stage methods, SSPEMS( $s, 2$ ), is given by [9]

$$\alpha_1 = \frac{(s-1)^2 - 1}{(s-1)^2}, \quad \alpha_s = \frac{1}{(s-1)^2}, \quad \beta_1 = \frac{s}{s-1},$$

with an SSP coefficient of  $C = \frac{s-2}{s-1} < 1$ . A third order SSPEMS( $4, 3$ ) scheme is

$$U^{n+1} = \frac{16}{27}(U^n + 3kL(U^n)) + \frac{11}{27}\left(U^{n-3} + \frac{12}{11}kL(U^{n-3})\right),$$

with an optimal SSP coefficient of  $1/3$ . Similar schemes, including schemes which employ more history terms and have a better SSP coefficient can be found in [4].

Implicit methods of very high order, including methods up to 8th order and 20 steps, can be found in [10]. The coefficients are obtained computationally and can be found in [10, 4]. However, due to Theorem 9.11, their use is limited.

## References

- [1] Peter Albrecht. A new theoretical approach to Runge–Kutta methods. *SIAM Journal on Numerical Analysis*, 24(2):391–406, 1987.
- [2] Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, volume 61 of Other Titles in Applied Mathematics. SIAM, 1998.
- [3] John C. Butcher. *Numerical Methods for Ordinary Differential Equations*, 2nd ed., John Wiley & Sons, Ltd., 2008.
- [4] Sigal Gottlieb, David I. Ketcheson, and Chi-Wang Shu. *Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations*. World Scientific, 2011.
- [5] Sigal Gottlieb and Chi-Wang Shu. Total variation diminishing Runge-Kutta schemes. *Mathematics of Computation*, 67(221):73–85, 1998.
- [6] Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1):89–112, 2001.
- [7] Ernest Hairer, Syvert P. Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems*, Springer, 2008.
- [8] J. F. B. M. Kraaijevanger. Contractivity of Runge-Kutta methods. *BIT Numerical Mathematics*, 31(3):482–528, 1991.

- [9] Hermanus W. J. Lenferink. Contractivity preserving explicit linear multistep methods. *Numerische Mathematik*, 55(2):213–223, 1989.
- [10] Hermanus W. J. Lenferink. Contractivity-preserving implicit linear multistep methods. *Mathematics of Computation*, 56(193):177–199, 1991.
- [11] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.
- [12] Marc N. Spijker. Contractivity in the numerical solution of initial value problems. *Numerische Mathematik*, 42(3):271–290, 1983.

## Chapter 10

# High-order accurate limiter-based methods

Let us return to the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

and its semidiscrete approximation in conservation form

$$\frac{dU_j}{dt} + \frac{F_{j+1/2} - F_{j-1/2}}{h} = 0.$$

For now we assume that the temporal integration is done with a suitable scheme and focus on the discretization of the spatial dimension. As we have discussed at length, there are two slightly different approaches to the approximation of the conservation law. In the finite difference method, we interpret  $U_j(t)$  as the approximation to the point values  $u(x_j, t)$  of the solution and the numerical flux,  $F_{j+1/2} = F(u_{j-p}, \dots, u_{j+q})$ , as an approximation to  $f(u_{j+1/2})$ .

In the finite volume method, on the other hand,  $U_j(t)$  represents the cell average  $\bar{u}_j(t)$  of the solution in the cell, centered at  $x_j$ . Furthermore,  $F_{j+1/2}$  approximates the flux at  $x_{j+1/2}$  by the recovery of  $u_{j+1/2}$  from the cell averages  $(\bar{u}_{j-p}, \dots, \bar{u}_{j+q})$  and the evaluation of  $f(u_{j+1/2})$ .

While it appears that these two formulations are conceptually quite different and lead to fundamentally different schemes, their close connection can be realized by considering [36, 37]

$$\frac{1}{h} \int_{x-h/2}^{x+h/2} F(y) dy = f(u(x)),$$

which postulates the existence of a function  $F$  with the property that  $F(x_{j\pm 1/2}) = F_{j\pm 1/2}$  is the numerical flux of the finite difference scheme. Differentiation of both sides of this expression yields

$$\frac{F(x+h/2) - F(x-h/2)}{h} = \frac{\partial f}{\partial x}.$$

Furthermore, by collocating at  $x_j$  we have

$$\bar{F}_j = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} F(x) dx = f(u_j),$$

which is a known quantity in a finite difference methods.

Hence, the essential difference between the two formulations lies in what is being recovered from the cell averages. In the finite volume method,  $u_{j+1/2}$  is recovered from cell averages of the solution  $u$  and  $f(u_{j+1/2})$  is used as the flux. However, in the finite difference method,  $F_{j+1/2}$  is recovered from cell averages of the numerical flux, defined as the point value of the flux of the conservation law.

In the simplest case of a first order scheme, these two viewpoints lead to the same schemes, as discussed in Part II. However, once we begin to consider higher-order accurate methods, differences between these two types of methods begin to emerge.

To construct a scheme of  $m$ th order accuracy, we must seek approximations such that

$$\frac{F_{j+1/2} - F_{j-1/2}}{h} = \frac{\partial f}{\partial x} \Big|_{x_j} + \mathcal{O}(h^m). \quad (10.1)$$

With this insight, it is clear that the construction of high-order accurate schemes reduces to the development of a reconstruction of the numerical flux or the solution at each cell interface, depending on which of the two viewpoints is being considered.

In the following we discuss families of high-order schemes for conservation laws, many of which are developed to achieve a fixed order of accuracy, typically second order. In subsequent chapters, we expand the discussion and introduce schemes that enable an arbitrary order of accuracy in smooth parts of the solution.

## 10.1 • Flux limited schemes

As implied by Godunov's theorem a nonlinear, i.e., solution dependent, scheme is required to reach higher-order accuracy. A fundamental and very successful approach to achieve this is to modify the numerical flux as

$$F_{j\pm 1/2}^n = F_{j\pm 1/2}^{n,L} + \phi_{j\pm 1/2} \left[ F_{j\pm 1/2}^{n,H} - F_{j\pm 1/2}^{n,L} \right],$$

where  $F_{j\pm 1/2}^{n,L}$  is a low-order monotone flux and  $F_{j\pm 1/2}^{n,H}$  a high-order numerical flux, satisfying (10.1) to order  $m$ . In other words, we redefine the numerical flux as a linear combination of a low-order flux and a high-order flux. The definition of  $\phi_{j\pm 1/2}$ , termed a flux limiter, is a central component of the scheme and must be defined with care. On one hand, for  $\phi_{j\pm 1/2} = 0$ , we recover the low-order scheme; on the other hand, when  $\phi_{j\pm 1/2} = 1$ , the result is the high-order scheme.

In the following we develop the insight to appropriately define the flux limiter  $\phi_{j\pm 1/2}$  and illustrate in subsection 10.1.4 the behavior of such schemes for some examples.

### 10.1.1 ■ Flux correction transport (FCT) schemes

Flux correction schemes, first proposed in [6], consider

$$A_{j \pm 1/2}^n = F_{j \pm 1/2}^{n,H} - F_{j \pm 1/2}^{n,L}$$

as a measure of the impact of insufficient dissipation in the high-order scheme. The limiter  $\phi_{j \pm 1/2}$  is defined by requiring that no new extrema will be introduced, and that existing extrema should not be accentuated. Following [6], this suggests

$$\phi_{j+1/2} A_{j+1/2}^n = \text{minmod}\left(A_{j+1/2}^n, \Delta^+ U_{j+1}^n, \Delta^- U_j^n\right), \quad (10.2)$$

where the minmod function is defined as

$$\text{minmod}(a_1, \dots, a_n) = \begin{cases} \text{sign}(a_1) \min_i |a_i| & \text{if } |\sum_i \text{sign}(a_i)| = n, \\ 0 & \text{otherwise.} \end{cases} \quad (10.3)$$

The minmod function returns the smallest argument, provided all arguments have the same sign, and otherwise returns zero. Its implementation is illustrated in `minmod.m`.

---

**Script 10.1. *minmod.m*: Implementation of the minmod function.**

---

```
function psi = minmod(v)
% function psi = minmod(v)
% Purpose: Implement the midmod function v is a vector
N = size(v,1); m = size(v,2); psi = zeros(N,1);
s = sum(sign(v),2)/m; ids = find(abs(s)==1);
if(~isempty(ids))
    psi(ids) = s(ids).*min(abs(v(ids,:)),[],2);
end
return;
```

---

We also introduce the intermediate state

$$U_j^* = U_j^n - \lambda \left[ F_{j+1/2}^{n,L} - F_{j-1/2}^{n,L} \right],$$

with  $\lambda = k/b$ .

As observed in [54], the absence of  $A_{j-1/2}^n$  and  $\Delta^+ U_j^n$  in (10.2) is surprising and results from the one-dimensional setting. An improved limiter function is given in [54] as

$$P_j^+ = \max(0, kA_{j-1/2}^n) - \min(0, kA_{j+1/2}^n), \quad Q_j^+ = (U_j^{max} - U_j^*)b.$$

Likewise, we define

$$P_j^- = \max(0, kA_{j+1/2}^n) - \min(0, kA_{j-1/2}^n), \quad Q_j^- = (U_j^* - U_j^{min})b,$$

and

$$R_j^\pm = \min(1, Q_j^\pm / P_j^\pm),$$

with  $R_j^\pm = 0$  if  $P_j^\pm = 0$ . The flux limiter is then defined as

$$\phi_{j+1/2} = \begin{cases} \min(R_{j+1}^+, R_j^-), & A_{j+1/2}^n \geq 0, \\ \min(R_j^+, R_{j+1}^-), & A_{j+1/2}^n < 0. \end{cases}$$

In this case, the extreme values of the intermediate solution  $U^*$  are given as

$$U_j^{max} = \max(U_{j-1}^*, U_j^*, U_{j+1}^*), \quad U_j^{min} = \min(U_{j-1}^*, U_j^*, U_{j+1}^*).$$

Furthermore, in [54] it is suggested that this be combined with a simple limiter such that  $A_{j+1/2}^n = 0$  if  $\Delta^+ U_j^n A_{j+1/2}^n < 0$  and either  $\Delta^+ U_{j+1}^n A_{j+1/2}^n < 0$  or  $\Delta^- U_j^n A_{j+1/2}^n < 0$ . It is, however, unclear if this is required to ensure stability, as no general stability theory is known.

The FCT scheme achieves high-order accuracy for solutions of sufficient smoothness, as highlighted in [5, 7]. Additional comparative studies can be found in [41]. The extension to multidimensional problems is discussed in [54], and a more extensive review can be found in [55]. Among several other developments, FCT schemes with pseudospectral accuracy [28], extensions to multiple dimensions, and element-based formulations are discussed in [15, 16, 17].

### 10.1.2 • TVD stable high-order schemes

While the two-step FCT scheme follows the overall vision of a flux limited scheme and suggests a strategy for designing the flux limiter, it lacks a rigorous stability theory. In order to address this, let us return to subsection 8.2.1 and consider flux limiters of the form

$$\phi_{j+1/2} = \phi_{j+1/2}(r_j), \quad r_j = \frac{\Delta^- U_j^n}{\Delta^- U_{j+1}^n} = \frac{\Delta^+ U_{j-1}^n}{\Delta^+ U_j^n} = \frac{\Delta^- U_j^n}{\Delta^+ U_j^n}.$$

As we have already discussed at length,  $\phi(1) = 1$  is required to enable second order accuracy.

The development of TVD-stable schemes requires the identification of the TVD region, illustrated in Fig. 8.5, for a particular choice of the two fluxes. To further explore this, let us consider an example in some detail, inspired by [39].

**Example 10.1.** Consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0,$$

subject to a smooth initial condition and periodic boundary conditions.

Let us take the low-order flux to be the Lax–Friedrichs flux

$$F_{j+1/2}^{n,L} = U_j^n,$$

and the high-order flux to be the Lax–Wendroff flux

$$F_{j+1/2}^{n,H} = \frac{1+\lambda}{2} U_j^n + \frac{1-\lambda}{2} U_{j+1}^n,$$

where  $\lambda = k/h$ . We seek a limiter  $\phi_{j+1/2}$  such that the modified flux

$$F_{j+1/2}^n = U_j^n - \phi_{j+1/2} \frac{1-\lambda}{2} [U_j^n - U_{j+1}^n] = U_j^n + \phi_{j+1/2} \frac{1-\lambda}{2} \Delta^+ U_j^n$$

leads to a TVD-stable scheme, given in the form

$$U_j^{n+1} = U_j^n - \lambda \Delta^- U_j^n - \frac{\lambda(1-\lambda)}{2} \Delta^- (\phi_{j+1/2} \Delta^+ U_j^n).$$

We recognize that the added term introduces antidiffusion, or compression, since  $0 \leq \lambda \leq 1$ . Hence, the role of the limiter is to control the level of artificial compression, required to reach second order accuracy in smooth regions, without introducing oscillations in nonsmooth regions of the solution.

Utilizing that

$$\Delta^+ U_j^n = \frac{\Delta^- U_j^n}{r_j},$$

we recover

$$U_j^{n+1} = U_j^n - \lambda \Delta^- U_j^n - \frac{\lambda(1-\lambda)}{2} \left( \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \right) \Delta^- U_j^n.$$

Reaching back to Theorem 8.4, we express this as

$$C_{j-1/2} = \frac{\lambda}{2} \left[ 2 + (1-\lambda) \left( \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \right) \right], \quad D_{j+1/2} = 0. \quad (10.4)$$

Thus, TVD-stability is ensured by the requirement

$$0 \leq \frac{\lambda}{2} \left[ 2 + (1-\lambda) \left( \frac{\phi_{j+1/2}}{r_j} - \phi_{j-1/2} \right) \right] \leq 1.$$

This condition can be met provided ( $0 \leq \lambda \leq 1$  to ensure stability)

$$0 \leq \phi(r) \leq 2, \quad 0 \leq \phi(r) \leq 2r, \quad (10.5)$$

assuming that  $\phi(r) = 0$ ,  $r \leq 0$ . We observe that if  $\phi(r) = 0$ , the scheme reduces to first order. As  $r \leq 0$  at any smooth extrema, this is in agreement with the result of Theorem 8.7. Borrowing the notation of Theorem 4.5, we have

$$0 \leq \phi(r) \leq 2(1 \wedge r^+), \quad (10.6)$$

and we recover the TVD-domain of admissible values of  $\phi(r)$ , illustrated in Fig. 10.1.

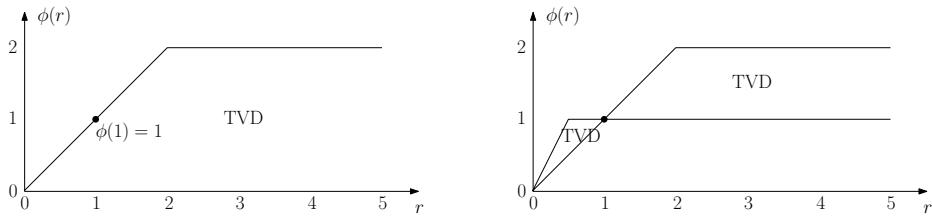
To illustrate the dependence of the TVD-region on the choice of the scheme, let us also consider the second order accurate Beam–Warming scheme [1, 50] with the numerical flux as

$$F_{j+1/2}^{n,H} = \frac{1}{2} (3U_j^n - U_{j-1}^n) + \frac{\lambda}{2} (U_j^n - U_{j-1}^n),$$

yielding the second order scheme

$$U_j^{n+1} = U_j^n - \frac{\lambda}{2} (3U_j^n - 4U_{j-1}^n + U_{j-2}^n) + \frac{\lambda^2}{2} (\Delta^-)^2 U_j^n.$$

We note that the main difference between the Lax–Wendroff scheme and the Beam–Warming method is the increased upwinding in the latter scheme. We also observe that if  $\phi(r) = r$ , in (10.4), we recover the Beam–Warming scheme.



**Figure 10.1.** Illustration of the TVD-region for the linear wave equation. On the left we show the TVD-region for a flux limited Lax–Wendroff scheme, and the right shows the region for a weighted scheme, comprising a Lax–Wendroff scheme, and a Beam–Warming scheme [1, 50] as the high-order scheme.

When rewritten as a flux limited scheme, this yields

$$U_j^{n+1} = U_j^n - \lambda \Delta^- U_j^n - \frac{\lambda(1-\lambda)}{2} \Delta^- (\phi_{j+1/2} \Delta^- U_j^n).$$

If we restrict attention to schemes utilizing  $(U_{j-2}^n, \dots, U_j^n)$ , the antidiffusion term should be a convex combination of the Lax–Wendroff and Beam–Warming schemes [39], i.e.,

$$\phi(r) = (1 - \theta(r))\phi^{LW}(r) + \theta(r)\phi^{BW}(r), \quad 0 \leq \theta(r) \leq 1.$$

To match terms, we observe that  $\phi^{LW}(r) = 1$  and  $\phi^{BW}(r) = r$  to obtain

$$\phi(r) = 1 - \theta(r) + r\theta(r),$$

which yields additional constraints on  $\phi$  as

$$\begin{cases} r \leq \phi(r) \leq 1, & r \leq 1, \\ 1 \leq \phi(r) \leq r, & r \geq 1. \end{cases}$$

Combining these with the previous conditions yields the TVD-region illustrated in Fig. 10.1. As expected,  $\phi(1) = 1$  is a natural member of the TVD-region. ■

Once the TVD-region is identified, the specific design of the limiter is flexible to within the region. Consequently, a substantial number of limiter functions  $\phi(r)$  have been proposed. In Fig. 10.2 we illustrate a number of these with the details of the limiter function given below.

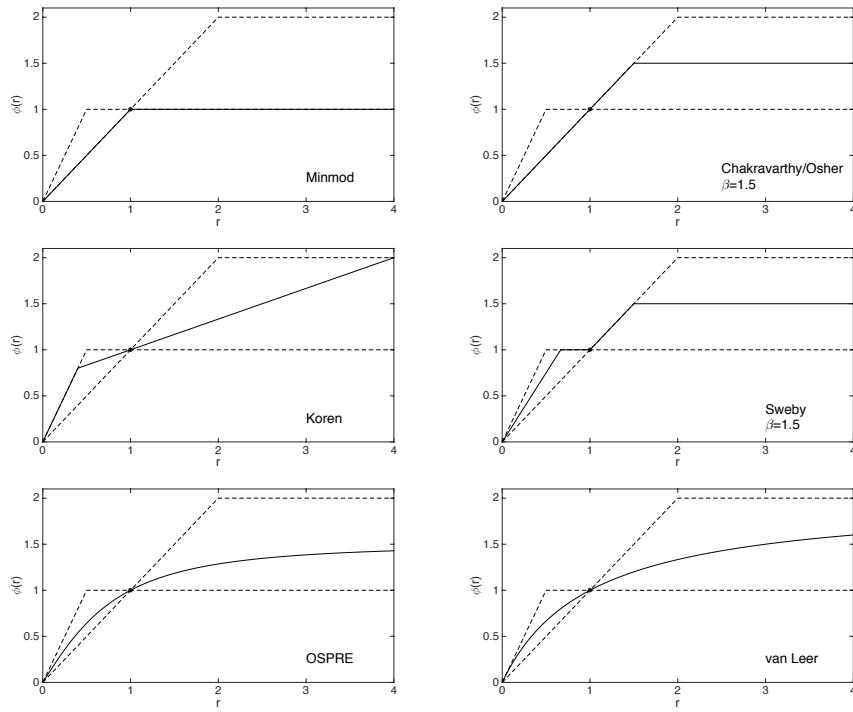
The Chakravarthy–Osher limiter is defined in [32]:

$$\phi(r) = \max[0, \min(r, \beta)], \quad 1 \leq \beta \leq 2.$$

The special case of  $\beta = 1$ , often referred to as the minmod limiter, is the most dissipative among the limiters discussed here. This property is easily realized from Fig. 10.2, from which it is clear that the minmod limiter traces the lower bound of the TVD-region and, thus, does not enable any compression.

In [21] it is proposed to use

$$\phi(r) = \max[0, \min(2r, (2+r)/3, 2)].$$



**Figure 10.2.** Illustration of limiters in the TVD-stability domain.

This exhibits third order accuracy for sufficiently smooth solutions and is known as the Koren limiter. An alternative is the Sweby limiter

$$\phi(r) = \max[0, \min(\beta r, 1), \min(r, \beta)], \quad 1 \leq \beta \leq 2,$$

proposed in [39]. The special case  $\beta = 2$ , known as the Superbee limiter [34], traces the outer boundary of the TVD-domain and, thus, enables maximum compression. Finally, smoother limiter functions have been proposed, e.g., the OSPRE limiter

$$\phi(r) = \frac{3(r^2 + r)}{2(r^2 + r + 1)}$$

in [51] and

$$\phi(r) = \max\left[0, \frac{2r}{1+r}\right],$$

often termed the van Leer limiter [44]. Numerous alternatives are discussed in [52, 40].

It is worth observing that flux limiter functions generally are symmetric in the sense that

$$\phi(r^{-1}) = \frac{\phi(r)}{r}.$$

Although this is not required, it is often preferred. We refer to [4] for a discussion of this aspect.

Whereas the discussion of the development of limiter functions has focused on the scalar linear problem, the application to both nonlinear scalar problems and systems is straightforward. In subsection 10.1.4 we return to a discussion of this, as well as a comparison of the properties of some limiter functions in more detail.

As we have previously seen, TVD-stability does not suffice to guarantee a cell entropy condition and convergence to the entropy solution. As illustrated in the following example, adapted from [30, 31], additional conditions are needed to ensure this.

**Example 10.2.** We consider the semidiscrete scheme

$$\frac{dU_j}{dt} = -\frac{F_{j+1/2} - F_{j-1/2}}{h},$$

and assume that the numerical flux  $F_{j+1/2}$  is an E-flux. Thus, we rewrite the scheme as

$$\frac{dU_j}{dt} = -\frac{1}{h} C_{j-1/2} \Delta^- U_j + \frac{1}{h} D_{j+1/2} \Delta^+ U_j,$$

where

$$C_{j-1/2} = \frac{f(U_j) - F_{j-1/2}}{\Delta^- U_j} \geq 0, \quad D_{j+1/2} = -\frac{F_{j+1/2} - f(U_j)}{\Delta^+ U_j} \geq 0$$

establishes TVD-stability by the properties of the E-flux. Note that if  $\Delta^- U_j = 0$ , consistency of the numerical flux ensures that  $C_{j-1/2} \Delta^- U_j = 0$  and, similarly,  $D_{j+1/2} \Delta^+ U_j = 0$  in case  $\Delta^+ U_j = 0$ .

As discussed in Ex. 8.5, such a scheme is, at best, first order accurate. Let us therefore consider the modified scheme

$$\frac{dU_j}{dt} = -\frac{1}{h} \left( 1 + \frac{1}{2} \Delta^+ \right) C_{j-1/2} \Delta^- U_j + \frac{1}{h} \left( 1 - \frac{1}{2} \Delta^- \right) D_{j+1/2} \Delta^+ U_j. \quad (10.7)$$

Writing out the flux, one recovers

$$\frac{dU_j}{dt} = -\frac{f(U_{j+1}) - f(U_{j-1})}{2h},$$

reflecting an expected second order accuracy in the smooth case. Expressing this in TVD-form, we have

$$\frac{dU_j}{dt} = -\frac{1}{h} \tilde{C}_{j-1/2} \Delta^- U_j + \frac{1}{h} \tilde{D}_{j+1/2} \Delta^+ U_j,$$

where

$$\tilde{C}_{j-1/2} = \frac{C_{j-1/2}}{2} \left[ 1 + \frac{f(U_{j+1}) - F_{j+1/2}}{f(U_j) - F_{j-1/2}} \right] = \frac{C_{j-1/2}}{2} \left[ 1 + \frac{1}{r_j^-} \right],$$

$$\tilde{D}_{j+1/2} = \frac{D_{j+1/2}}{2} \left[ 1 + \frac{F_{j-1/2} - f(U_{j-1})}{F_{j+1/2} - f(U_j)} \right] = \frac{D_{j+1/2}}{2} \left[ 1 + \frac{1}{r_j^+} \right],$$

and we have defined the two variables

$$r_j^- = \frac{f(U_j) - F_{j-1/2}}{f(U_{j+1}) - F_{j+1/2}}, \quad r_j^+ = \frac{F_{j+1/2} - f(U_j)}{F_{j-1/2} - f(U_{j-1})}.$$

Clearly, it is not possible to control these variables to ensure TVD-stability.

Let us consider the limited scheme as

$$\frac{dU_j}{dt} = -\frac{1}{h} \left( 1 + \frac{1}{2} \Delta^+ \phi(r_{j-1}^-) \right) C_{j-1/2} \Delta^- U_j + \frac{1}{h} \left( 1 - \frac{1}{2} \Delta^- \phi(r_{j+1}^+) \right) D_{j+1/2} \Delta^+ U_j, \quad (10.8)$$

where  $\phi(r)$  is the flux limiter. Expressed in TVD-form this yields

$$\begin{aligned} \hat{C}_{j-1/2} &= C_{j-1/2} \left[ 1 + \frac{1}{2} \left( \frac{\phi(r_j^-)}{r_j^-} - \phi(r_{j-1}^-) \right) \right], \\ \hat{D}_{j+1/2} &= D_{j+1/2} \left[ 1 + \frac{1}{2} \left( \frac{\phi(r_j^+)}{r_j^+} - \phi(r_{j+1}^+) \right) \right]. \end{aligned}$$

By (10.5) the scheme is TVD.

To establish a cell entropy condition, we recall Ex. 8.5 and the condition

$$\int_{U_j}^{U_{j+1}} \eta''(w) [H_{j+1/2} - f(w)] dw \leq 0,$$

for entropy stability. Here  $(\eta, \phi)$  is the entropy pair.

The numerical flux in (10.8) is given as

$$H_{j+1/2} = F_{j+1/2} - \frac{1}{2} \phi(r_{j+1}^+) [F_{j+1/2} - f(U_j)] + \frac{1}{2} \phi(r_j^-) [f(U_{j+1}) - F_{j+1/2}].$$

By integration by parts, one easily confirms that

$$\begin{aligned} - \int_{U_j}^{U_{j+1}} f(w) dw &= -\Delta^+ U_j \frac{f(U_j) + f(U_{j+1})}{2} \\ &\quad + \frac{1}{2} \int_{U_j}^{U_{j+1}} \left[ \left( \frac{\Delta^+ U_j}{2} \right)^2 - \left( w - \frac{U_j + U_{j+1}}{2} \right)^2 \right] f''(w) dw. \end{aligned}$$

To proceed, let us assume that the flux is  $f(u) = u^2/2$  in Burgers' equation, and that the entropy is  $\eta(u) = u^2/2$ , to obtain

$$\int_{U_j}^{U_{j+1}} \left[ \left( \frac{\Delta^+ U_j}{2} \right)^2 - \left( w - \frac{U_j + U_{j+1}}{2} \right)^2 \right] dw = \frac{(\Delta^+ U_j)^3}{6}.$$

The cell entropy then becomes

$$\begin{aligned} & \int_{U_j}^{U_{j+1}} [H_{j+1/2} - f(w)] dw \\ &= \frac{1}{2} \Delta^+ U_j \left[ (1 - \phi(r_{j+1}^+)) (F_{j+1/2} - f(U_j)) + (1 - \phi(r_j^-)) (F_{j+1/2} - f(U_{j+1})) \right] + \frac{(\Delta^+ U_j)^3}{12} \\ &= -\frac{(\Delta^+ U_j)^2}{2} \left[ (1 - \phi(r_{j+1}^+)) D_{j+1/2} + (1 - \phi(r_j^-)) C_{j+1/2} \right] + \frac{(\Delta^+ U_j)^3}{12}. \end{aligned}$$

Since the scheme is TVD,  $C_{j+1/2}$  and  $D_{j+1/2}$  are positive. Furthermore, for  $U_j \geq U_{j+1}$  ( $\Delta^+ U_j \leq 0$ ), as for a entropy satisfying shock, the integral is negative, and  $\phi(r) \leq 1$  suffices to guarantee a cell entropy condition. However, for the case of  $U_{j+1} \geq U_j$  ( $\Delta^+ U_j \geq 0$ ) additional dissipation is needed, requiring  $\phi$  to decrease below one, thus eliminating the possibility for second order accuracy.

Hence, to seek a second order accurate TVD-scheme with a cell entropy condition, additional modifications are needed. Following [31], we introduce the modified flux differences

$$\begin{aligned} \overline{F_{j+1/2} - f(U_j)} &= (F_{j+1/2} - f(U_j)) \left( 1 - a_{j+1/2}^- \frac{\partial_1 F(U_{j+1}, U_{j+1}) - \partial_1 F(U_j, U_j)}{D_{j+1/2}} \right), \\ \overline{f(U_{j+1}) - F_{j+1/2}} &= (f(U_{j+1}) - F_{j+1/2}) \left( 1 - a_{j+1/2}^+ \frac{\partial_2 F(U_{j+1}, U_{j+1}) - \partial_2 F(U_j, U_j)}{C_{j+1/2}} \right). \end{aligned}$$

The coefficients  $a_{j+1/2}^\pm$  depend on  $(U_j, U_{j+1})$  and are restricted to guarantee that the parentheses in the above take values between 0 and 2, i.e.,

$$\begin{aligned} \left| a_{j+1/2}^- (\partial_1 F(U_{j+1}, U_{j+1}) - \partial_1 F(U_j, U_j)) \right| &\leq D_{j+1/2}, \\ \left| a_{j+1/2}^+ (\partial_2 F(U_{j+1}, U_{j+1}) - \partial_2 F(U_j, U_j)) \right| &\leq C_{j+1/2}. \end{aligned}$$

Defining the modified indicator functions

$$\bar{r}_j^- = \frac{\overline{f(U_j) - F_{j-1/2}}}{\overline{f(U_{j+1}) - F_{j+1/2}}}, \quad \bar{r}_j^+ = \frac{\overline{F_{j+1/2} - f(U_j)}}{\overline{F_{j-1/2} - f(U_{j-1})}},$$

we recover the modified numerical flux

$$H_{j+1/2} = F_{j+1/2} - \frac{1}{2} \phi(\bar{r}_{j+1}^+) \left[ \overline{F_{j+1/2} - f(U_j)} \right] + \frac{1}{2} \phi(\bar{r}_j^-) \left[ \overline{f(U_{j+1}) - F_{j+1/2}} \right].$$

To establish a cell entropy condition, it now suffices to find  $a_{j+1/2}^\pm$  such that

$$\begin{aligned} & \int_{U_j}^{U_{j+1}} \left[ \left( \frac{\Delta^+ U_j}{2} \right)^2 - \left( w - \frac{U_j + U_{j+1}}{2} \right)^2 \right] f''(w) dw \\ & - a_{j+1/2}^- (\Delta^+ U_j)^2 \left[ \partial_1 F(U_{j+1}, U_{j+1}) - \partial_1 F(U_j, U_j) \right] \\ & - a_{j+1/2}^+ (\Delta^+ U_j)^2 \left[ \partial_2 F(U_{j+1}, U_{j+1}) - \partial_2 F(U_j, U_j) \right] \leq 0, \end{aligned}$$

since the contribution of the additional term is controlled by the flux limiter.

The choice for  $a_{j+1/2}^\pm$  depends on the flux  $f(w)$  and the choice of the numerical flux  $F_{j+1/2}$ . For Burgers' equation and a Lax-Friedrichs flux, one easily proves that  $a_{j+1/2}^\pm \geq 1/6$  suffices for the cell entropy condition to hold. Other choices of the flux yield slightly different values [31]. ■

### 10.1.3 • Positive schemes

While the development of TVD-stable flux limited high-order schemes represents a major advance, it builds on the concept of TVD-stability which, as discussed in subsection 8.2.1, holds only for problems in one spatial dimension. Even though TVD-stable schemes have been used with substantial success beyond one dimension, this is not supported by analysis.

Looking for alternatives, one realizes that the only known bounded functional for general linear hyperbolic systems is the energy [13]. If we consider the linear system

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{A}(x) \frac{\partial \mathbf{u}}{\partial x} = 0,$$

one easily proves that there exists a positive constant  $c$  such that

$$\|\mathbf{u}(t)\|_2 \leq \exp(ct) \|\mathbf{u}(0)\|_2,$$

provided  $\mathbf{A}(x)$  is Lipschitz continuous. The extension to systems in multiple dimensions is immediate.

In [13] it is shown that solutions to such problems can be approximated by schemes of the form

$$\mathbf{U}_j^{n+1} = \sum_{|k| \leq K} \mathbf{C}_K(j, k) \mathbf{U}_{j+k}^n. \quad (10.9)$$

Here  $\mathbf{U}_j^n$  is an  $m$ -vector of unknowns. The coefficient matrices  $\mathbf{C}_K(j, k)$  depend on  $(\mathbf{U}_{j-k}^n, \dots, \mathbf{U}_{j+k}^n)$ . Furthermore, if the hyperbolic system is symmetric (or symmetrizable),  $\mathbf{C}_K$  is also symmetric.

Following [25], consider

$$|\mathbf{U}_j^{n+1}|^2 = \sum_{|k| \leq K} \langle \mathbf{U}_j^{n+1}, \mathbf{C}_K(j, k) \mathbf{U}_{j+k}^n \rangle,$$

where  $\langle \cdot, \cdot \rangle$  is the scalar product. Let us assume that  $\mathbf{C}_K$  is semipositive, in which case the algebraic-geometric inequality yields

$$\langle \mathbf{U}_j^{n+1}, \mathbf{C}_K(j, k) \mathbf{U}_{j+k}^n \rangle \leq \frac{1}{2} \langle \mathbf{U}_j^{n+1}, \mathbf{C}_K(j, k) \mathbf{U}_j^{n+1} \rangle + \frac{1}{2} \langle \mathbf{U}_{j+k}^n, \mathbf{C}_K(j, k) \mathbf{U}_{j+k}^n \rangle.$$

Further assuming

$$\sum_{|k| \leq K} \mathbf{C}_K(j, k) = \mathbf{I}, \quad (10.10)$$

we recover that

$$\sum_{|k| \leq K} \langle \mathbf{U}_j^{n+1}, \mathbf{C}_K(j, k) \mathbf{U}_j^{n+1} \rangle = |\mathbf{U}_j^{n+1}|^2,$$

and, thus, that

$$|U_j^{n+1}|^2 \leq \sum_{|k| \leq K} \langle U_{j+k}^n, C_K(j, k) U_{j+k}^n \rangle.$$

Summing over  $j$ , we recover

$$\|U^{n+1}\|_2^2 \leq \sum_j \sum_{|k| \leq K} \langle U_{j+k}^n, C_K(j, k) U_{j+k}^n \rangle = \sum_{|k| \leq K} \sum_i \langle U_i^n, C_K(i - k, k) U_i^n \rangle,$$

where  $i = j + k$ .

If we now further assume that  $C_K$  is Lipschitz continuous, we recover

$$C_K(i - k, k) \simeq C_K(i, k) + \mathcal{O}(h),$$

which, when combined with (10.10), yields

$$\|U^{n+1}\|_2^2 \leq (1 + \mathcal{O}(h)) \|U^n\|_2^2,$$

from which discrete stability follows.

Hence, if we consider schemes of the form (10.9) and assume that  $C_K$  is symmetric, semipositive, Lipschitz continuous and satisfies (10.10), the scheme is stable. Such schemes are called positive schemes [25], and they offer a path to the development of stable, high-order accurate schemes for symmetrizable hyperbolic systems in multiple dimensions.

Before pursuing such developments, it is worth realizing that once shocks form, the stability result fails, as  $C_K$  will no longer be Lipschitz continuous. Nevertheless, as demonstrated in [25, 24], the scheme performs very well even in this case.

The development of positive schemes follows a path similar to that of the flux limited schemes by considering the numerical flux

$$F_{j \pm 1/2}^n = F_{j \pm 1/2}^{n,L} + \phi_{j \pm 1/2} \left[ F_{j \pm 1/2}^{n,H} - F_{j \pm 1/2}^{n,L} \right].$$

The development of a particular positive scheme requires that the choice of these two fluxes and the flux limiter to allow a proof that  $C_K$  satisfies the required conditions. We shall illustrate this development through an example, taken from [25].

**Example 10.3.** Consider the symmetric system of conservation laws

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u})}{\partial x} = 0, \quad x \in \Omega_x \subset \mathbb{R},$$

subject to appropriate initial and boundary conditions. We assume that  $\mathbf{u}$  is an  $m$ -vector.

We select the Roe flux

$$F_{j \pm 1/2}^{n,L} = \frac{f(U_j^n) + f(U_{j+1}^n)}{2} - \frac{1}{2} |\hat{\mathbf{A}}| (U_{j+1}^n - U_j^n),$$

as the low-order flux. The Roe matrix  $\hat{\mathbf{A}}$  is discussed in subsection 6.2.1. We assume that  $\hat{\mathbf{A}}$  is symmetric and diagonalizable,  $\hat{\mathbf{A}} = \mathbf{S} \mathbf{A} \mathbf{S}^{-1}$ , and define

$$|\hat{\mathbf{A}}| = \hat{\mathbf{A}}^+ - \hat{\mathbf{A}}^-,$$

with  $\hat{A}^\pm = S\Lambda^\pm S^{-1}$  and

$$\Lambda_{ii}^+ = \lambda_i \vee 0, \quad \Lambda_{ii}^- = \lambda_i \wedge 0.$$

We recall that the eigenvalues of the Roe matrix,  $\lambda_i$ , are the characteristic speeds of the system.

As the high-order flux we choose a centered central flux

$$F_{j+1/2}^{n,H} = \frac{f(U_j^n) + f(U_{j+1}^n)}{2},$$

which is expected to be second order accurate in smooth parts of the solution.

Under these assumptions, the flux limited numerical flux becomes

$$F_{j\pm 1/2}^n = \frac{f(U_j^n) + f(U_{j+1}^n)}{2} - \frac{1}{2} |\hat{A}| \Delta^+ U_j^n + \frac{1}{2} P_{j\pm 1/2} |\hat{A}| \Delta^+ U_j^n,$$

where we have introduced the matrix

$$P_{j+1/2} = S \Phi S^{-1}, \quad \Phi = \text{diag}(\phi(r_j^i)).$$

Here  $\phi(r)$  represents the scalar flux limiter discussed previously. Following the development for the scalar case, we have

$$r_j^i = \begin{cases} \frac{\langle s_i^{-1}, \Delta^- U_j^n \rangle}{\langle s_i^{-1}, \Delta^+ U_j^n \rangle}, & \lambda_i \geq 0, \\ \frac{\langle s_i^{-1}, \Delta^+ U_{j+1}^n \rangle}{\langle s_i^{-1}, \Delta^- U_{j+1}^n \rangle}, & \lambda_i \leq 0, \end{cases}$$

where  $s_i^{-1}$  is the  $i$ th row of  $S^{-1}$ . Hence,  $r_j^i$  is an indicator for the flux in characteristic form.

If we now consider  $\lambda_i \geq 0$ , we have

$$\langle s_i^{-1}, \Delta^+ U_j^n \rangle = \frac{1}{r_j^i} \langle s_i^{-1}, \Delta^- U_j^n \rangle.$$

Defining

$$\tilde{\Phi} = \text{diag} \left( \frac{\phi(r_j^i)}{r_j^i} \right),$$

this yields

$$\Phi_{ii} \langle s_i^{-1}, \Delta^+ U_j^n \rangle = \tilde{\Phi}_{ii} \langle s_i^{-1}, \Delta^- U_j^n \rangle,$$

and, subsequently,

$$\hat{A}^+ P \Delta^+ U_j^n = \hat{A}^+ \tilde{P} \Delta^- U_j^n,$$

where  $\tilde{P} = S \tilde{\Phi} S^{-1}$ . We note that these operators vary for every cell interface. In a

similar fashion, we recover

$$\hat{A}^- P \Delta^- U_j^n = \hat{A}^- \tilde{P} \Delta^+ U_j^n.$$

If we combine this with the scheme in (10.9), we recover

$$U_j^{n+1} = C_{-1} U_{j-1}^n + C_0 U_j^n + C_1 U_{j+1}^n,$$

with

$$\begin{aligned} C_{-1} &= -\frac{k}{2h} \left( -\hat{A}_{j-1/2}^+ (2I - P_{j-1/2}) - \hat{A}_{j+1/2}^+ \tilde{P}_{j+1/2} \right), \\ C_1 &= -\frac{k}{2h} \left( \hat{A}_{j+1/2}^- (2I - P_{j+1/2}) + \hat{A}_{j-1/2}^- \tilde{P}_{j-1/2} \right), \end{aligned}$$

and

$$\begin{aligned} C_0 &= I - \frac{k}{2h} \left( -2\hat{A}_{j+1/2}^- + 2\hat{A}_{j-1/2}^+ \right. \\ &\quad \left. + \hat{A}_{j+1/2}^+ \tilde{P}_{j+1/2} + \hat{A}_{j+1/2}^- P_{j+1/2} - \hat{A}_{j-1/2}^+ P_{j-1/2} - \hat{A}_{j-1/2}^- \tilde{P}_{j-1/2} \right). \end{aligned}$$

One easily realizes that condition (10.10) is satisfied and the symmetry of  $C_K$  follows from the symmetry of  $\hat{A}$ . Let us first establish positivity for  $C_{-1}$ . Since  $\hat{A}$  and  $P$  commute, the scalar-valued flux limiter satisfies (10.6), and  $\hat{A}^+$  is positive; this follows directly. The arguments for  $C_1 \geq 0$  are similar.

Let us now turn the attention to  $C_0$ . We have

$$C_0 \geq I - \frac{k}{2h} \left( -2\hat{A}_{j+1/2}^- + \hat{A}_{j+1/2}^+ \tilde{P}_{j+1/2} \right) - \frac{k}{2h} \left( 2\hat{A}_{j-1/2}^+ - \hat{A}_{j-1/2}^- \tilde{P}_{j-1/2} \right),$$

by definition of  $\hat{A}^\pm$ ,  $P_{j\pm 1/2}$ , and  $\tilde{P}_{j\pm 1/2}$ . By the bounds on the scalar flux limiter, positivity is guaranteed provided

$$\frac{k}{h} \max |\lambda_i| \leq \frac{1}{2}.$$

Second order accuracy in time can be achieved by combining this with a second order strongly stable scheme.

The scheme combining a Roe flux and a central flux through a scalar flux limiter is a positive scheme and, thus, stable. As for the TVD-stable schemes, this does not suffice to guarantee that an entropy solution is recovered.

Naturally, the construction discussed above is not unique, and other positive schemes are developed in [25, 24, 26]. ■

#### 10.1.4 • Flux limited schemes in action

In the following we evaluate the performance of the flux limited schemes through some examples. We focus on the TVD-stable schemes, and refer to [25, 24, 26] for a thorough

evaluation of the positive schemes. It should be noted that the implementation of the positive schemes is very similar to that of TVD-stable schemes.

Compared to previous schemes, the new element is the flux limiter function  $\phi$  defined in *FluxLimit.m*.

---

**Script 10.2. *FluxLimit.m*: Routine for defining flux limiter functions.**

---

```
function phi = FluxLimit(r,type,beta)
% function phi = FluxLimit(r,type,beta)
% Define flux limiter function based on type. Beta is used in some limiters
r = max(0,r);

% No flux limiter
if (type==0) phi = zeros(length(r),1); end
% Chakravarthy/Osher limiter
if (type==1) phi = min(r,beta); end
% Koren limiter
if (type==2) phi = min(min(2*r,(2+r)/3),2); end
% Sweby limiter
if (type==3) phi = max(min(beta*r,1),min(r,beta)); end
% OSPRE limiter
if (type==4) phi = 1.5*(r.^2+r)./(r.^2+r+1); end
% van Leer limiter
if (type==5) phi = 2*r./(1+r); end

phi = max(0,phi);
return
```

---

We consider schemes in which the low-order numerical flux is the Lax–Friedrichs flux and the high-order numerical flux is the Lax–Wendroff flux, while keeping in mind that the TVD-schemes are not limited to this combination. In smooth regions of the solution we should expect a second order accurate solution.

### Linear wave equation

Let us first consider the linear wave problem,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions, as discussed in subsection 1.4.1. Since the driver *LinwaveFLDriver1D.m* and the temporal integration routine *LinwaveFL1D.m* are essentially identical to those used for monotone schemes in subsection 5.3.2, they are not shown here. On the other hand, the evaluation of the right-hand side is unique and shown in *LinwaveFLrhs1D.m*, where the choice of the scalar flux limiter is also made. Since the test problem assumes a right propagating wave, only one indicator function  $r$  is required.

---

**Script 10.3. *LinwaveFLrhs1D.m*: Evaluation of the right-hand side of the linear wave equation using a flux limited scheme.**

---

```
function [du] = LinwaveFLrhs1D(x,u,h,k,maxvel)
% function [du] = LinwaveFLrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for linear wave equation using a
% flux limited scheme
N = length(x);
```

```

% Choose flux limiter = 0:LF; 1:CO; 2:Koren; 3:Sweby; 4:OSPRE; 5:van Leer
type = 5; beta=1.5;

% Periodic boundary conditions
[xe,ue] = extend(x,u,h,1,'P',0,'P',0);

% Compute indicator function and define flux limiter
r = (ue(2:N+1) - ue(1:N))./(ue(3:N+2)-ue(2:N+1));
[xe,re] = extend(x,r,h,1,'N',0,'N',0);
phiL = FluxLimit(re(1:N),type,beta);
phiR = FluxLimit(re(2:N+1),type,beta);

% Compute left flux in cell - Change numerical flux here
Fluxlow = LinwaveLF(ue(1:N),ue(2:N+1),k/h,maxvel);
Fluxhigh = LinwaveLW(ue(1:N),ue(2:N+1),k/h,maxvel);
FluxL = Fluxlow + phiL.* (Fluxhigh - Fluxlow);

% Compute right flux in cell - Change numerical flux here
Fluxlow = LinwaveLF(ue(2:N+1),ue(3:N+2),k/h,maxvel);
Fluxhigh = LinwaveLW(ue(2:N+1),ue(3:N+2),k/h,maxvel);
FluxR = Fluxlow + phiR.* (Fluxhigh - Fluxlow);

% Compute RHS
du = -(FluxR - FluxL)/h;
return

```

Figure 10.3 illustrates the results for the wave test problem at different resolutions and with different flux limiters. When compared to the results obtained with a monotone scheme, the improvements are substantial, and the flux limited scheme proves superior even on substantially coarser grids.

While the differences across the different flux limited schemes are less pronounced, we observe that the minmod limiter is the most dissipative. This is in agreement with the fact that the minmod limiter does not allow any artificial compression, as discussed in relation to Fig. 10.2. Other limiters yield very similar results for this case.

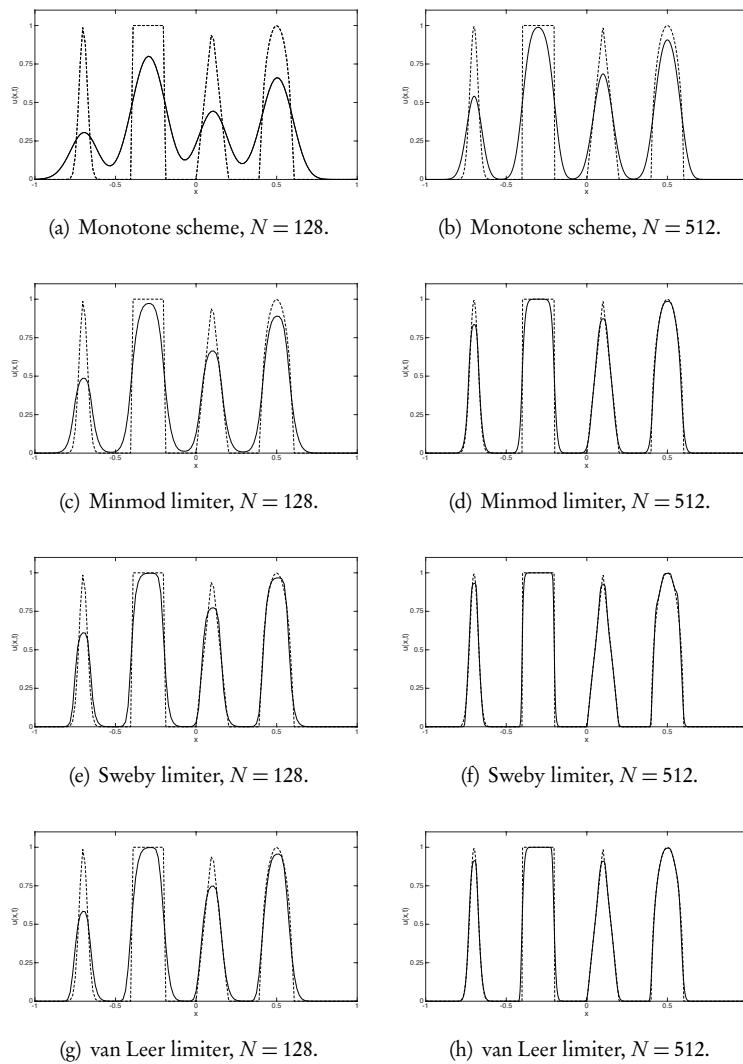
## Burgers' equation

Let us return to Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

discussed in detail in subsection 1.4.1. The driver BurgersFLDriver1D.m and the temporal integration routine BurgersFL1D.m are identical to the ones used for the monotone schemes, discussed in subsection 5.3.3, and are not shown here. In BurgersFLrhs1D.m we illustrate the evaluation of the right-hand side of the flux limited scheme for Burgers' equation. We notice the use of two indicator functions  $r$ , which depend on the direction of the flow. In particular, we define

$$r_j = \begin{cases} \frac{\Delta^- U_j^n}{\Delta^+ U_j^n}, & U_j^n \geq 0, \\ \frac{\Delta^+ U_{j+1}^n}{\Delta^- U_{j+1}^n}, & U_j^n \leq 0. \end{cases} \quad (10.11)$$



**Figure 10.3.** Solution (solid line) of the linear wave equation using TVD-stable flux limited schemes. All computations are shown at  $T = 4.0$  and computed with a CFL number of 0.9. The dashed line represents the exact solution.

**Script 10.4.** *BurgersFLrhs1D.m: Evaluation of the right-hand side of Burgers' equation using a flux limited scheme.*

---

```

function [du] = BurgersFLrhs1D(x,u,h,k,maxvel)
% function [du] = BurgersFLrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for Burgers equation using a
% flux limited scheme
N = length(x);

% Choose flux limiter = 0:LF; 1:CO; 2:Koren; 3:Sweby; 4:OSPRE; 5:van Leer
type = 5; beta=2;

```

```

% Boundary conditions
[xe,ue] = extend(x,u,h,1,'P',0,'P',0); % Periodic boundary conditions
%[xe,ue] = extend(x,u,b,1,'D',2,'N',0); % Constant boundary conditions

% Compute indicator function and define flux limiter
r = (ue(2:N+1) - ue(1:N))./(ue(3:N+2)-ue(2:N+1));
[xe,re] = extend(x,r,h,1,'N',0,'N',0); rm = 1./re;
phiLp = FluxLimit(re(1:N),type,beta);
phiRp = FluxLimit(re(2:N+1),type,beta);
phiLm = FluxLimit(rm(2:N+1),type,beta);
phiRm = FluxLimit(rm(3:N+2),type,beta);

ufilt = (u>=0);
phiL = ufilt.*phiLp + (1-ufilt).*phiLm;
phiR = ufilt.*phiRp + (1-ufilt).*phiRm;

% Compute left flux - Change numerical flux here
Fluxlow = BurgersLF(ue(1:N),ue(2:N+1),0,maxvel);
Fluxhigh = BurgersLW(ue(1:N),ue(2:N+1),k/h,maxvel);
FluxL = Fluxlow + phiL.* (Fluxhigh - Fluxlow);

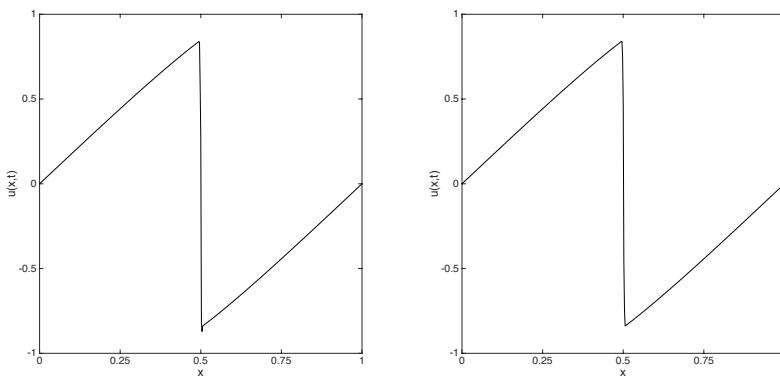
% Compute right flux - Change numerical flux here
Fluxlow = BurgersLF(ue(2:N+1),ue(3:N+2),0,maxvel);
Fluxhigh = BurgersLW(ue(2:N+1),ue(3:N+2),k/h,maxvel);
FluxR = Fluxlow + phiR.* (Fluxhigh - Fluxlow);

% Compute RHS
du = -(FluxR - FluxL)/h;
return

```

The importance of this distinction in the definition of the indicator function is illustrated in Fig. 10.4, where the left panel shows the results obtained for the stationary shock solution with the indicator function defined as

$$r_j = \frac{\Delta^- U_j^n}{\Delta^+ U_j^n},$$



**Figure 10.4.** Solution of Burgers' equation at  $T = 0.2$  using a TVD-stable flux limited scheme. On the left, one-way flux limiting is applied while the right shows limiting applied in an upwind fashion.

for all values of  $U_j^n$ . Close inspection reveals a small oscillation at the bottom of the shock. When the definition of the indication function follows (10.11), the oscillations are removed, as also shown in Fig. 10.4.

## Euler equations

As a final example, let us consider the Euler equations, discussed in detail in subsection 1.4.1. In EulerFLrhs1D.m we illustrate the evaluation of the right-hand side of the Euler equations, solved using a flux limited scheme. The driver EulerFLDriver1D.m and the temporal integration routine EulerFL1D.m are essentially identical to those for the monotone scheme in subsection 5.3.5 and, consequently, not shown here.

**Script 10.5. EulerFLrhs1D.m: Evaluation of the right-hand side of the Euler equations using a flux limited scheme based on the conserved variables.**

---

```

function [dq] = EulerFLrhs1D(x,q,gamma,h,k,maxvel)
% function [dq] = EulerFLrhs1D(x,q,gamma,h,k,maxvel);
% Purpose: Evaluate right hand side for Euler equations using a
% flux limited scheme
N = length(x);

% Choose flux limiter - 0:LF; 1:CO; 2:Koren; 3:Sweby; 4:OSPRE; 5:van Leer
type = 5; beta=1;

% Extend data and assign boundary conditions
[xe,re] = extend(x,q(:,1),h,1,'D',1.0,'D',0.125);
[xe,me] = extend(x,q(:,2),h,1,'D',0,'N',0);
[xe,Ee] = extend(x,q(:,3),h,1,'D',2.5,'N',0);

% [xe,re] = extend(x,q(:,1),h,1,'D',3.857143,'N',0);
% [xe,me] = extend(x,q(:,2),h,1,'D',10.141852,'D',0);
% [xe,Ee] = extend(x,q(:,3),h,1,'D',39.166661,'N',0);

% Compute indicator function and define flux limiter
rr = (re(2:N+1) - re(1:N))./(re(3:N+2)-re(2:N+1));
rm = (me(2:N+1) - me(1:N))./(me(3:N+2)-me(2:N+1));
rE = (Ee(2:N+1) - Ee(1:N))./(Ee(3:N+2)-Ee(2:N+1));

[xe,rre] = extend(x,rr,h,1,'N',0,'N',0);
[xe,rme] = extend(x,rm,h,1,'N',0,'N',0);
[xe,rEe] = extend(x,rE,h,1,'N',0,'N',0);

phirL = FluxLimit(rre(1:N),type,beta);
phirR = FluxLimit(rre(2:N+1),type,beta);
phimL = FluxLimit(rme(1:N),type,beta);
phimR = FluxLimit(rme(2:N+1),type,beta);
phiEL = FluxLimit(rEe(1:N),type,beta);
phiER = FluxLimit(rEe(2:N+1),type,beta);

% Compute fluxes
q = [re(2:N+1) me(2:N+1) Ee(2:N+1)];
qp = [re(3:N+2) me(3:N+2) Ee(3:N+2)];
qm = [re(1:N) me(1:N) Ee(1:N)];
FluxlowR = EulerLF(q,qp,gamma,k/h,maxvel);
FluxhighR = EulerLW(q,qp,gamma,k/h,maxvel);
FluxlowL = EulerLF(qm,q,gamma,k/h,maxvel);
FluxhighL = EulerLW(qm,q,gamma,k/h,maxvel);

FluxL(:,1) = FluxlowL(:,1) + phirL.* (FluxhighL(:,1) - FluxlowL(:,1));
FluxL(:,2) = FluxlowL(:,2) + phimL.* (FluxhighL(:,2) - FluxlowL(:,2));

```

```

FluxL (:,3) = FluxlowL (:,3) + phiEL.* (FluxhighL (:,3) - FluxlowL (:,3));
FluxR (:,1) = FluxlowR (:,1) + phirR.* (FluxhighR (:,1) - FluxlowR (:,1));
FluxR (:,2) = FluxlowR (:,2) + phimR.* (FluxhighR (:,2) - FluxlowR (:,2));
FluxR (:,3) = FluxlowR (:,3) + phiER.* (FluxhighR (:,3) - FluxlowR (:,3));

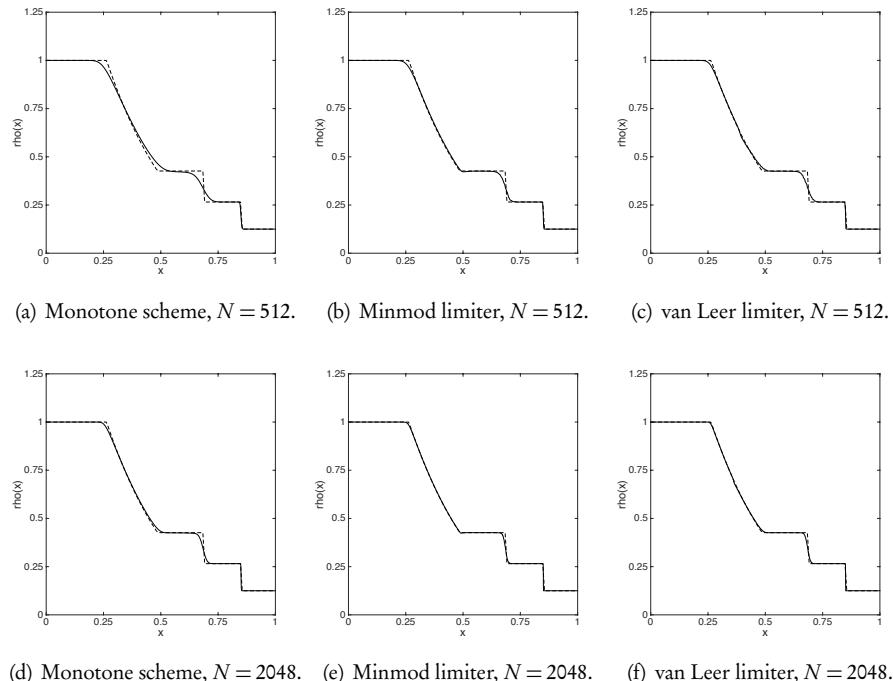
% Compute RHS
dq = - (FluxR - FluxL)/h;
return

```

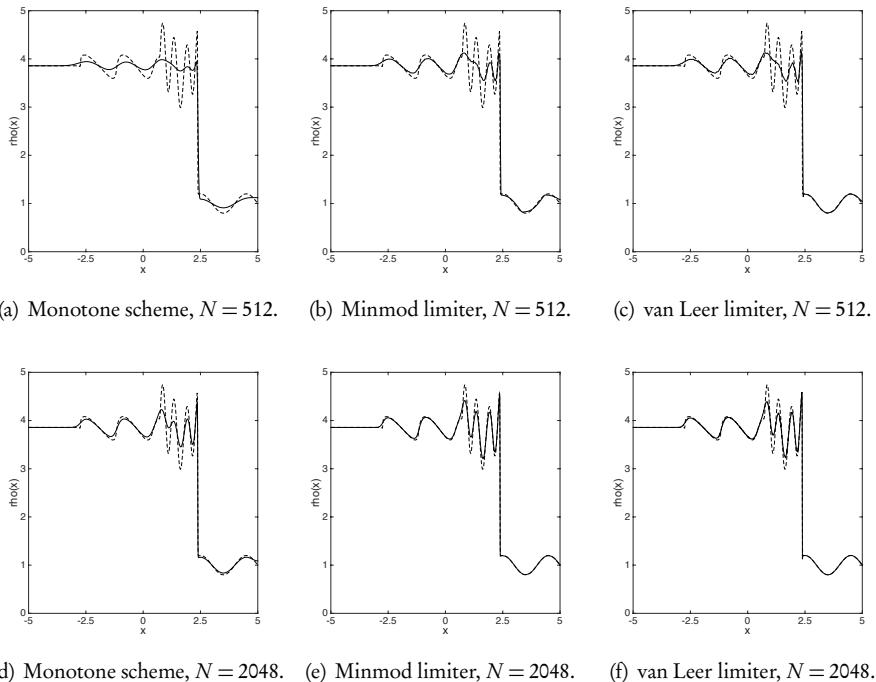
We note that the indicator function  $r_j$  assumes a right propagating wave. Furthermore, the flux limiting is based on the conserved variables and not, as one would perhaps expect, on the characteristic variables. Nevertheless, the numerical results for Sod's problem in Fig. 10.5 show excellent agreement with the exact solution and a clear improvement over a monotone scheme. This is particularly the case for the contact wave and the rarefaction wave. In spite of the relatively simple approach used for the flux limiting, there are no visible signs of oscillations.

When considering the shock entropy problem, Fig. 10.6 allows us to make similar qualitative conclusions. The results obtained with the flux limited scheme are clearly better than those with the monotone scheme, in particular in the resolution of the oscillations upstream of the shock.

Nevertheless, a close inspection of the flux limited solution at high resolution reveals a small oscillation at the bottom of the shock, qualitatively similar to what we observed for Burgers' equation. This suggests that a more careful approach is needed.



**Figure 10.5.** Computed density (solid line) of Euler equations for Sod's problem using a TVD-stable flux limited scheme. All computations are shown at  $T = 0.2$  and with a CFL number of 0.9. The dashed line represents the exact solution.



**Figure 10.6.** Computed density (solid line) of the Euler equations for the shock entropy problem using TVD-stable flux limited schemes. All computations are shown at  $T = 1.8$  and with a CFL number of 0.9. The dashed line represents the reference solution.

In order to get a better view of this, a detail of the solution computed at  $T = 0.7$  and with  $N = 2048$  is shown in Fig. 10.7. The small oscillation obtained with the flux limited scheme, based on a Chakravarthy–Osher limiter with  $\beta = 1.5$ , is similar to what is shown in Fig. 10.2, and is a result of an overly aggressive compression. This is confirmed by the results in Fig. 10.6, where the use of the minmod limiter allows no compression and exhibits no spurious oscillations. This conclusion is confirmed by decreasing  $\beta$  to 1.25, which results in an elimination of the oscillation, as shown in Fig. 10.7.

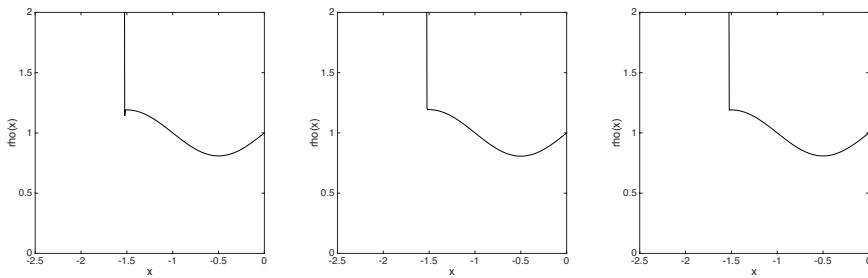
We previously observed that upwinding in the flux limiting eliminates minor oscillations. To pursue this approach for a system, we follow the discussion in Ex. 10.3 and define the flux limiter as

$$P_{j+1/2} = S \Phi S^{-1}, \quad \Phi = \text{diag}(\phi(r_j^i)),$$

where

$$\nabla_u f = A(u), \quad A = S \Lambda S^{-1},$$

is the characteristic decomposition of the flux Jacobian and  $\phi$  represents the scalar flux limiter. The indicator function  $r_j^i$  is defined in (10.11). While this approach is the natural generalization of the scalar case to the system case, it is computationally quite



**Figure 10.7.** Detail of computed density of the Euler equations for the shock entropy problem using a TVD-stable flux limited scheme. Results are shown at  $T = 0.7$ , obtained with  $N = 2048$  and a Chakravarthy–Osher flux limiter. On the left is shown the result with  $\beta = 1.5$  in the limiter, whereas the middle figure is obtained with  $\beta = 1.25$ . In both cases, limiting is based on the conserved variables as implemented in EulerFLrhs1D.m. The right figure is obtained with  $\beta = 1.5$ , but with limiting based on the characteristic decomposition at each cell interface, as implemented in EulerFLcharrhs1D.m.

expensive, as the characteristic decomposition is required at every cell interface. In EulerFLcharrhs1D.m we illustrate an implementation of this approach, based on the characteristic decomposition of the Euler equations discussed in Ex. 3.3 and expressed in EulerChar.m.

---

**Script 10.6. EulerFLcharrhs1D.m: Evaluation of the right-hand side of the Euler equations using a flux limited scheme based on the characteristic variables.**

---

```

function [dq] = EulerFLcharrhs1D(x,q,gamma,h,k,maxvel)
% function [dq] = EulerFLcharrhs1D(x,q,gamma,h,k,maxvel);
% Purpose: Evaluate right hand side for Euler equations using flux-limit
% scheme with limiting on characteristic variables
N = length(x); dq = zeros(N,3); phir = zeros(3,1);

% Choose flux limiter - 0:LF; 1:CO; 2:Koren; 3:Sweby; 4:OSPRE; 5:van Leer
type = 5; beta=1.5;

% Extend data and assign boundary conditions
[xe,re] = extend(x,q(:,1),h,2,'D',1.0,'D',0.125);
[xe,me] = extend(x,q(:,2),h,2,'D',0,'N',0);
[xe,Ee] = extend(x,q(:,3),h,2,'D',2.5,'N',0);

% [xe,re] = extend(x,q(:,1),h,2,'D',3.857143,'N',0);
% [xe,me] = extend(x,q(:,2),h,2,'D',10.141852,'D',0);
% [xe,Ee] = extend(x,q(:,3),h,2,'D',39.166661,'N',0);

qe = [re me Ee];
for i=1:N
    qloc = qe(i:i+4,:);
    % Left cell interface
    q0 = (qloc(2,:)+qloc(3,:))/2;
    [S, iS, Lam] = EulerChar(q0,gamma); Rloc = (iS*qloc)';
    for j=1:3
        if (Lam(j,j)>=0) % Upwind
            phir(j) = ...

```

```

    FluxLimit((Rloc(2,j)-Rloc(1,j))/(Rloc(3,j)-Rloc(2,j)),type, beta);
else % Downwind
    phir(j) = ...
    FluxLimit((Rloc(4,j)-Rloc(3,j))/(Rloc(3,j)-Rloc(2,j)),type, beta);
end
end
phiL = S*diag(phir(:))*iS;

% Right cell interface
q0 = (qloc(3,:)+qloc(4,:))/2;
[S, iS, Lam] = EulerChar(q0, gamma); Rloc = (iS*qloc')';
for j=1:3
    if (Lam(j,j)>=0) % Upwind
        phir(j) = ...
        FluxLimit((Rloc(3,j)-Rloc(2,j))/(Rloc(4,j)-Rloc(3,j)),type, beta);
    else % Downwind
        phir(j) = ...
        FluxLimit((Rloc(5,j)-Rloc(4,j))/(Rloc(4,j)-Rloc(3,j)),type, beta);
    end
end
phiR = S*diag(phir(:))*iS;

% Compute fluxes
q = [re(i+2) me(i+2) Ee(i+2)];
qp = [re(i+3) me(i+3) Ee(i+3)];
qm = [re(i+1) me(i+1) Ee(i+1)];
FluxlowR = EulerLF(q, qp, gamma, k/h, maxvel)';
FluxhighR = EulerLW(q, qp, gamma, k/h, maxvel)';
FluxlowL = EulerLF(qm, q, gamma, k/h, maxvel)';
FluxhighL = EulerLW(qm, q, gamma, k/h, maxvel)';
FluxL = FluxlowL + phiL*(FluxhighL - FluxlowL);
FluxR = FluxlowR + phiR*(FluxhighR - FluxlowR);

% Compute RHS
dq(i,:) = - (FluxR' - FluxL')/h;
end
return

```

---

**Script 10.7. EulerChar.m: Characteristic decomposition of the one-dimensional Euler equations.**

---

```

function [S, iS, Lam] = EulerChar(q0, gamma);
% Purpose: Compute characteristic decomposition for Euler equations at q0
r0 = q0(1); ru0 = q0(2); E0 = q0(3); u0 = ru0/r0;
p0 = (gamma-1)*(E0 - 0.5*ru0.^2/r0); c0 = sqrt(gamma*p0./r0);
H0 = (E0+p0)/r0;

S = zeros(3,3); iS = zeros(3,3); Lam = zeros(3,3);

S(1,1)=1/(2*c0); S(1,2)=1.0; S(1,3)=1/(2*c0);
S(2,1)=(u0+c0)/(2*c0); S(2,2)=u0; S(2,3)=(u0-c0)/(2*c0);
S(3,1)=(H0+c0*u0)/(2*c0); S(3,2)=u0.^2/2; S(3,3)=(H0-c0*u0)/(2*c0);

iS(1,1)=(0.5*(gamma-1)*u0.^2-c0*u0)/c0;
iS(1,2)=-(gamma-1)*u0-c0)/c0;
iS(1,3)=(gamma-1)/c0;
iS(2,1)=1-0.5*(gamma-1)*u0.^2/c0.^2;
iS(2,2)=(gamma-1)/c0.^2*u0;
iS(2,3)=-(gamma-1)/c0.^2;
iS(3,1)=(0.5*(gamma-1)*u0.^2+c0*u0)/c0;
iS(3,2)=-(gamma-1)*u0+c0)/c0;

```

```

iS (3,3)=(gamma-1)/c0;

Lam(1,1) = u0+c0; Lam(2,2) = u0; Lam(3,3) = u0-c0;
return

```

The results obtained with a Chakravarthy–Osher flux limiter with  $\beta = 1.5$ , shown in Fig. 10.7, confirm that the use of the characteristic variables introduces sufficient dissipation to eliminate the spurious oscillation. However, the computational cost is very substantial and raises the question of whether a combination of an increased resolution and a less aggressive flux limiter is, at least in this case, not to be preferred.

## 10.2 • Slope limited schemes

The high resolution flux limited schemes achieve high accuracy by carefully modifying the numerical flux. As discussed at the beginning of this chapter, this strategy is closely related to finite difference methods.

To recover high resolution schemes, closer in spirit to Godunov’s method, it is natural to also construct methods that rely on a careful reconstruction of the cell interface values.

### 10.2.1 • Monotone upstream-centered scheme for conservation laws (MUSCL)

One of the first high resolution schemes for conservation laws was proposed in a series of papers from 1974 to 1979 [43, 44, 45, 46, 47], with a precursor as early as 1972 [20], which presented a similar development that appears to have been unknown until recently [49].

The formulation of this method, now known as the monotone upstream-centered scheme for conservation laws (MUSCL), was recast and simplified [48]. Here we discuss that particular formulation.

The stating point is the assumption that the solution in each cell  $[x_{j-1/2}, x_{j+1/2}]$  can be expressed as a linear polynomial

$$u_j(x) = \bar{u}_j + \frac{x - x_j}{h} \delta u_j,$$

where  $\delta u_j/h$  is an approximation to the slope of the solution in the local cell. Clearly, if

$$\left| \frac{\delta u_j}{h} - \frac{du(x)}{dx} \Big|_{x_j} \right| \simeq \mathcal{O}(h),$$

one can hope to recover a scheme of second order accuracy. At the cell interfaces we recover the approximate solution values

$$u(x_{j\pm 1/2}) \simeq u_{j\mp 1/2}^\pm = \bar{u}_j \mp \frac{\delta u_j}{2},$$

where  $u^\pm$  refers to the value left ( $u^+$ ) and right ( $u^-$ ) of the cell center, respectively; e.g.,

$$u_{j+1/2}^\pm = \lim_{\varepsilon \rightarrow 0^+} u(x_{j+1/2} \pm \varepsilon).$$

Assuming that we have a good approximation to the local slope, a second order accurate scheme is obtained by first solving

$$\bar{u}_j^{n+1/2} = \bar{u}_j^n - \frac{k}{2b} [f(u_{j+1/2}^n) - f(u_{j-1/2}^n)] = \bar{u}_j^n - \frac{k}{2b} f'(\bar{u}_j^n) \delta u_j^n,$$

to obtain an approximation to  $\bar{u}(x_j, t^n + k/2)$ . With this, the solution values at the interfaces can be approximated as

$$u_{j\mp 1/2}^\pm = \bar{u}_j^{n+1/2} \mp \frac{\delta u_j^n}{2},$$

where the use of the original slope  $\delta u_j^n$  suffices for second order accuracy. The final stage is

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{b} [F(u_{j+1/2}^-, u_{j+1/2}^+) - F(u_{j-1/2}^-, u_{j-1/2}^+)],$$

where  $F(u, v)$  is a monotone flux. The use of central differences in time enables second order accuracy in both space and time, provided only that the slopes  $\delta u_j$  are chosen carefully.

Before proceeding, we observe that the numerical flux  $F(u, v)$  based on the solution of a Riemann problem should be modified to be fully consistent. The discussion of the solution to the Riemann problem in Chapter 3 was done under the assumption that the solution on either side of the interface is piecewise constant. However, in the above case, the solution is piecewise linear and solving the associated Riemann problem, known as the generalized Riemann problem, is considerably more complicated. Schemes based on solving this problem have been proposed [3, 2] but have received limited attention due to their complexity. Generally, the numerical flux is defined under the assumption that the simple Riemann problem with piecewise constant states is considered, and we also follow this approach here.

The central element of the scheme is the definition of the slope  $\delta u_j^n$ . In deciding how to approach this issue, we need to consider both accuracy and stability of the scheme, including TVD-stability. Keeping the stencil compact, a general expression is

$$\delta u_j^n = \frac{1}{2}(1+\omega)\Delta^- \bar{u}_j^n + \frac{1}{2}(1-\omega)\Delta^+ \bar{u}_j^n, \quad (10.12)$$

where  $\omega \in [-1, 1]$ . Taking  $\omega = 0$ , we recover an algebraic average while  $\omega = \pm 1$  yields full upwinding and downwinding, respectively. To understand the implications of the definition of the slope, let us consider a simple example.

**Example 10.4.** Consider the linear wave problem

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

subject to periodic boundary conditions. For simplicity, we assume that  $a > 0$ . The discussion for  $a < 0$  is similar.

Formulating a MUSCL scheme, we recover

$$u_{j-1/2}^+ = \bar{u}_j^n - \gamma_{j-1/2} \delta u_j^n, \quad u_{j+1/2}^- = \bar{u}_j^n + \gamma_{j+1/2} \delta u_j^n.$$

We have the two constants

$$\gamma_{j-1/2} = \frac{1+\lambda}{2}, \quad \gamma_{j+1/2} = \frac{1-\lambda}{2},$$

where  $\lambda = ak/h$  is the CFL number. We note that  $0 \leq \gamma \leq 1$  by the CFL condition.

Using upwinding in the numerical flux, we recover the scheme

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \lambda \left[ u_{j+1/2}^- - u_{j-1/2}^- \right]$$

which, when written out, yields

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \lambda \Delta^- \bar{u}_j^n - \lambda \gamma_{j+1/2} \Delta^- \delta u_j^n.$$

We observe that the additional term has a negative sign and, thus, reflects a compression term that may enable second order accuracy.

One easily shows that for  $\omega = -1$  in (10.12), the Lax–Wendroff scheme is recovered, while for  $\omega = 1$  the Beam–Warming scheme is obtained. The scheme for  $\omega = 0$  is known as the Fromm scheme [14], and is among the very first high resolution schemes proposed for solving wave equations. ■

The message from the above example is that the MUSCL scheme has the potential to yield second order accuracy in smooth regions. However, as we have seen previously, such schemes are not monotonicity preserving. Following the development of the flux limited schemes, we thus look for ways to modify the scheme to ensure that no artificial oscillations are introduced near discontinuities while enabling second order accuracy in smooth parts of the solution.

The following result is essential to enable the development of TVD-stable schemes.

**Theorem 10.5.** *If the following conditions hold*

$$\min(\bar{u}_{j-1}^n, \bar{u}_j^n) \leq u_{j-1/2}^+ \leq \max(\bar{u}_{j-1}^n, \bar{u}_j^n),$$

$$\min(\bar{u}_j^n, \bar{u}_{j+1}^n) \leq u_{j+1/2}^- \leq \max(\bar{u}_j^n, \bar{u}_{j+1}^n),$$

and the MUSCL scheme

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{h} \left[ F(u_{j+1/2}^-, u_{j+1/2}^+) - F(u_{j-1/2}^-, u_{j-1/2}^+) \right]$$

is advanced with a monotone flux, then the MUSCL scheme is TVD-stable.

**Proof.** It follows immediately from the properties of monotone schemes; see Theorem 4.5. □

To understand the implications of this result, let us return to the previous example.

**Example 10.6.** We consider the MUSCL scheme for the linear wave equation for which we have

$$u_{j-1/2}^+ = \bar{u}_j^n - \gamma_{j-1/2} \delta u_j^n, \quad u_{j+1/2}^- = \bar{u}_j^n + \gamma_{j+1/2} \delta u_j^n.$$

Theorem 10.5 guarantees that the scheme is TVD-stable provided

$$\begin{aligned}\Delta^- \bar{u}_j^n \geq 0 : 0 \leq \delta u_j^n \leq \beta_{j-1/2} \Delta^- \bar{u}_j^n, \\ \Delta^- \bar{u}_j^n \leq 0 : \beta_{j-1/2} \Delta^- \bar{u}_j^n \leq \delta u_j^n \leq 0, \\ \Delta^+ \bar{u}_j^n \geq 0 : 0 \leq \delta u_j^n \leq \beta_{j+1/2} \Delta^+ \bar{u}_j^n, \\ \Delta^+ \bar{u}_j^n \leq 0 : \beta_{j+1/2} \Delta^+ \bar{u}_j^n \leq \delta u_j^n \leq 0,\end{aligned}$$

where  $\beta_{j\pm 1/2} = (\gamma_{j\pm 1/2})^{-1} > 0$ .

To satisfy these conditions we must substitute the slope  $\delta u_j^n$  with the limited slope  $\overline{\delta u_j^n}$ . When  $(\Delta^- \bar{u}_j^n)(\Delta^+ \bar{u}_j^n) \leq 0$ , only  $\overline{\delta u_j^n} = 0$  satisfies the conditions. If  $\Delta^- \bar{u}_j^n$  and  $\Delta^+ \bar{u}_j^n$  are both positive, then

$$0 \leq \overline{\delta u_j^n} \leq \min(\beta_{j-1/2} \Delta^- \bar{u}_j^n, \beta_{j+1/2} \Delta^+ \bar{u}_j^n),$$

whereas if  $\Delta^- \bar{u}_j^n$  and  $\Delta^+ \bar{u}_j^n$  are both negative, we require

$$\max(\beta_{j-1/2} \Delta^- \bar{u}_j^n, \beta_{j+1/2} \Delta^+ \bar{u}_j^n) \leq \overline{\delta u_j^n} \leq 0.$$

By using the minmod function, the limited slope is

$$\overline{\delta u_j^n} = \text{minmod}(\beta_{j-1/2} \Delta^- \bar{u}_j^n, \beta_{j+1/2} \Delta^+ \bar{u}_j^n). \quad \blacksquare$$

Reaching back to the previous section, we recognize that we have introduced both a flux limiter,  $\phi(r)$ , and a limited slope,  $\overline{\delta u_j^n}$ . It seems of value to understand what, if any, connection exists between these two. Let us again address this through an example.

**Example 10.7.** We return to the problem in Ex. 10.4 and consider

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \lambda [F_{j+1/2}^n - F_{j-1/2}^n]$$

as an approximation of the linear wave equation with advection speed  $a$ .

In order to establish the connection, we consider the upwind biased schemes. For the flux limited scheme, an upwind flux is based on the numerical flux

$$F_{j+1/2} = \begin{cases} a \bar{u}_j^n + \phi(r_j) a \gamma_{j+1/2} \Delta^+ \bar{u}_j^n, & a > 0, \\ a \bar{u}_{j+1}^n - \phi(r_j) a \gamma_{j-1/2} \Delta^+ \bar{u}_j^n, & a < 0. \end{cases}$$

Similarly, for the slope limited scheme, we have the flux

$$F_{j+1/2} = \begin{cases} a \bar{u}_j^n + a \gamma_{j+1/2} \overline{\delta u_j^n}, & a > 0, \\ a \bar{u}_{j+1}^n - a \gamma_{j-1/2} \overline{\delta u_{j+1}^n}, & a < 0, \end{cases}$$

from which we recover

$$\phi(r_j) \Delta^+ \bar{u}_j^n = \begin{cases} \overline{\delta u_j^n}, & a > 0, \\ \overline{\delta u_{j+1}^n}, & a < 0. \end{cases} \quad (10.13)$$

For  $\phi(1) = 1$  we should recover the second order accurate Lax–Wendroff scheme ( $\omega = -1$  in (10.12)) which suggests that the slope  $\delta u_j$  must be chosen as

$$\delta u_j^n = \begin{cases} \Delta^+ \bar{u}_j^n, & a > 0, \\ \Delta^- \bar{u}_j^n, & a < 0. \end{cases}$$

Hence, the general form of the slope limiter should be

$$\overline{\delta u_j} = \psi(\Delta^- \bar{u}_j^n, \Delta^+ \bar{u}_j^n) = \begin{cases} \psi(r_j^{-1}) \Delta^- \bar{u}_j^n, & a > 0, \\ \psi(r_j) \Delta^+ \bar{u}_j^n, & a < 0, \end{cases} \quad (10.14)$$

where  $\psi(a, b)$  or  $\psi(r)$  is the slope limiter and

$$r_j = \frac{\Delta^- \bar{u}_j^n}{\Delta^+ \bar{u}_j^n}.$$

The minmod function (10.3) is an example of this. Comparing (10.13) and (10.14), it is clear that there is a direct relation between the two forms of limiting for the linear case. For the nonlinear case, this direct relation no longer exists. ■

With the insight gained in the previous example, we can define slope limiter functions  $\psi$  to ensure TVD-stability through (10.3).

The simplest minmod slope limiter is

$$\psi_{mm}(a, b) = \text{minmod}(a, b),$$

while a slightly generalized version, known as the MUSCL slope limiter, takes the form [49]

$$\psi_{mu}(a, b) = \text{minmod}\left(\frac{a+b}{2}, 2a, 2b\right).$$

The Superbee limiter [39, 34] is given as

$$\psi_{sb}(a, b) = \text{minmod}(\text{maxmod}(a, b), \text{minmod}(2a, 2b)),$$

where the maxmod function is defined analogously to the minmod function. Its implementation is illustrated in `maxmod.m`.

---

**Script 10.8. `maxmod.m`: Implementation of the maxmod function.**

---

```
function mfunc = maxmod(v)
% function mfunc = maxmod(v)
% Purpose: Implement the maxmod function on vector v
N = size(v, 1); m = size(v, 2); psi = zeros(N, 1);
s = sum(sign(v), 2)/m; ids = find(abs(s)==1);
if(~isempty(ids))
    psi(ids) = s(ids).*max(abs(v(ids ,:)), [], 2);
end
return;
```

---

Finally, an often used slope limiter is the van Albada limiter in the form [42]

$$\psi_{va}(a, b) = \frac{(a^2 + c^2)b + (b^2 + c^2)a}{a^2 + b^2 + 2c^2}.$$

Here  $c^2$  is a constant of  $\mathcal{O}(h^3)$ , used to control dissipation at local extrema. Further insight can be gained by writing this as [49]

$$\psi_{va}(r) = \frac{a+b}{2} \left( 1 - \frac{(a-b)^2}{a^2 + b^2 + 2c^2} \right),$$

which highlights that for the smooth case where  $a \simeq b$ , we recover the second order central difference approximation. We return to a more quantitative comparison of these different slope limiters in Chapter 10.2.3.

As discussed at length in subsection 8.2.1, requiring TVD-stability reduces the order of the scheme to first order at smooth extrema. In particular, we discussed that an  $\mathcal{O}(h^2)$  correction allows TVB-stability, yet is flexible enough to impact the accuracy a local extrema.

This can be achieved by a minor change in the slope limiter through the definition of the TVB minmod function [35, 9, 8]

$$\text{minmodTVB}(a_1, \dots, a_n, M, h) = \begin{cases} a_1, & |a_1| \leq Mh^2, \\ \text{minmod}(a_1, \dots, a_n), & \text{otherwise,} \end{cases} \quad (10.15)$$

where  $a_i$  represents the slopes and  $M$  is an estimate of the maximum curvature of the solution, in line with the discussion in subsection 8.2.1.

The estimate of  $M$  is delicate for a general solution, e.g., taking  $M$  too small results in an increasing dissipation at local extrema, while too high a value reintroduces oscillations. The implementation of the TVB minmod function is illustrated in `minmodTVB.m`.

---

**Script 10.9. *minmodTVB.m: Implementation of the TVB minmod function.***

---

```
function psi = minmodTVB(v, M, h)
% function psi = minmodTVB(v, M, h)
% Purpose: Implement the TVB modified midmod function on row vector v
psi = v(:, 1); ids = find(abs(psi) > M*h.^2);
if (size(ids, 1) > 0)
    psi(ids) = minmod(v(ids, :));
end
return
```

---

## 10.2.2 • Polynomial methods based on Lagrangian reconstruction

The fundamental idea of the MUSCL scheme is the reconstruction of a linear approximation to the solution  $u_j(x)$  for  $x \in [x_{j-1/2}, x_{j+1/2}]$ , appropriately limited to ensure TVD-stability. One can, at least in principle, extend this idea and seek a local approximation of higher polynomial degree with the intention of further increasing the order of the scheme and, hence, the accuracy in smooth parts of the solution. As we have already discussed, this ought to be pursued with care to ensure stability of the underlying scheme.

Let us recall the starting point of Godunov's method,

$$\bar{u}_j^{n+1} - \bar{u}_j^n = -\frac{1}{h} \int_{t^n}^{t^{n+1}} f(u(x_{j+1/2}, s), s) - f(u(x_{j-1/2}, s), s) ds,$$

which originates from the integral form of the conservation law. For the first order accurate Godunov's scheme, a simple approximation of the integral suffices, resulting in the conservation form.

However, if we seek higher order accuracy, the approximation has to be carried out with more care. In the MUSCL scheme, this is achieved by using the midpoint rule to approximate the solution at  $\bar{u}_j^{n+1/2}$ . Following a different approach [11, 10], one could also consider a trapezoidal approximation as

$$\int_{t^n}^{t^{n+1}} f(u(x_{j+1/2}, s), s) ds = \frac{k}{2} (f(u_{j+1/2}^n, t^n) + f(u_{j+1/2}^{n+1}, t^{n+1})).$$

This approach raises two questions: we need to clarify how  $u_{j+1/2}^n$  is obtained and, perhaps more importantly, how to recover  $f(u_{j+1/2}^{n+1}, t^{n+1})$  without formulating an implicit scheme. In order to answer these questions, let us focus on a scheme for the linear wave equation

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0,$$

subject to appropriate initial and boundary conditions.

To ensure second order accuracy, we consider the local approximation of the MUSCL scheme

$$u_j^n(x) = \bar{u}_j^n + \frac{x - x_j}{h} \overline{\delta u_j^n}, \quad (10.16)$$

where the limited slope ensures TVD-stability. We approximate  $f(u_{j+1/2}^n, t^n)$  by solving the Riemann problem

$$f(u_{j+1/2}^n, t^n) \simeq f(u^*(u_{j+1/2}^-, u_{j+1/2}^+)),$$

using the reconstructed interface values.

To recover  $f(u_{j+1/2}^{n+1}, t^{n+1})$ , we explore knowledge of the characteristics of the underlying equation, namely  $u(x, t^{n+1}) = u(x - ak, t^n)$ . The approach of tracing back along the characteristic is often referred to as Lagrangian. In this particular case, we have  $\lambda \leq 1$  for stability, so we are guaranteed to never cross cell boundaries. However, in a more general Lagrangian schemes, this is neither necessary nor desirable [38, 12, 33].

Hence, if we define  $\hat{x}_{j+1/2} = x_{j+1/2} - ak$  we recover

$$u(x_{j+1/2}, t^{n+1}) = u_j^n(\hat{x}),$$

where  $u_j^n(x)$  is given by (10.16). Once this is recovered, solving the Riemann problem at  $x_{j+1/2}$  yields an approximation to  $f(u_{j+1/2}^{n+1}, t^{n+1})$ , which completes the scheme.

This scheme is known as the piecewise linear method (PLM) and was first introduced in [10]. For the linear advection problem it is equivalent to the MUSCL scheme [40].

Its extension to nonlinear scalar problems is relatively straightforward. The only substantial difference is the Lagrangian step  $u(x, t^{n+1}) = u(x - f'(u)k, t^n)$ , which requires a nonlinear solution. This complicates matters in the case of nonlinear systems, where the use of characteristic variables becomes necessary. The details for the Euler equations can be found in [10].

The extension of this approach to schemes of higher than second order accuracy leads to a powerful and widely used method, known as the piecewise parabolic method (PPM), first proposed in [11]. The fundamental advance in this approach is a piecewise parabolic approximation to the local solution, which enables third order accuracy in smooth parts of the solution.

While the development in [11] concerns a general grid, we discuss the method under the assumption of an equidistant grid. We seek a locally reconstructed solution of the form

$$u_j^n(x) = u_{j-1/2}^+ + \tilde{x}(\Delta^+ u_{j-1/2}^n + \alpha(1 - \tilde{x})), \quad x \in [x_{j-1/2}, x_{j+1/2}], \quad (10.17)$$

where

$$\tilde{x} = \frac{x - x_{j-1/2}}{h}, \quad \Delta^+ u_{j-1/2}^n = u_{j+1/2}^- - u_{j-1/2}^+, \quad \alpha = 6 \left( \bar{u}_j^n - \frac{1}{2} (u_{j+1/2}^- + u_{j-1/2}^+) \right).$$

One easily proves for (10.17) that

$$\frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u_j^n(x) dx = \bar{u}_j^n.$$

The final step is to recover accurate values for  $u_{j\mp 1/2}^\pm$ . To achieve this, we seek a polynomial approximation to  $u$  in the neighborhood of  $x_j$ . Since the only information we have is the cell averages, this polynomial must recover the cell averages.

To achieve this, consider the primitive

$$U(x) = \int_x^{x_{j+1/2}} u(\xi, t^n) d\xi,$$

for which we have

$$U(x_{j+1/2}) = \sum_{l \leq j} \bar{u}_l^n h.$$

We now seek an interpolating polynomial of order four, using the five nodes  $x_{j+1/2+l}$ ,  $l = 0, \pm 1, \pm 2$ , and the associated  $U^n(x_{j+1/2+l})$ . The motivation for this construction becomes clear when we consider

$$\begin{aligned} \frac{1}{h} \int_{x_{j-1/2+l}}^{x_{j+1/2+l}} u(\xi, t^n) d\xi &= \frac{1}{h} \int_{x_{j-1/2+l}}^{x_{j+1/2+l}} U'(\xi) d\xi \\ &= \frac{1}{h} [U(x_{j+1/2+l}) - U(x_{j-1/2+l})] = \bar{u}_{j+l}^n, \end{aligned}$$

i.e., the cell average is maintained. Since  $U$  is an  $\mathcal{O}(h^4)$  approximation, differentiation suggests that

$$U'(x_{j+1/2}) = u(x_{j+1/2}, t^n) = u_{j+1/2}^n + \mathcal{O}(h^3)$$

in smooth regions of the solution.

Using this construction yields

$$u_{j+1/2}^- = u_{j+1/2}^+ = \bar{u}_j^n + \frac{1}{2} \Delta^+ \bar{u}_j^n - \frac{1}{6} (\sigma \bar{u}_{j+1}^n - \sigma \bar{u}_j^n),$$

where

$$\sigma \bar{u}_j^n = \frac{1}{2} (\Delta^+ \bar{u}_j^n + \Delta^- \bar{u}_j^n)$$

is the average slope in the cell.

Applying slope limiting on  $\sigma \bar{u}_j^n$  guarantees that  $u_{j\mp 1/2}^\pm$  lies between the cell averages and, thus, ensures TVD-stability. In [11] the MUSCL limiter is used although alternatives are possible.

The use of a quadratic polynomial within a cell enables the creation of new extrema within the cell, which destroys TVD-stability. Carefully considering the parabolic reconstruction [11] identifies two special cases where this is the case, and suggests a robust remedy.

1. If  $\bar{u}_j^n$  is either a local maximum or a local minimum,

$$u_{j\mp 1/2}^\pm = \bar{u}_j^n,$$

namely the solution is set to be cell average, thus reducing the local accuracy to first order.

2. If  $|\Delta^+ \bar{u}_{j-1/2}^n| \leq |\alpha|$ , then

$$\begin{aligned} u_{j-1/2}^+ &= 3\bar{u}_j^n - 2u_{j+1/2}^- && \text{if } \alpha \geq |\Delta^+ \bar{u}_{j-1/2}^n|, \\ u_{j+1/2}^- &= 3\bar{u}_j^n - 2u_{j-1/2}^+ && \text{if } \alpha \leq -|\Delta^+ \bar{u}_{j-1/2}^n|. \end{aligned}$$

In [11], this is combined with the MUSCL slope limiter on  $\sigma \bar{u}_j^n$  to yield a robust and accurate scheme. The extension to the system case is discussed in substantial detail in [11] and extensive computational results and comparative studies are provided in [53].

### 10.2.3 • Slope limited schemes in action

As for the flux limited schemes, let us evaluate the performance of the slope limited schemes through a few examples. We focus on the MUSCL scheme. For the Lagrangian scheme and its extensions to the nonlinear case and systems, we refer to [11, 10, 53].

The new element is the slope limiter function  $\psi(a, b)$  defined in `SlopeLimit.m`.

---

#### Script 10.10. *SlopeLimit.m: Routine for defining slope limiter functions.*

---

```
function psi = SlopeLimit(a,b,type,c,M,h)
% function psi = SlopeLimit(a,b,type,c,M,h)
% Define slope limiter function based on type. c is used in one limiter
% M,h is used for TVB limiting
N = length(a); zero = zeros(N,1);

% No slope
if (type==0) psi = zeros(N,1); end
% minmod limiter
if (type==1) psi = minmod([a b]); end
```

---

```

% MUSCL limiter
if (type==2) psi = minmod([(a+b)/2 2*a 2*b]); end
% Superbee limiter
if (type==3) psi = minmod([ maxmod([ a b]) minmod([ 2*a 2*b]) ]); end
% van Albada limiter
if (type==4) psi = minmod([( ((a.^2 + c.^2).*b + ...
(b.^2 + c.^2).*a)./(a.^2+b.^2+c.^2)) 2*a 2*b]); end
% van Leer limiter
if (type==5) psi = minmod([ 2.*a.*b./(a+b) 2*a 2*b ]); end
% TVB limiting
if (type==6) psi = minmodTVB([(a+b)/2 2*a 2*b],M,h); end
return

```

---

### Linear wave equation

We consider the linear wave problem,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions, as discussed in subsection 1.4.1. The driver `LinwaveSLDriver1D.m` and the temporal integration routine `LinwaveSL1D.m` are essentially identical to the ones used previously and discussed in subsection 5.3.2. These are not shown here. However, the evaluation of the right-hand side, shown in `LinwaveSLrhs1D.m`, is unique.

---

**Script 10.11. `LinwaveSLrhs1D.m`: Evaluation of the right-hand side of the linear wave equation using a slope limited scheme.**

---

```

function [du] = LinwaveSLrhs1D(x,u,h,k,maxvel)
% function [du] = LinwaveSLrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for the linear wave equation
% using a slope limited scheme based on MUSCL approach
N = length(x);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% Periodic boundary conditions
[xe,ue] = extend(x,u,h,2,'P',0,'P',0);

% Compute element slope and limit
dup = ue(3:N+4)-ue(2:N+3); dum = ue(2:N+3) - ue(1:N+2);
duL = SlopeLimit(dup,dum,type,c,M,h);

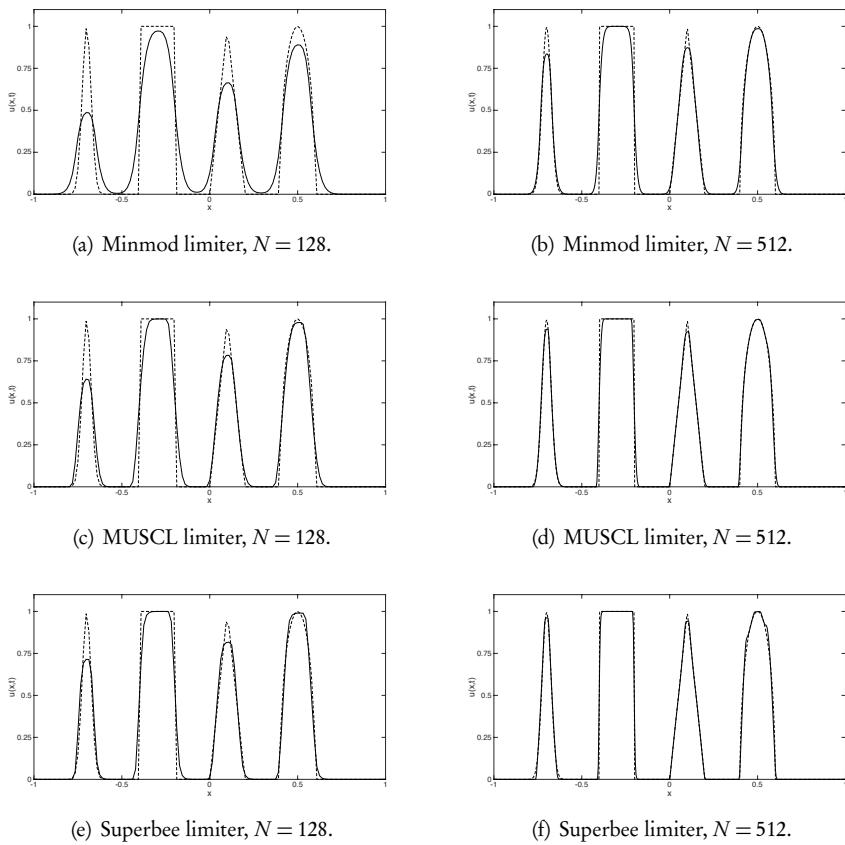
% Compute cell interface values - for f'(u) = 1;
uh = ue(2:N+3) - k/(2*h)*duL; uL = uh - duL/2; uR = uh + duL/2;

% Compute RHS
du = -(LinwaveLF(uR(2:N+1),uL(3:N+2),0,maxvel) ...
- LinwaveLF(uR(1:N),uL(2:N+1),0,maxvel))/h;
return

```

---

Figure 10.8 illustrates the performance of the scheme when solving the wave problem at different resolutions and with different slope limiters. The results for the monotone scheme can be found in Fig. 10.3. The results are similar to those obtained with the



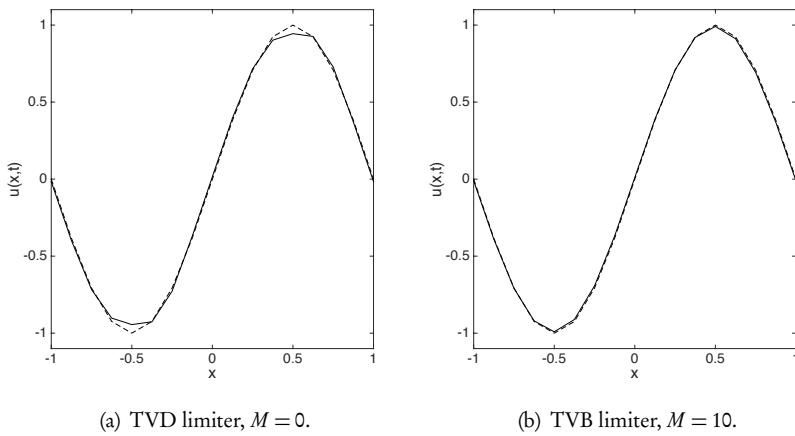
**Figure 10.8.** Solution (solid line) of the linear wave equation using TVD-stable slope limited schemes. All computations are shown at  $T = 4.0$  and obtained using a CFL number of 0.9. The dashed line represents the exact solution. For comparison, results with a monotone scheme can be found in Fig. 10.3.

flux limited scheme and are clearly superior to the results obtained with the monotone scheme. The results obtained with the Superbee limiter show effects of overcompression and a tendency to create square waves out of smooth waves. However, at increased resolution, this effect is reduced.

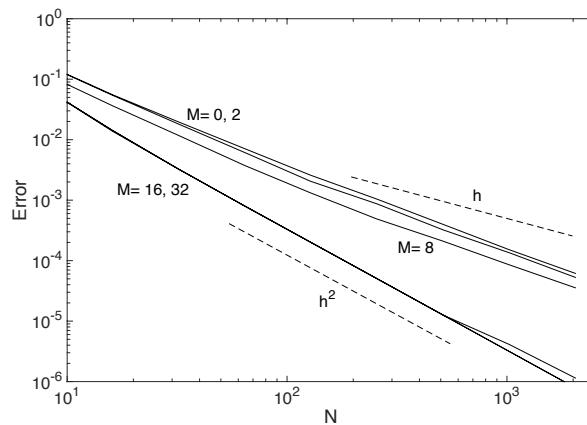
As discussed previously, the reduction of accuracy around local extrema can be addressed by considering a TVB-stable limiter. Let us first consider the wave equation with periodic boundary conditions and the initial condition  $u(x, 0) = \sin(\pi x)$ . Figure 10.9 illustrates the impact of TVB-limiting by comparing the results obtained with the TVD limiter. The improvement in local accuracy at the extrema is clear.

To quantify this improvement, we show the convergence in the maximum norm with increasing values of  $M$  in Fig. 10.10. It is clear that as  $M$  increases beyond the maximum curvature  $\pi^2$  of the smooth solution, full second order accuracy is recovered.

However, it is important to realize that the choice of  $M$  is a compromise and care must be exercised. In Fig. 10.11 we show part of the problem in Fig. 10.9 for increasing values of  $M$ . For  $M = 0$ , we recover the TVD-stable solution and notice substantial loss



**Figure 10.9.** Solution (solid line) of the linear wave equation using TVB slope limited scheme at  $T = 4$ . We use  $N = 16$ . The dashed line represents the exact solution.



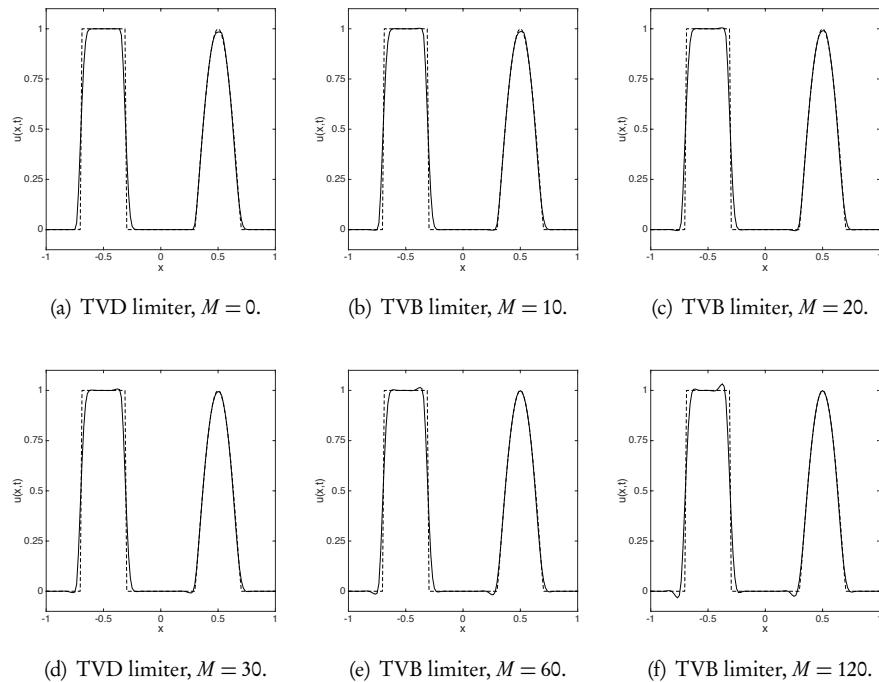
**Figure 10.10.** Maximum error for the linear wave equation when solved using a TVB slope limited scheme with different values of  $M$ .

of accuracy at the smooth extrema. As  $M$  increases, the accuracy at the smooth extrema visibly improves. However, taking too high values of  $M$  reintroduces oscillations, as would be the case for an unlimited scheme. For this test, the maximum curvature is  $(\pi/0.4)^2 \simeq 60$ .

### Burgers' equation

Let us also briefly consider a slope limited scheme for Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$



**Figure 10.11.** Solution (solid line) of the linear wave equation using a TVB-stable slope limiter based on the MUSCL limiter with different values of  $M$ . Final time is  $T = 4$  and  $N = 128$  cells are used.

discussed in subsection 1.4.1. The driver `BurgersSLDriver1D.m` and the temporal integration routine `BurgersSL1D.m` are identical to those for the monotone schemes, discussed in subsection 5.3.3, and they are not shown here. In `BurgersSLrhs1D.m`, we illustrate the evaluation of the right-hand side of the slope limited scheme for Burgers' equation.

**Script 10.12. `BurgersSLrhs1D.m`: Evaluation of the right-hand side for Burgers' equation using a slope limited scheme.**

```

function [du] = BurgersSLrhs1D(x,u,h,k,maxvel)
% function [du] = BurgersSLrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for Burgers equation using slope
% limited scheme
N = length(x);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% Boundary conditions
[xe,ue] = extend(x,u,h,2,'P',0,'P',0); % Periodic boundary conditions
%[xe,ue] = extend(x,u,h,2,'D',2,'N',0); % Constant boundary conditions

% Compute element slope and limit
dup = ue(3:N+4)-ue(2:N+3); dum = ue(2:N+3) - ue(1:N+2);

```

---

```

duL = SlopeLimit(dup,dum,type,c,M,h);

% Compute cell interface values - for  $f'(u) = 2u$ ;
uloc = ue(2:N+3);
uh = uloc - 2*uloc.*k/(2*h).*duL; uL = uh - duL/2; uR = uh + duL/2;

% Compute RHS
du = -(BurgersLF(uR(2:N+1),uL(3:N+2),0,maxvel) ...
        - BurgersLF(uR(1:N),uL(2:N+1),0,maxvel))/h;
return

```

---

Compared to the scheme for the linear wave equation, the differences are minor and the performance is indistinguishable from the results in Fig. 10.4.

### Euler equations

Finally, let us consider the solution of the Euler equations using a slope limited scheme. In this case, the MUSCL approach turns out to be problematic, since it requires the transport of the characteristic variables before limiting  $u_j^{n+1/2}$ . However, this is the step that enables second order temporal accuracy in the schemes considered so far.

To facilitate the temporal integration, we use a third order SSPERK(3,3) scheme, discussed in Chapter 9. This avoids the need to introduce the characteristic variables for the intermediate stage. The implementation is illustrated in EulerSL1D.m. The driver EulerSLDriver1D.m is similar to that of the monotone scheme, and is not shown here.

---

**Script 10.13. EulerSL1D.m: Advancing the slope limited scheme for the Euler equations.**

---

```

function [q] = EulerSL1D(x,q,h,CFL,gamma,FinalTime)
% function [q] = EulerFL1D(x,q,CFL,gamma,FinalTime)
% Purpose : Integrate 1D Euler equation until FinalTime using a
% slope limited scheme and SSP-RK3.
time = 0; tstep = 0;

% integrate scheme
while (time<FinalTime)
    % Set timestep
    p = (gamma-1)*(q(:,3) - 0.5*q(:,2).^2./q(:,1)); c = sqrt(gamma*p./q(:,1));
    maxvel = max(c+abs(q(:,2)./q(:,1))); k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    rhsq = EulerSLrhs1D(x,q,gamma,h,k,maxvel); q1 = q + k*rhsq;
    rhsq = EulerSLrhs1D(x,q1,gamma,h,k,maxvel); q2 = (3*q + q1 + k*rhsq)/4;
    rhsq = EulerSLrhs1D(x,q2,gamma,h,k,maxvel); q = (q + 2*q2 + 2*k*rhsq)/3;
    time = time+k; tstep = tstep+1;
end
return

```

---

Routine EulerSLrhs1D.m evaluates the right-hand side for the Euler equations using a slope limited scheme. We note that slope limiting is performed on the conserved variables as opposed to the flux limited case, which relies on the characteristics variables. One could perform the slope limiting on the characteristic variables although the benefits should be evaluated carefully against the considerable cost of this approach.

---

**Script 10.14. EulerSLrhs1D.m: Evaluation of the right-hand side for the Euler equations using a slope limited scheme.**

---

```

function [dq] = EulerSLrhs1D(x,q,gamma,h,k,maxvel)
% function [dq] = EulerSLrhs1D(x,q,gamma,h,k,maxvel);
% Purpose: Evaluate right hand side for the Euler equations using
% slope-limited method with limiting on characteristic variables
N = length(x); dq = zeros(N,3); qL = zeros(N+2,3); qR = zeros(N+2,3);
dum = zeros(N+2,3); duL = zeros(N+2,3); duR = zeros(N+2,3);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=h^3; M=150;

% Extend data and assign boundary conditions
[xe,re] = extend(x,q(:,1),h,2,'D',1.0,'D',0.125);
[xe,me] = extend(x,q(:,2),h,2,'D',0,'N',0);
[xe,Ee] = extend(x,q(:,3),h,2,'D',2.5,'N',0);

% [xe,re] = extend(x,q(:,1),h,2,'D',3.857143,'N',0);
% [xe,me] = extend(x,q(:,2),h,2,'D',10.141852,'D',0);
% [xe,Ee] = extend(x,q(:,3),h,2,'D',39.166661,'N',0);

% Extract variables
qe = [re me Ee];

% Compute left and right differences, evaluate slopes and interface values
dum = qe(3:N+4,:) - qe(2:N+3,:); duL = qe(2:N+3,:) - qe(1:N+2,:);
duL(:,1) = SlopeLimit(dup(:,1), dum(:,1), type, c, M, h);
duL(:,2) = SlopeLimit(dup(:,2), dum(:,2), type, c, M, h);
duL(:,3) = SlopeLimit(dup(:,3), dum(:,3), type, c, M, h);
qL = qe(2:N+3,:) - duL/2; qR = qe(2:N+3,:) + duL/2;

% Evaluate right hand side using numerical flux
dq = - (EulerLF(qR(2:N+1,:),qL(3:N+2,:),gamma, k/h, maxvel) - ...
          EulerLF(qR(1:N,:),qL(2:N+1,:),gamma, k/h, maxvel))/h;
return

```

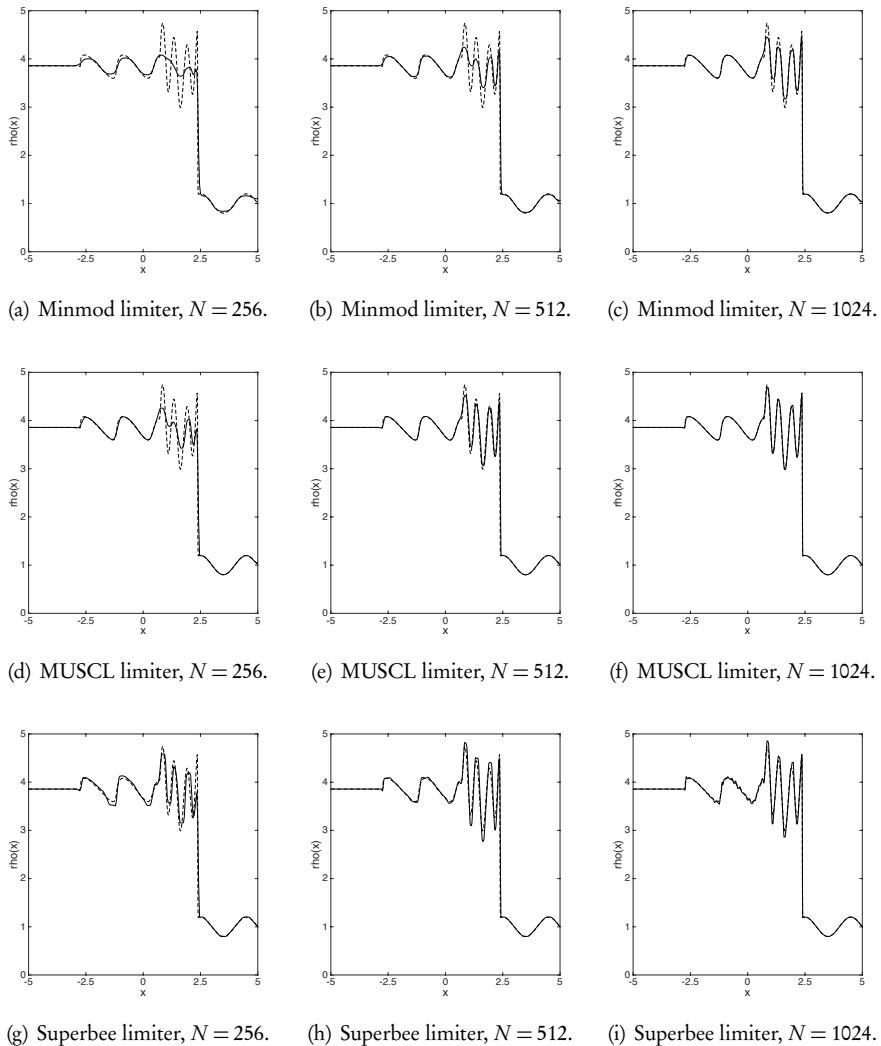
---

We consider the shock-entropy wave example and compare the results obtained with different choices of limiter and resolution in Fig. 10.12. We again observe substantial improvements over the results obtained with the monotone scheme; see Fig. 10.6. Also, we find that the results of the slope limited scheme appear superior to those obtained with the flux limited scheme. Furthermore, clear signs of overcompression, e.g., overshoots and flattening of smooth extrema, are observed when the Superbee limiter is used.

Finally, in Fig. 10.13 we show the impact of using the TVB-stable slope limiter with increasing values of  $M$ . When we increase  $M$ , the resolution of the smooth extrema improves, although artificial elements begin to emerge due to insufficient limiting. By further increasing the value of  $M$ , oscillations begin to appear, first observed at the bottom of the shock.

## 10.3 • Central schemes

Let us return to central schemes, introduced briefly in section 6.3, and discuss how to extend such schemes to higher-order accuracy. Since these schemes do not require a Riemann or an approximate Riemann solver, they are simple and have the potential to be efficient.



**Figure 10.12.** Computed density (solid line) of the Euler equations using TVD-stable slope limiters. The dashed line indicates the reference solution.

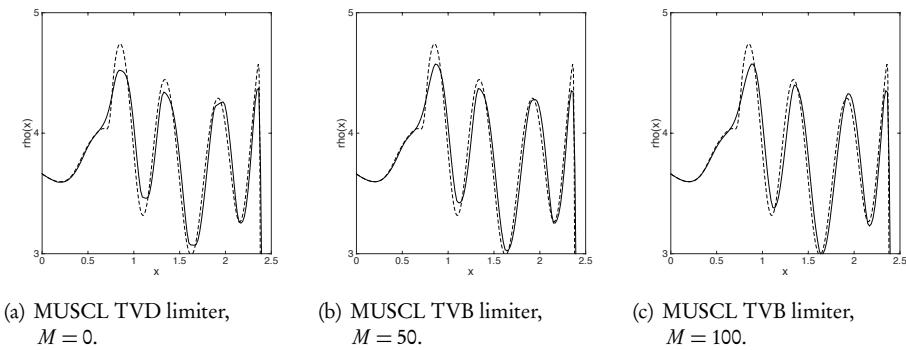
Recall the conservation law in integral form:

$$\bar{u}(x, t^{n+1}) - \bar{u}(x, t^n) = -\frac{1}{b} \left( \int_{t^n}^{t^{n+1}} f(u(x + h/2, s)) - f(u(x - h/2, s)) ds \right),$$

where

$$\bar{u}(x, t) = \frac{1}{b} \int_{x-h/2}^{x+h/2} u(s, t) ds$$

is a sliding cell average.



**Figure 10.13.** Detail of the computed density (solid line) of the Euler equations using a TVB-stable slope limiter based on the MUSCL limiter. The dashed line indicates the reference solution.  $N = 512$  cells are used.

To construct a scheme, we assume that  $\bar{u}(x, t)$  can be well approximated by a piecewise polynomial

$$\bar{u}(x, t) = \bigoplus_{j=1}^N \bar{u}_j(x, t) \simeq \bigoplus_{j=1}^N p_j(x) = \bar{u}_b(x, t),$$

where  $p_j$  is a polynomial defined locally on cell  $j$ . This yields the exact evolution of  $\bar{u}_b(x, t)$  as

$$\bar{u}_b(x, t^{n+1}) - \bar{u}_b(x, t^n) = -\frac{1}{b} \left( \int_{t^n}^{t^{n+1}} f(u_b(x + b/2, s)) - f(u_b(x - b/2, s)) ds \right). \quad (10.18)$$

To obtain a scheme, we must define a cell or, in other words, specify at which points  $x_j$  we require this integral expression to hold.

Choosing  $x_j$  as the cell centers, we recover the Godunov scheme

$$\bar{u}_j^{n+1} - \bar{u}_j^n = -\frac{1}{b} \left( \int_{t^n}^{t^{n+1}} f(u_{j+1/2}(s)) - f(u_{j-1/2}(s)) ds \right),$$

where  $u_{j\pm 1/2}(s)$  is recovered by solving the (generalized) Riemann problem with

$$u_{j+1/2}(x) = \begin{cases} p_j(x), & x \leq x_{j+1/2}, \\ p_{j+1}(x), & x \geq x_{j+1/2}. \end{cases}$$

As we have already experienced, the solution of this problem can be both complex and costly for general systems of conservation laws, even when an approximate Riemann solver is available.

The choice of points where (10.18) is enforced is, however, not unique, and different choices result in different schemes. In [29], a general family of central schemes is

introduced by choosing the cell interfaces  $x_{j+1/2}$  as identifiers, resulting in

$$\bar{u}_{j+1/2}^{n+1} - \bar{u}_{j+1/2}^n = -\frac{1}{b} \left( \int_{t^n}^{t^{n+1}} f(u_{j+1}(s)) - f(u_j(s)) ds \right).$$

This results in a staggered scheme since we advance the solution at  $x_{j+1/2}$  by using the solution at the cell centers. If we insist that the solution be piecewise polynomial in the cells, we obtain

$$\bar{u}_{j+1/2}^n = \frac{1}{b} \left( \int_{x_j}^{x_{j+1/2}} p_j(x) dx + \int_{x_{j+1/2}}^{x_{j+1}} p_{j+1}(x) dx \right).$$

The key difference between this scheme and the classic Godunov scheme is that we need to solve the local Riemann problem at the cell centers  $x_j$  to advance the solution. However, at the cell center the solution is expressed as a smooth polynomial  $p_j(x)$ , and its propagation can be done efficiently and without the need for a Riemann solver. This assumption of smoothness holds provided we choose  $k$  sufficiently small, i.e.,  $CFL \leq 1/2$ . In this case, discontinuities at  $x_{j\pm 1/2}$  never reach the cell center. This is the key idea behind the central scheme.

In the simplest case when the cellwise solution is simply a constant, we recover

$$\bar{u}_{j+1/2}^{n+1} = \bar{u}_{j+1/2}^n - \frac{k}{b} (f(\bar{u}_{j+1}^n) - f(\bar{u}_j^n)).$$

Realizing that

$$\bar{u}_{j+1/2}^n = \frac{1}{b} \left( \int_{x_j}^{x_{j+1/2}} p_j(x) dx + \int_{x_{j+1/2}}^{x_{j+1}} p_{j+1}(x) dx \right) = \frac{\bar{u}_{j+1}^n + \bar{u}_j^n}{2},$$

we obtain the first order scheme

$$\bar{u}_{j+1/2}^{n+1} = \frac{1}{2} (\bar{u}_{j+1}^n + \bar{u}_j^n) - \frac{k}{b} (f(\bar{u}_{j+1}^n) - f(\bar{u}_j^n)),$$

which is the Lax–Friedrichs scheme, discussed extensively in section 5.2.

The construction of higher-order accurate schemes follows the same approach. Let us assume that the local polynomial approximation to  $u_j$  is given as

$$p_j(x) = \bar{u}_j^n + \frac{1}{b} (x - x_j) \delta u_j^n,$$

where  $\delta u_j^n$  represents a local approximation to the slope of the solution, such that  $\delta u_j^n / b = d u / dx(x_j) + \mathcal{O}(b)$ . Then  $p_j$  is a second order accurate local approximation to  $u$  over  $[x_{j-1/2}, x_{j+1/2}]$ .

To recover a second order accurate scheme, we first consider

$$\bar{u}_{j+1/2}^n = \frac{1}{b} \left( \int_{x_j}^{x_{j+1/2}} p_j(x) dx + \int_{x_{j+1/2}}^{x_{j+1}} p_{j+1}(x) dx \right) = \frac{1}{2} (\bar{u}_j^n + \bar{u}_{j+1}^n) + \frac{1}{8} (\delta u_j^n - \delta u_{j+1}^n).$$

To recover an  $\mathcal{O}(k^2)$  accurate scheme we approximate the temporal integral by the midpoint rule,

$$\int_{t^n}^{t^{n+1}} f(u_j(s)) ds \simeq k f(u_j(t^{n+1/2})) + \mathcal{O}(k^2),$$

and use the idea of the MUSCL scheme to recover

$$u_j(t^n + k/2) \simeq \bar{u}_j^n + \frac{k}{2} \frac{d\bar{u}_j}{dt} = \bar{u}_j^n - \frac{k}{2} f'(\bar{u}_j^n) \frac{d\bar{u}_j}{dx},$$

where we use the conservation law. Using the local slope, this yields

$$u_j^{n+1/2} = \bar{u}_j^n - \frac{k}{2b} f'(\bar{u}_j^n) \delta u_j^n.$$

This first step, combined with

$$\bar{u}_{j+1/2}^{n+1} = \frac{1}{2} (\bar{u}_j^n + \bar{u}_{j+1}^n) + \frac{1}{8} (\delta u_j^n - \delta u_{j+1}^n) - \frac{k}{b} (f(\bar{u}_{j+1}^{n+1/2}) - f(\bar{u}_j^{n+1/2})), \quad (10.19)$$

yields the second order accurate central scheme, first proposed in [29]. It can be written in conservation form,

$$u_{j+1/2}^{n+1} = \bar{u}_j^n - \frac{k}{b} (F_{j+1}^n - F_j^n),$$

by introducing the numerical flux

$$F_j^n = f(u_j^{n+1/2}) + \frac{b}{2k} \left( \frac{1}{4} \delta u_j^n - \bar{u}_j^n \right).$$

As we have previously discussed, we must limit the slope  $\delta u_j^n$  to guarantee TVD-stability and prevent the introduction of oscillations at discontinuities.

TVD-stability is established in the following result [29].

**Theorem 10.8.** *Assume that  $\overline{\delta u_j^n}$  and  $f'(\bar{u}_j^n) \overline{\delta u_j^n}$  satisfy the conditions of Theorem 10.5, and that the time step  $k$  satisfies the CFL condition*

$$\frac{k}{b} \max_u |f'(u)| \leq C \leq \frac{1}{2};$$

*then the second order central scheme is TVD-stable.*

**Proof.** Let us first write (10.19) as

$$\bar{u}_{j+1/2}^{n+1} = \frac{1}{2} (\bar{u}_{j+1}^n + \bar{u}_j^n) - \lambda [G_{j+1}^n - G_j^n],$$

where we introduce the modified flux

$$G_j^n = f(u_j^{n+1/2}) + \frac{1}{8\lambda} \delta u_j^n,$$

with  $\lambda = k/b$ . We recall that

$$u_j^{n+1/2} = \bar{u}_j^n - \frac{k}{2b} f'(\bar{u}_j^n) \delta u_j^n.$$

Now consider

$$\bar{u}_{j+1/2}^{n+1} - \bar{u}_{j-1/2}^{n+1} = \left( \frac{1}{2} + \lambda \frac{\Delta^- G_j}{\Delta^- \bar{u}_j^n} \right) \Delta^- \bar{u}_j^n + \left( \frac{1}{2} - \lambda \frac{\Delta^+ G_j}{\Delta^+ \bar{u}_j^n} \right) \Delta^+ \bar{u}_j^n.$$

Recalling Theorem 8.4 and its proof, we immediately observe that TVD-stability is ensured provided

$$\lambda \left| \frac{\Delta^+ G_j}{\Delta^+ \bar{u}_j^n} \right| \leq \frac{1}{2}.$$

Consider

$$\begin{aligned} \lambda \left| \frac{\Delta^+ G_j}{\Delta^+ \bar{u}_j^n} \right| &\leq \lambda \left| \frac{\Delta^+ f(\bar{u}_j^{n+1/2})}{\Delta^+ \bar{u}_j^n} \right| + \frac{1}{8} \left| \frac{\Delta^+ \delta u_j^n}{\Delta^+ \bar{u}_j^n} \right| \\ &\leq \lambda \left| \frac{\Delta^+ f(\bar{u}_j^{n+1/2})}{\Delta^+ \bar{u}_j^{n+1/2}} \right| \left| \frac{\Delta^+ \bar{u}_j^{n+1/2}}{\Delta^+ \bar{u}_j^n} \right| + \frac{1}{8} \left| \frac{\Delta^+ \delta u_j^n}{\Delta^+ \bar{u}_j^n} \right|. \end{aligned} \quad (10.20)$$

The first term is controlled by the CFL condition

$$\frac{k}{h} \max_u |f'(u)| \leq C,$$

while the second term can be bounded as

$$\left| \frac{\Delta^+ \bar{u}_j^{n+1/2}}{\Delta^+ \bar{u}_j^n} \right| \leq 1 + \frac{\lambda}{2} \left| \frac{\Delta^+ f(\bar{u}_j^n)}{\Delta^+ \bar{u}_j^n} \right|.$$

We now apply the conditions for TVD-stability from Theorem 10.5 and require

$$0 \leq \overline{\delta u_j^n} \leq \alpha^{-1} \minmod(\Delta^- \bar{u}_j^n, \Delta^+ \bar{u}_j^n)$$

and

$$0 \leq |f'(\bar{u}_j^n)| \overline{\delta u_j^n} \leq \beta \minmod(\Delta^- \bar{u}_j^n, \Delta^+ \bar{u}_j^n),$$

from which it follows that

$$\left| \frac{\Delta^+ \delta u_j^n}{\Delta^+ \bar{u}_j^n} \right| \leq \alpha^{-1}, \quad \left| \frac{\Delta^+ f(\bar{u}_j^n)}{\Delta^+ \bar{u}_j^n} \right| \leq \beta.$$

Defining the CFL number  $C = \lambda \alpha \beta$ , we recover from (10.20) the condition

$$C \left( 1 + \frac{\lambda}{2} \beta \right) + \frac{1}{8} \alpha^{-1} = C \left( 1 + \frac{1}{2} \frac{C}{\alpha} \right) + \frac{1}{8} \alpha^{-1} \leq \frac{1}{2}.$$

This yields the condition on the CFL constant  $C$

$$C \leq -\alpha + \sqrt{\alpha^2 + \alpha - 1/4}.$$

The CFL constant  $C$  is positive for  $\alpha > 1/4$  and  $\beta > 0$ . Note that in the special case of  $\alpha^{-1} = 0$ , we recover the Lax–Friedrichs scheme and  $C \leq 1/2$ , as expected.  $\square$

Thus, when limited slopes are chosen appropriately, the second order central scheme is TVD-stable. However, as we have previously discussed, this does not suffice to guarantee convergence to the entropy solution. In [29] a cell entropy condition

$$\eta(\bar{u}_{j+1/2}^{n+1}) \leq \frac{1}{2} (\eta(\bar{u}_{j+1}^n) + \eta(\bar{u}_j^n)) - \lambda(\Psi_{j+1}^n - \Psi_j^n)$$

is established for the scalar conservation law under slightly stronger conditions than those for TVD-stability. Here,  $\Psi_j^n$  represents a consistent entropy flux. This establishes the following result [29].

**Theorem 10.9.** *Consider the approximation of the scalar conservation law by a central scheme, satisfying the conditions for TVD-stability and the existence of a local cell entropy condition. Provided the CFL condition holds, the central scheme converges to the unique entropy solution of the conservation law.*

Before continuing, let us briefly return to the first order scheme for the linear wave equation

$$\bar{u}_{j+1/2}^{n+1} = \frac{1}{2} (\bar{u}_{j+1}^n + \bar{u}_j^n) - \lambda(\bar{u}_{j+1}^n - \bar{u}_j^n),$$

$\lambda = ak/h$ , and consider the modified equation to leading order

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = \frac{1}{8} \frac{h}{\lambda} \frac{\partial^2 u}{\partial x^2}.$$

The dissipative term comes from the averaging term. We note that the magnitude of the dissipation depends on  $\lambda^{-1}$ , i.e., to reduce dissipation, one should maximize the CFL number. For the second order scheme, the first two terms result in a term like  $-Ch^3 \lambda^{-1} u^{(4)}$ , expressing a similar behavior. Such dissipative effects can be substantial for problems requiring time steps well below the CFL limit, e.g., for problems where a stiff source requires a small time step. There are several techniques, based on a more elaborate problem-dependent definition of the boundaries of the volume in (10.18), to overcome this [23].

While the scheme is convergent and simple to implement, a disadvantage is the need to account for solutions at two sets of grids,  $x_j$  and  $x_{j+1/2}$ . We already touched on this in section 6.3 for the first order schemes. However, to recover a nonstaggered version of higher-order accuracy, we must be more careful to avoid a reduction of the order of the scheme. Following [18], we define

$$u_h^{n+1}(x) = \bigoplus_{j=1}^N p_{j+1/2}^{n+1}(x),$$

for which  $p_{j+1/2}^{n+1}(x_{j+1/2}) = u_{j+1/2}^{n+1}$ . The average of arbitrary order is then given as

$$\bar{u}_j^{n+1} = \frac{1}{b} \left[ \int_{x_{j-1/2}}^{x_j} p_{j-1/2}^{n+1}(x) dx + \int_{x_j}^{x_{j+1/2}} p_{j+1/2}^{n+1}(x) dx \right].$$

To recover a second order accurate nonstaggered scheme, we consider

$$p_{j+1/2}^{n+1}(x) = \bar{u}_{j+1/2}^{n+1} + \frac{x - x_{j+1/2}}{h} \delta u_{j+1/2}^{n+1},$$

and obtain the nonstaggered version of the second order accurate central scheme as

$$\begin{aligned} \bar{u}_j^{n+1} &= \frac{1}{h} \left( \int_{x_{j-1/2}}^{x_j} p_{j-1/2}^{n+1}(x) dx + \int_{x_j}^{x_{j+1/2}} p_{j+1/2}^{n+1}(x) dx \right) \\ &= \frac{1}{2} (\bar{u}_{j-1/2}^{n+1} + \bar{u}_{j+1/2}^{n+1}) - \frac{1}{8} (\delta u_{j+1/2}^{n+1} - \delta u_{j-1/2}^{n+1}). \end{aligned}$$

This yields the scheme

$$\begin{aligned} \bar{u}_j^{n+1} &= \frac{1}{4} (\bar{u}_{j+1}^n + 2\bar{u}_j^n + \bar{u}_{j-1}^n) - \frac{1}{16} (\delta u_{j+1}^n - \delta u_{j-1}^n) \\ &\quad - \frac{k}{2h} (f(\bar{u}_{j+1}^{n+1/2}) - f(\bar{u}_{j-1}^{n+1/2})) - \frac{1}{8} (\delta u_{j+1/2}^n - \delta u_{j-1/2}^n). \end{aligned}$$

This approach, which leads to the second order central scheme, can also be pursued to recover central schemes of third order, provided the temporal integration is done using RKSSP schemes. Such developments are discussed in detail in [27], and the extension of the central scheme to multidimensional problems is addressed in [19] and also discussed in section 10.4.

### 10.3.1 • Central schemes in action

Let us now turn to a more practical evaluation of the central schemes and focus on the second order accurate method. As compared to the nonstaggered scheme, a slightly modified approach is required to impose boundary conditions on the staggered solution. This issue arises since no solution is known exactly at the boundaries, and imposing the boundary conditions must be altered, as illustrated in `extendstag.m`.

**Script 10.15. `extendstag.m`: Extension of staggered data to impose boundary conditions.**

---

```
function [xe,ue] = extendstag(x,u,h,m,BCl,ul,BCr,ur)
% Purpose: Extend dependent and independent vectors (x,u), by m cells
% subject to appropriate boundary conditions. Grid is assumed staggered
% BC = "D" - Dirichlet; BC = "N" - Neumann; BC = "P" - periodic
% ul/ur - BC value - only active for Dirichlet BC

x1 = min(x); xr = max(x); N = length(u);
xe = zeros(N+2*m,1); ue = zeros(N+2*m,1); q = [1:m];

% Extend x
xe(m-q+1) = x1-q*h; xe(N+m+q) = xr + q*h; xe((m+1):(N+m)) = x(1:N);

% Periodic extension of u
if (BCl=='P') | (BCr=='P')
    ue(m-q+1) = u(N-q+1); ue(N+m+q) = u(q); ue((m+1):(N+m)) = u(1:N);
    return;
end;

% Left extension
```

*To download the MATLAB code for this section, go to the SIAM website at <http://www.siam.org/journals/ojsa.php>.*

```

if BCl=='D'
    ue(m-q+1) = -u(q)+2*u1;
else
    ue(m-q+1) = u(q);
end

% Right extension
if BCr=='D'
    ue(N+m+q) = -u(N-q+1)+2*ur;
else
    ue(N+m+q) = u(N-q+1);
end
ue((m+1):(N+m)) = u(1:N);
return

```

To simplify matters, we take two steps of size  $k/2$  during the evaluation of the right-hand side and implement the shifts between the nonstaggered and the staggered grid within these two steps. Apart from the evaluation of the right-hand side, the staggered grid is not visible.

Also, to conform to the algorithmic style used so far, we return the temporal derivative from the routines rather than the advanced solution. This is a choice of style only.

### Linear wave equation

Consider the linear wave problem,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions, as discussed in subsection 1.4.1. The driver LinwaveCDriver1D.m and the temporal integration routine LinwaveC1D.m are essentially identical to the ones used previously, discussed in subsection 5.3.2, and are not shown here. The evaluation of the right-hand side, implementing the central scheme, is shown in LinwaveCrhs1D.m, where the choice of the slope limiter is also made.

**Script 10.16. LinwaveCrhs1D.m: Evaluation of the right-hand-side when solving the linear wave equation using a second order accurate central scheme on a staggered grid.**

```

function [du] = LinwaveCrhs1D(x,u,h,k,maxvel)
% function [du] = LinwaveCrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for linear wave equation using 2nd
% order central scheme - NOTE two steps of k/2 is taken
N = length(x); Ns = N-1; k1=k/2; us = zeros(Ns,1); uns = zeros(N,1);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% First step from non-staggered to staggered grid
[xe,ue] = extend(x,u,h,2,'P',0,'P',0); % Periodic BC

% Compute element slope and limit
dup = ue(3:N+4)-ue(2:N+3); dum = ue(2:N+3) - ue(1:N+2);
duL = SlopeLimit(dup,dum,type,c,M,h);

```

```

% Compute limited intermediate solution
uh = ue(2:N+3) - k1/(2*h)*duL;

% Advance solution k/2
us = (u(1:N-1)+u(2:N))/2 + (duL(2:N)-duL(3:N+1))/8 - ...
      k1/h*(uh(3:N+1)-uh(2:N));

% Second step from staggered to non-staggered grid
[xe,ue] = extendstag(x(1:Ns)+h/2,us,h,2,'P',0,'P',0); % Periodic BC

% Compute element slope and limit
dup = ue(3:Ns+4)-ue(2:Ns+3); dum = ue(2:Ns+3) - ue(1:Ns+2);
duL = SlopeLimit(dup,dum,type,c,M,h);

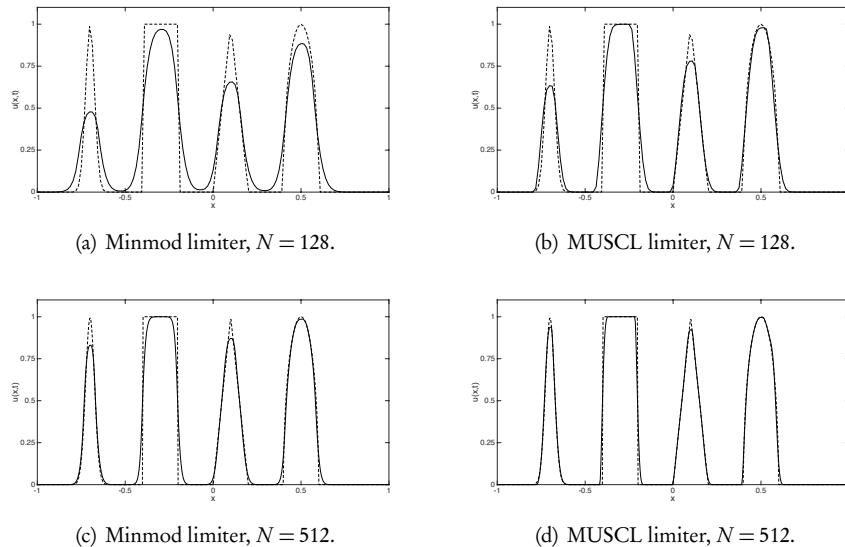
% Compute limited intermediate solution
uh = ue(2:Ns+3) - k1/(2*h)*duL;

% Advance solution k/2
uns = (ue(2:Ns+2)+ue(3:Ns+3))/2 + (duL(1:Ns+1)-duL(2:Ns+2))/8 - ...
      k1/h*(uh(2:Ns+2)-uh(1:Ns+1));

% Restore residual
du = (uns-u)/k;
return

```

Figure 10.14 illustrates the performance of the scheme for solving the wave problem at different resolutions and with the use of different slope limiters. The results for the monotone scheme can be found in Fig. 10.3. The results are similar to those obtained with the flux limited scheme and they are clearly superior to the results obtained with



**Figure 10.14.** Solution (solid line) of the linear wave equation using a second order accurate central scheme. All computations are shown at  $T = 4.0$  and obtained with a CFL number of 0.9. The dashed line represents the exact solution. For comparison, results with a monotone scheme can be found in Fig. 10.3.

the monotone scheme. In particular, the results compared with the more aggressive MUSCL limiter are superior to most previous results.

### Burgers' equation

Let us briefly consider a central scheme for Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

introduced in subsection 1.4.1. As compared to the linear wave equation, the changes needed to solve Burgers' equation are minimal and only involve the nonlinear flux and the flux Jacobian needed in the MUSCL step. Their implementations are shown in BurgersFlux.m and BurgersJac.m. These two elements are the only pieces that require a change if a different scalar equation is to be solved. This highlights the simplicity and flexibility of the central schemes.

---

**Script 10.17. BurgersFlux.m: Evaluation of the flux of Burgers' equation.**

---

```
function [ fu ] = BurgersFlux(u);
% function [fu] = BurgersFlux(u);
% Purpose: Evaluate flux for Burgers
fu = u.^2;
return
```

---



---

**Script 10.18. BurgersJac.m: Evaluation of the flux Jacobian for Burgers' equation.**

---

```
function [ fu ] = BurgersJac(u);
% function [fu] = BurgersJac(u);
% Purpose: Evaluate Jacobian for Burgers' flux
fu = 2*u;
return
```

---

In BurgersCrhs1D.m we show the evaluation of the right-hand side, which implements the central scheme across two steps of size  $k/2$ .

---

**Script 10.19. BurgersCrhs1D.m: Evaluation of the right-hand side for Burgers' equation, solved using a second order accurate central scheme on a staggered grid.**

---

```
function [ du ] = BurgersCrhs1D(x,u,h,k,maxvel)
% function [du] = BurgersCrhs1D(x,u,h,k,maxvel);
% Purpose: Evaluate right hand side for Burgers equation using second order
% central scheme - NOTE two steps of k/2 is taken
N = length(x); Ns = N-1; kl=k/2; us = zeros(Ns,1); uns = zeros(N,1);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% First step from non-staggered to staggered grid
[xe,ue] = extend(x,u,h,2,'P',0,'P',0); % Periodic BC
[%xe,ue] = extend(x,u,h,2,'D',2,'N',0); % Constant BC
```

```

% Compute element slope and limit
dup = ue(3:N+4)-ue(2:N+3); dum = ue(2:N+3) - ue(1:N+2);
duL = SlopeLimit(dup,dum,type,c,M,h);

% Compute limited intermediate solution - f'(u)=2*u
uh = ue(2:N+3) - BurgersJac(ue(2:N+3)).*k1/(2*h).*duL;

% Advance solution k/2
us = (u(1:N-1)+u(2:N))/2 + (duL(2:N)-duL(3:N+1))/8 ...
- k1/h*(BurgersFlux(uh(3:N+1))-BurgersFlux(uh(2:N)));

% Second step from staggered to non-staggered grid
[xe,ue] = extendstag(x(1:Ns)+h/2,us,h,2,'P',0,'P',0); % Periodic BC
%[xe,ue] = extendstag(x(1:Ns)+h/2,us,h,2,'D',2,'N',0); % Constant BC

% Compute element slope and limit
dup = ue(3:Ns+4)-ue(2:Ns+3); dum = ue(2:Ns+3) - ue(1:Ns+2);
duL = SlopeLimit(dup,dum,type,c,M,h);

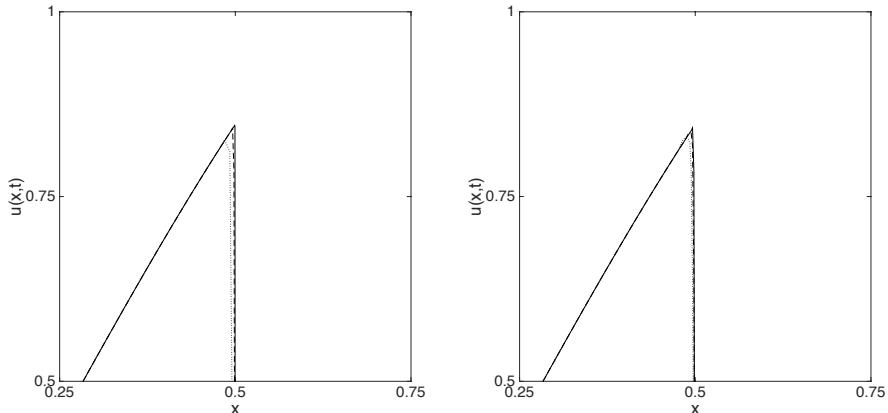
% Compute limited intermediate solution - f'(u)=2*u
uh = ue(2:Ns+3) - BurgersJac(ue(2:Ns+3)).*k1/(2*h).*duL;

% Advance solution k/2
uns = (ue(2:Ns+2)+ue(3:Ns+3))/2 + (duL(1:Ns+1)-duL(2:Ns+2))/8 ...
- k1/h*(BurgersFlux(uh(2:Ns+2))-BurgersFlux(uh(1:Ns+1)));

% Restore residual
du = (uns-u)/k;
return

```

Figure 10.15 shows the results obtained with different resolutions, confirming convergence of the discontinuous solution without the introduction of any oscillations. It also highlights that reducing the time step results in the expected increase in dissipation.



**Figure 10.15.** Solution of Burgers' equation using a central second order accurate scheme. On the left, we show a part of the solution at  $T = 0.2$  obtained with  $N = 128$  (dotted line),  $N = 512$  (dashed line) and  $N = 2048$  (solid line), illustrating the convergence. On the right, we show the solution for  $N = 512$  with  $CFL = 0.9$  (solid lined),  $CFL = 0.09$  (dashed line) and  $CFL = 0.009$  (dotted line), illustrating the increased dissipation with a decreasing CFL number.

## Euler equations

Let us also consider the solution of the Euler equations, solved using the second order central scheme. The implementation of this is illustrated in EulerCrhs1D.m. The driver EulerCDriver1D.m and the temporal integration routine EulerC1D.m is similar to what is used for the monotone scheme and these are not shown here.

**Script 10.20. EulerCrhs1D.m: Evaluation of the right-hand side for solving the Euler equations using a second order accurate central scheme on a staggered grid.**

```

function [dq] = EulerCrhs1D(x,q,gamma,h,k,maxvel)
% function [dq] = EulerCrhs1D(x,q,gamma,h,k,maxvel);
% Purpose: Evaluate right hand side for Euler equations using second order
% central scheme - NOTE two steps of k/2 is taken
N = length(x); dq = zeros(N,3); qh = zeros(N+2,3);
Ns = N-1; kl=k/2; duL = zeros(N+2,3); duLs = zeros(Ns+2,3);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% First step from non-staggered to staggered grid
% Extend data and assign boundary conditions
[xe,re] = extend(x,q(:,1),h,2,'D',1.0,'D',0.125);
[xe,me] = extend(x,q(:,2),h,2,'D',0,'N',0);
[xe,Ee] = extend(x,q(:,3),h,2,'D',2.5,'N',0);

% [xe,re] = extend(x,q(:,1),h,2,'D',3.857143,'N',0);
% [xe,me] = extend(x,q(:,2),h,2,'D',10.141852,'D',0);
% [xe,Ee] = extend(x,q(:,3),h,2,'D',39.166661,'N',0);

% Extract variables
qe = [re me Ee];

% Compute left and right differences, evaluate slopes
dup = qe(3:N+4,:)-qe(2:N+3,:); dum = qe(2:N+3,:)-qe(1:N+2,:);
duL(:,1) = SlopeLimit(dup(:,1), dum(:,1), type, c, M, h);
duL(:,2) = SlopeLimit(dup(:,2), dum(:,2), type, c, M, h);
duL(:,3) = SlopeLimit(dup(:,3), dum(:,3), type, c, M, h);

% Compute intermediate values.  $f'(u) = Au$ 
for i=1:N+2
    A = EulerJac(qe(i+1,:),gamma);
    qh(i,:) = qe(i+1,:)-k/(2*h)*duL(i,:)*A';
end

% Advance solution k/2
qs = (q(1:N-1,:)+q(2:N,:))/2 + (duL(2:N,:)-duL(3:N+1,:))/8 ...
    - kl/h*(EulerFlux(qh(3:N+1,:),gamma)-EulerFlux(qh(2:N,:),gamma));

% Second step from staggered to non-staggered grid
% Extend data and assign boundary conditions
xs = x(1:Ns)+h/2;
[xe,re] = extendstag(xs,qs(:,1),h,2,'D',1.0,'D',0.125);
[xe,me] = extendstag(xs,qs(:,2),h,2,'D',0,'N',0);
[xe,Ee] = extendstag(xs,qs(:,3),h,2,'D',2.5,'N',0);

% [xe,re] = extendstag(xs,qs(:,1),h,2,'D',3.857143,'N',0);
% [xe,me] = extendstag(xs,qs(:,2),h,2,'D',10.141852,'D',0);
% [xe,Ee] = extendstag(xs,qs(:,3),h,2,'D',39.166661,'N',0);

% Extract variables

```

```

qe = [ re me Ee ];

% Compute left and right differences, evaluate slopes
dup = qe(3:Ns+4,:) - qe(2:Ns+3,:); dum = qe(2:Ns+3,:) - qe(1:Ns+2,:);
duLs(:,1) = SlopeLimit(dup(:,1), dum(:,1), type, c, M, h);
duLs(:,2) = SlopeLimit(dup(:,2), dum(:,2), type, c, M, h);
duLs(:,3) = SlopeLimit(dup(:,3), dum(:,3), type, c, M, h);

% Compute intermediate values.  $f'(u) = Au$ 
for i=1:Ns+2
    A = EulerJac(qe(i+1,:), gamma);
    qh(:, :) = qe(i+1,:) - k1/(2*h)*duLs(i,:)*A';
end

% Advance solution k/2
qns = (qe(2:Ns+2,:)+qe(3:Ns+3,:))/2 + (duLs(1:Ns+1,:)-duLs(2:Ns+2,:))/8 ...
    - k1/h*(EulerFlux(qh(2:Ns+2,:), gamma)-EulerFlux(qh(1:Ns+1,:), gamma));

% Restore residual
dq = (qns-q)/k;
return

```

We note that slope limiting is performed on the conserved variables and not, as we have explored previously, on the characteristics variables. Although one could perform the slope limiting on the characteristic variables, the considerable cost of this approach is not warranted for the examples considered here.

The details of the Euler equations are implemented in EulerFlux.m and EulerJac.m, reflecting the nonlinear flux and the flux Jacobian  $\nabla_u f$ , respectively.

---

**Script 10.21. EulerFlux.m: Evaluation of the flux of the Euler equation.**

---

```

function [fu] = EulerFlux(q, gamma);
% function [fu] = EulerFlux(q, gamma);
% Purpose: Compute flux for 1D Euler equations
N = length(q(:,1)); fu = zeros(N,3);

r = q(:,1); ru = q(:,2); E = q(:,3);
fu(:,1) = ru; p = (gamma-1)*(E - 0.5*ru.^2./r);
fu(:,2) = ru.^2./r + p; fu(:,3) = (E+p).*ru./r;
return

```

---

**Script 10.22. EulerJac.m: Evaluation of the flux Jacobian for the Euler equation.**

---

```

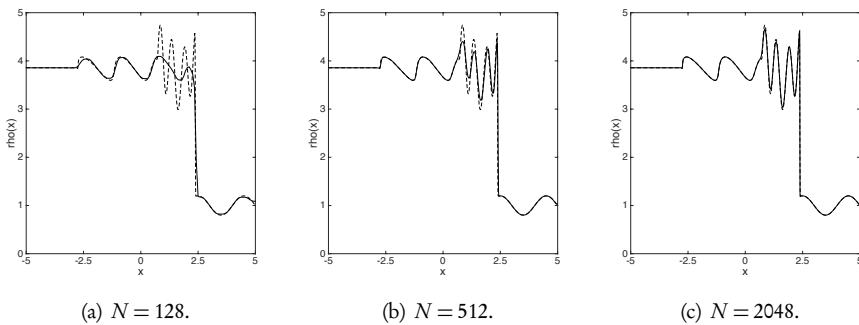
function [A] = EulerJac(q, gamma);
% function [A] = EulerJac(q, gamma);
% Purpose: Compute flux Jacobian for 1D Euler equations
A = zeros(3,3); r = q(1); ru = q(2); E = q(3); u = ru/r;
A(2,1) = -(3-gamma)/2*u^2; A(2,2) = (3-gamma)*u; A(2,3) = gamma-1;
A(1,2)=1; A(3,1) = -gamma*E*u/r + (gamma-1)*u^3;
A(3,2) = gamma*E/r - 3*(gamma-1)/2*u^2; A(3,3) = gamma*u;
return

```

---

We remark that these two routines are the only ones that need to be changed to solve a different system of conservation laws.

To evaluate the performance of the scheme, we consider the shock-entropy wave example. The results obtained for different resolutions are shown in Fig. 10.16. We



**Figure 10.16.** Computed density (solid line) of the Euler equations using a second order central scheme with a MUSCL limiter. The dashed line indicates the reference solution at  $T = 1.8$ .

note the substantial improvements over the results obtained with the monotone scheme in Fig. 10.6 or with most slope limited schemes, see Fig. 10.12.

## 10.4 • Extension to multidimensional problems

The extension of flux limited and slope limited schemes to the two-dimensional case is relatively straightforward and follows the dimension-by-dimension approach already discussed for monotone schemes in Chapter 7.

The extension of staggered central schemes is, however, slightly more involved due to the more complex grid structure. For a detailed discussion of the formulation of multidimensional staggered schemes, we refer to [19]. In the following, we use the second order central scheme

$$\begin{aligned} \bar{u}_{i+1/2,j+1/2}^{n+1} = & \left\langle \frac{1}{4} (\bar{u}_{i+1,:}^n + \bar{u}_{i,:}^n) + \frac{1}{8} (\overline{\delta_x \bar{u}_{i,:}^n} - \overline{\delta_x \bar{u}_{i+1,:}^n}) - \frac{k}{b_x} (f(\bar{u}_{i+1,:}^n) - f(\bar{u}_{i,:}^n)) \right\rangle_{j+1/2} \\ & + \left\langle \frac{1}{4} (\bar{u}_{:,j+1}^n + \bar{u}_{:,j}^n) + \frac{1}{8} (\overline{\delta_y \bar{u}_{:,j}^n} - \overline{\delta_y \bar{u}_{:,j+1}^n}) \right. \\ & \left. - \frac{k}{b_y} (f(\bar{u}_{:,j+1}^n) - f(\bar{u}_{:,j}^n)) \right\rangle_{i+1/2}, \end{aligned}$$

where

$$\langle v_{i,:} \rangle_{j+1/2} = \frac{1}{2} (v_{i,j+1} + v_{i,j}), \quad \langle v_{:,j} \rangle_{i+1/2} = \frac{1}{2} (v_{i+1,j} + v_{i,j}).$$

Furthermore, we define the slopes,  $\overline{\delta_x v}$  and  $\overline{\delta_y v}$ , as the limited slopes in the  $x$ - and  $y$ -direction, respectively. Finally, we evaluate the intermediate solution as

$$u_{:,:}^{n+1/2} = \bar{u}_{:,:}^n - \frac{k}{2b_x} \nabla_u f_x \overline{\delta_x \bar{u}_{:,:}^n} - \frac{k}{2b_y} \nabla_u f_y \overline{\delta_y \bar{u}_{:,:}^n},$$

where the general flux is  $f = [f_x, f_y]$ .

### 10.4.1 ▀ Burgers' equation

Let us first consider the two-dimensional Burgers' equation with periodic boundary conditions as defined in subsection 7.1.1. The problem is relatively simple and its solution evolves from a smooth initial condition to a propagating shock which eventually dissipates.

The implementation of limited schemes is very similar to that of the monotone schemes. We shall not show routines for initializing the solvers and time-stepping, as they are similar to those of the monotone scheme in subsection 7.1.1. On the other hand, the evaluation of the right-hand side is different. This is shown in BurgersFLrhs2D.m for the flux limited scheme, based on a combination of a Lax–Friedrichs and a Lax–Wendroff scheme.

**Script 10.23.** *BurgersFLrhs2D.m: Evaluation of the right-hand side for the two-dimensional Burgers' equation using a flux limited scheme.*

---

```

function [du] = BurgersFLrhs2D (x,y,u,hx,hy,k,maxvel)
% function [du] = BurgersFLrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D Burgers equation
% using a flux limited scheme
Nx = size(x); Nx = Nx(2); Ny = Nx(1); du = zeros(Ny,Nx);

% Choose flux limiter - 0:LF; 1:CO; 2:Koren; 3:Sweby; 4:OSPRE; 5:van Leer
type = 5; beta = 1.5;

% Extend data and assign boundary conditions in x-direction
for i=1:Ny
    [xe,ue] = extend(x(i,:),u(i,:),hx,1,'P',0,'P',0);

    % Compute indicator function and define flux limiter
    r = (ue(2:Nx+1) - ue(1:Nx))./(ue(3:Nx+2)-ue(2:Nx+1));
    [xe,re] = extend(x(i,:),r,hx,1,'N',0,'N',0); rm = 1./re;
    phiLp = FluxLimit(re(1:Nx),type,beta);
    phiRp = FluxLimit(re(2:Nx+1),type,beta);
    phiLm = FluxLimit(rm(2:Nx+1),type,beta);
    phiRm = FluxLimit(rm(3:Nx+2),type,beta);

    ufilt = (u(i,:)'>=0);
    phiL = ufilt.*phiLp + (1-ufilt).*phiLm;
    phiR = ufilt.*phiRp + (1-ufilt).*phiRm;

    % Compute left flux - Change numerical flux here
    Fluxlow = BurgersLF(ue(1:Nx),ue(2:Nx+1),0,maxvel);
    Fluxhigh = BurgersLW(ue(1:Nx),ue(2:Nx+1),k/hx,maxvel);
    FluxL = Fluxlow - phiL.* (Fluxlow - Fluxhigh);

    % Compute right flux - Change numerical flux here
    Fluxlow = BurgersLF(ue(2:Nx+1),ue(3:Nx+2),0,maxvel);
    Fluxhigh = BurgersLW(ue(2:Nx+1),ue(3:Nx+2),k/hx,maxvel);
    FluxR = Fluxlow - phiR.* (Fluxlow - Fluxhigh);

    % Update residual
    du(i,:) = -(FluxR' - FluxL')/hx;
end

% Extend data and assign boundary conditions in y-direction
for j=1:Nx
    [xe,ue] = extend(y(:,j),u(:,j),hy,1,'P',0,'P',0);

```

---

```

% Compute indicator function and define flux limiter
r = (ue(2:Ny+1) - ue(1:Ny))./(ue(3:Ny+2)-ue(2:Ny+1));
[xe,re] = extend(y(:,j),r,hy,1,'N',0,'N',0); rm = 1./re;
phiLp = FluxLimit(re(1:Ny),type, beta);
phiRp = FluxLimit(re(2:Ny+1),type, beta);
phiLm = FluxLimit(rm(2:Ny+1),type, beta);
phiRm = FluxLimit(rm(3:Ny+2),type, beta);

ufilt = (u(:,j)>=0);
phiL = ufilt.*phiLp + (1-ufilt).*phiLm;
phiR = ufilt.*phiRp + (1-ufilt).*phiRm;

% Compute left flux - Change numerical flux here
Fluxlow = BurgersLF(ue(1:Ny),ue(2:Ny+1),0,maxvel);
Fluxhigh = BurgersLW(ue(1:Ny),ue(2:Ny+1),k/hy,maxvel);
FluxL = Fluxlow - phiL.* (Fluxlow - Fluxhigh);

% Compute right flux - Change numerical flux here
Fluxlow = BurgersLF(ue(2:Ny+1),ue(3:Ny+2),0,maxvel);
Fluxhigh = BurgersLW(ue(2:Ny+1),ue(3:Ny+2),k/hy,maxvel);
FluxR = Fluxlow - phiR.* (Fluxlow - Fluxhigh);

% Update residual
du(:,j) = du(:,j) - (FluxR - FluxL)/hy;
end
return

```

In *BurgersSLrhs2D.m* we illustrate the implementation of the slope limited scheme, based on the MUSCL approach.

**Script 10.24. *BurgersSLrhs2D.m: Evaluation of the right-hand side for the two-dimensional Burgers' equation using a slope limited scheme.***

```

function [du] = BurgersSLrhs2D(x,y,u,hx,hy,k,maxvel)
% function [du] = BurgersSLrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D Burgers equation
% using slope limited method
Nx = size(x); Nx = Nx(2); Ny = Nx(1); du = zeros(Ny,Nx);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 5; c=0; M=10;

% Extend data and assign boundary conditions in x-direction
for i=1:Ny
    [xe,ue] = extend(x(i,:),u(i,:),hx,2,'P',0,'P',0);

    % Compute element slopes and slope limit
    dup = ue(3:Nx+4)-ue(2:Nx+3); dum = ue(2:Nx+3)-ue(1:Nx+2);
    duL = SlopeLimit(dup,dum,type,c,M,hx);

    % Compute cell interface values
    uloc = ue(2:Nx+3);
    uh = uloc - BurgersJac2Dx(uloc).*k/(2*hx).*duL;
    uL = uh - duL/2; uR = uh + duL/2;

    % Compute rhs
    du(i,:) = -(BurgersLF(uR(2:Nx+1),uL(3:Nx+2),0,maxvel) ...
                - BurgersLF(uR(1:Nx),uL(2:Nx+1),0,maxvel))/hx;
end

```

---

```

% Extend data and assign boundary conditions in y-direction
for j=1:Nx
    [xe,ue] = extend(y(:,j),u(:,j),hy,2,'P',0,'P',0);

    % Compute element slopes and slope limit
    dup = ue(3:Ny+4)-ue(2:Ny+3); dum = ue(2:Ny+3)-ue(1:Ny+2);
    duL = SlopeLimit(dup,dum,type,c,M,hy);

    % Compute cell interface values - for  $f'(u) = 2*u$ ;
    uloc = ue(2:Ny+3);
    uh = uloc - BurgersJac2Dy(uloc).*k/(2*hy).*duL;
    uL = uh - duL/2; uR = uh + duL/2;

    % Compute rhs
    du(:,j) = du(:,j) - (BurgersLF(uR(2:Ny+1),uL(3:Ny+2),0,maxvel) ...
        - BurgersLF(uR(1:Ny),uL(2:Ny+1),0,maxvel))/hy;
end
return

```

---

We finally also consider the evaluation of the right-hand side using the second order central staggered scheme, illustrated in *BurgersCrhs2D.m*.

**Script 10.25. *BurgersCrhs2D.m*: Evaluation of the right-hand side for the two-dimensional Burgers' equation using a second order accurate central scheme.**

---

```

function [du] = BurgersCrhs2D(x,y,u,hx,hy,k,maxvel)
% function [du] = BurgersCrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D Burgers equation
% using second order central scheme
Nx=size(x); Nx=Nxy(2); Ny=Nxy(1); Nxs=Nx-1; Nys=Ny-1; k1 = k/2;
du=zeros(Ny,Nx); duLx = zeros(Ny+2,Nx+2); duLy = zeros(Ny+2,Nx+2);
uG = zeros(Ny+4,Nx+4); uGs = zeros(Nys+4,Nxs+4);
duLxs = zeros(Nys+2,Nxs+2); duLys = zeros(Nys+2,Nxs+2);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% First step from non-staggered to staggered grid
% Extend data and assign boundary conditions on global data
for i=1:Ny
    [xe,ue] = extend(x(1,:),u(i,:),hx,2,'P',0,'P',0);
    uG(i+2,:)=ue';
end
for j=1:Nx+4
    [xe,ue] = extend(y(:,1),uG(3:Ny+2,j),hy,2,'P',0,'P',0);
    uG(:,j)=ue;
end

% Compute slopes
for i=1:Ny+2;
    dup = uG(i+1,3:Nx+4)-uG(i+1,2:Nx+3);
    dum = uG(i+1,2:Nx+3)-uG(i+1,1:Nx+2);
    duLx(i,:)=SlopeLimit(dup',dum',type,c,M,hx)';
end
for j=1:Nx+2
    dup = uG(3:Ny+4,j+1)-uG(2:Ny+3,j+1);
    dum = uG(2:Ny+3,j+1)-uG(1:Ny+2,j+1);

```

```

duLy(:, j) = SlopeLimit(dup, dum, type, c, M, hy);
end

% Compute intermediate solution
uloc = uG(2:Ny+3,2:Nx+3);
uhG = uloc - kl/2.* (BurgersJac2Dx(uloc).*duLx/hx ...
+ BurgersJac2Dy(uloc).*duLy/hy);

% Compute flux in x-direction
for i=1:Ny-1
    % Left term
    ue = uG(i+2,:); duL = duLx(i+1,:); uh = uhG(i+1,:);
    Flux1 = (ue(3:Nx+1)+ue(4:Nx+2))/4 + (duL(2:Nx)-duL(3:Nx+1))/8 ...
    - kl/hx*(BurgersFlux2Dx(uh(3:Nx+1))-BurgersFlux2Dx(uh(2:Nx)));
    % Right term
    ue = uG(i+3,:); duL = duLx(i+2,:); uh = uhG(i+2,:);
    Flux2 = (ue(3:Nx+1)+ue(4:Nx+2))/4 + (duL(2:Nx)-duL(3:Nx+1))/8 ...
    - kl/hx*(BurgersFlux2Dx(uh(3:Nx+1))-BurgersFlux2Dx(uh(2:Nx)));
    % Compute rhs
    us(i,:)= (Flux1+Flux2)'/2;
end

% Compute flux in y-direction
for j=1:Nx-1
    % Down term
    ue = uG(:,j+2); duL = duLy(:,j+1); uh = uhG(:,j+1);
    Flux1 = (ue(3:Ny+1)+ue(4:Ny+2))/4 + (duL(2:Ny)-duL(3:Ny+1))/8 ...
    - kl/hy*(BurgersFlux2Dy(uh(3:Ny+1))-BurgersFlux2Dy(uh(2:Ny)));
    % Up term
    ue = uG(:,j+3); duL = duLy(:,j+2); uh = uhG(:,j+2);
    Flux2 = (ue(3:Ny+1)+ue(4:Ny+2))/4 + (duL(2:Ny)-duL(3:Ny+1))/8 ...
    - kl/hy*(BurgersFlux2Dy(uh(3:Ny+1))-BurgersFlux2Dy(uh(2:Ny)));
    % Compute rhs
    us(:,j)= us(:,j)+(Flux1+Flux2)/2;
end

% Second step from staggered to non-staggered grid
% Extend data and assign boundary conditions on global data
for i=1:Nys
    [xe,ue]=extendstag(x(1,1:Nxs),us(i,:),hx,2,'P',0,'P',0);
    uGs(i+2,:)=ue';
end
for j=1:Nxs+4
    [xe,ue]=extendstag(y(1:Nys,1),uGs(3:Nys+2,j),hy,2,'P',0,'P',0);
    uGs(:,j)=ue';
end

% Compute slopes
for i=1:Nys+2;
    dup = uGs(i+1,3:Nxs+4)-uGs(i+1,2:Nxs+3);
    dum = uGs(i+1,2:Nxs+3) - uGs(i+1,1:Nxs+2);
    duLxs(i,:)= SlopeLimit(dup',dum',type,c,M,hx)';
end
for j=1:Nxs+2
    dup = uGs(3:Nys+4,j+1)-uGs(2:Nys+3,j+1);
    dum = uGs(2:Nys+3,j+1) - uGs(1:Nys+2,j+1);
    duLys(:,j)= SlopeLimit(dup,dum,type,c,M,hy);
end

% Compute intermediate solution -
uloc = uGs(2:Nys+3,2:Nxs+3);
uhGs = uloc - kl/2.* (BurgersJac2Dx(uloc).*duLxs/hx ...
+ BurgersJac2Dy(uloc).*duLys/hy);

```

```

% Compute flux in x-direction
for i=1:Nxs+1
    % Left term
    ue = uGs(i+1,:); duL = duLxs(i,:); uh = uhGs(i,:);
    Flux1 = (ue(2:Nxs+2)+ue(3:Nxs+3))/4 + (duL(1:Nxs+1)-duL(2:Nxs+2))/8 ...
        - k1/hx*(BurgersFlux2Dx(uh(2:Nxs+2))-BurgersFlux2Dx(uh(1:Nxs+1)));
    % Right term
    ue = uGs(i+2,:); duL = duLxs(i+1,:); uh = uhGs(i+1,:);
    Flux2 = (ue(2:Nxs+2)+ue(3:Nxs+3))/4 + (duL(1:Nxs+1)-duL(2:Nxs+2))/8 ...
        - k1/hx*(BurgersFlux2Dx(uh(2:Nxs+2))-BurgersFlux2Dx(uh(1:Nxs+1)));
    % Compute rhs
    uns(i,:) = (Flux1+Flux2)/2;
end

% Compute flux in y-direction
for j=1:Nys+1
    % Down term
    ue = uGs(:,j+1); duL = duLys(:,j); uh = uhGs(:,j);
    Flux1 = (ue(2:Nys+2)+ue(3:Nys+3))/4 + (duL(1:Nys+1)-duL(2:Nys+2))/8 ...
        - k1/hy*(BurgersFlux2Dy(uh(2:Nys+2))-BurgersFlux2Dy(uh(1:Nys+1)));
    % Up term
    ue = uGs(:,j+2); duL = duLys(:,j+1); uh = uhGs(:,j+1);
    Flux2 = (ue(2:Nys+2)+ue(3:Nys+3))/4 + (duL(1:Nys+1)-duL(2:Nys+2))/8 ...
        - k1/hy*(BurgersFlux2Dy(uh(2:Nys+2))-BurgersFlux2Dy(uh(1:Nys+1)));
    % Compute rhs
    uns(:,j) = uns(:,j)+(Flux1+Flux2)/2;
end

% Restore residual
du = (uns - u)/k;
return

```

This relies on routines for evaluating the flux in the  $x$ -direction, `BurgersFlux2Dx.m`, and the  $y$ -direction, `BurgersFlux2Dy.m`, respectively.

---

**Script 10.26.** *BurgersFlux2Dx.m: Evaluation of the  $x$ -flux for the two-dimensional Burgers' equation.*

---

```

function [fu] = BurgersFlux2Dx(u)
% function [fu] = BurgersFlux2Dx(u)
% Purpose: Evaluate x-component of flux for 2D Burgers
fu = u.^2;
return

```

---

**Script 10.27.** *BurgersFlux2Dy.m: Evaluation of the  $y$ -flux for the two-dimensional Burgers' equation.*

---

```

function [fu] = BurgersFlux2Dy(u)
% function [fu] = BurgersFlux2Dy(u)
% Purpose: Evaluate y-component of flux for 2D Burgers
fu = u.^2;
return

```

---

It finally relies on routines for evaluating the flux Jacobian for the Burgers' flux along the  $x$ -direction, `BurgersJac2Dx.m`, and the  $y$ -direction, `BurgersJac2Dy.m`.

**Script 10.28. BurgersJac2Dx.m:** Evaluation of the  $x$ -flux Jacobian for the two-dimensional Burgers' equation.

---

```
function [ fu ] = BurgersJac2Dx (u)
% function [fu] = BurgersJac2Dx (u)
% Purpose: Evaluate x-component of flux for Jacobian 2D Burgers
fu = 2*u;
return
```

---

**Script 10.29. BurgersJac2Dy.m:** Evaluation of the  $y$ -flux Jacobian for the two-dimensional Burgers' equation.

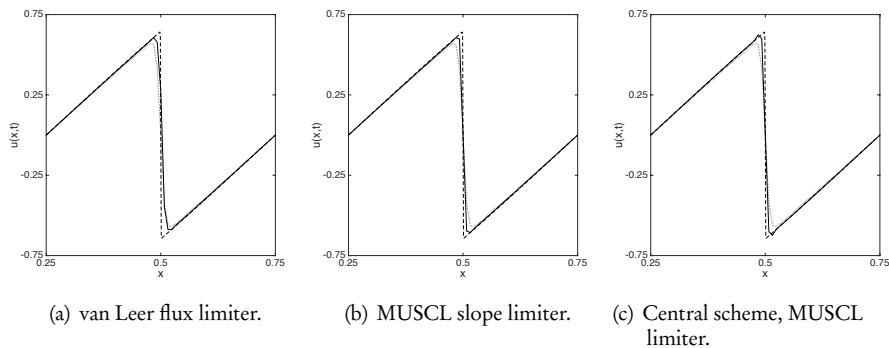
---

```
function [ fu ] = BurgersJac2Dy (u)
% function [fu] = BurgersJac2Dy (u)
% Purpose: Evaluate y-component of flux for Jacobian 2D Burgers
fu = 2*u;
return
```

---

If a different scalar equation is to be solved, only these four routines need to be changed.

The performance of the different approaches is illustrated in Fig. 10.17. Most importantly, we observe that the high-order schemes provide improved accuracy when compared to the monotone scheme. The results, obtained with  $N_x = N_y = 128$ , also suggest that the accuracy of the slope limited scheme is superior to that of the flux limited scheme and comparable to the results obtained with the central scheme.



**Figure 10.17.** Cross section of solution (solid line) of the two-dimensional Burgers' equation, computed with  $N_x = N_y = 128$  at  $T = 0.1$  and different second order accurate schemes. The results with other choices of limiters are similar to the ones shown. The dashed line shows the exact solution, while the dotted line reflects the results obtained with a monotone scheme.

#### 10.4.2 • Nonconvex scalar equation

Let us now consider the nonconvex problem, introduced in subsection 1.4.2. We solve this problem using the three methods already considered for Burgers' equation. Because of the close similarity to the implementation of Burgers' equation, we do not show the scripts here. Indeed, the main difference is the routines for evaluating the flux in the  $x$ -direction, KPPFlux2Dx.m, and in the  $y$ -direction, KPPFlux2Dy.m, and the flux

Jacobian for the KPP flux along the  $x$ -direction, `KPPJac2Dx.m`, and the  $y$ -direction, `KPPJac2Dy.m`, respectively.

**Script 10.30.** `KPPFlux2Dx.m`: *Evaluation of the  $x$ -flux for the two-dimensional KPP equation.*

---

```
function [fu] = KPPFlux2Dx(u)
% function [fu] = KPPFlux2Dx(u)
% Purpose: Evaluate x-component of flux for 2D KPP equation
fu = sin(u);
return
```

---

**Script 10.31.** `KPPFlux2Dy.m`: *Evaluation of the  $y$ -flux for the two-dimensional KPP equation.*

---

```
function [fu] = KPPFlux2Dy(u)
% function [fu] = KPPFlux2Dy(u)
% Purpose: Evaluate y-component of flux for 2D KPP equation
fu = cos(u);
return
```

---

**Script 10.32.** `KPPJac2Dx.m`: *Evaluation of the  $x$ -flux Jacobian for the two-dimensional KPP equation.*

---

```
function [fu] = KPPJac2Dx(u)
% function [fu] = KPPJac2Dx(u)
% Purpose: Evaluate x-component of flux for Jacobian 2D KPP equation
fu = cos(u);
return
```

---

**Script 10.33.** `KPPJac2Dy.m`: *Evaluation of the  $y$ -flux Jacobian for the two-dimensional KPP equation.*

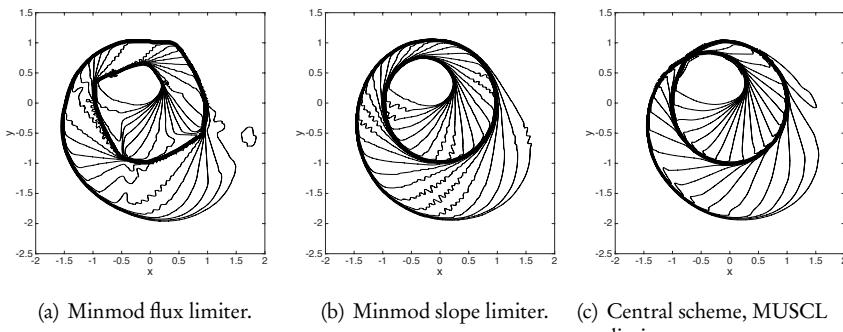
---

```
function [fu] = KPPJac2Dy(u)
% function [fu] = KPPJac2Dy(u)
% Purpose: Evaluate y-component of flux for Jacobian 2D KPP equation
fu = -sin(u);
return
```

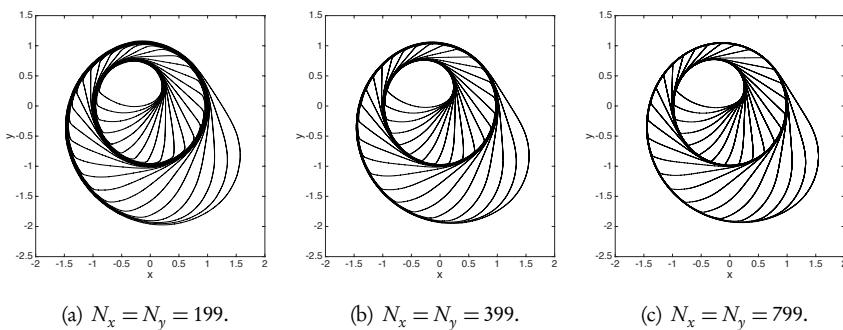
---

Figure 10.18 illustrates the results of solving the KPP problem with  $N_x = N_y = 128$  and the three schemes. Both the flux limited scheme and the slope limited scheme rely on the maximally dissipative minmod limiter, while the central scheme uses the less dissipative MUSCL limiter. When compared to the reference solution in Fig. 1.9, it is clear that all three solutions exhibit problems of different character. The lack of convergence is a result of insufficient dissipation in combination with the complex wave structure associated with the nonconvex flux. This has also been observed in past work [22].

For the flux limited and the slope limited case in Fig. 10.18, the results are obtained with the maximally dissipative minmod limiter. Increasing the dissipation of the limiter even more will eliminate the potential for second order accuracy. Another option



**Figure 10.18.** Contours of the computed solution of the KPP problem, obtained with  $N_x = N_y = 128$  at  $T = 1.0$ , and different second order schemes.  $CFL = 0.9$  is used in all computations. A high resolution reference solution is shown in Fig. 1.9.



**Figure 10.19.** Contours of the computer solution of the KPP problem at  $T = 1.0$  and a second order accurate central scheme with a minmod limiter.  $CFL = 0.9$  is used in all computations. A high-resolution reference solution is given in Fig. 1.9.

is to decrease the time step substantially to increase the overall dissipation, albeit at considerable computational cost.

The results in Fig. 10.18 for the central scheme are, however, obtained with the less dissipative MUSCL limiter. If we use the minmod limiter in the central scheme, we recover the results in Fig. 10.19 for increasing resolutions. A comparison with the high-resolution reference solution in Fig. 1.9, reveals excellent agreement.

### 10.4.3 • Euler equations

Finally, let us consider the solution of the two-dimensional Euler equations

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f} = 0,$$

where the conserved variables,  $q$ , and the flux,  $f$ , are given as

$$q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, f_x = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (E + p)u \end{bmatrix}, f_y = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (E + p)v \end{bmatrix},$$

with  $f = [f_x, f_y]$ . Further details can be found in subsection 1.4.2.

We consider the solution of the Euler equations using a slope limited scheme and a central second order scheme. The slope limited scheme comprises EulerSLDriver2D.m, essentially identical to the driver for the monotone scheme and not shown here, and EulerSL2D.m to enable the temporal integration using a third order strong stability preserving method. The slope limited scheme itself is expressed in EulerSLrhs2D.m.

---

**Script 10.34. EulerSL2D.m: Temporal integration of slope limited scheme for the two-dimensional Euler equations.**

---

```
function [q] = EulerSL2D(x,y,q,hx,hy,gamma,CFL,FinalTime)
% function [q] = EulerSL2D(x,y,q,hx,hy,gamma,CFL,FinalTime)
% Purpose : Integrate 2D Euler equation until FinalTime
% using a slope limited scheme.
time = 0; tstep = 0;

while (time<FinalTime)
    % Set timestep
    r = q(:,:,1); ru = q(:,:,2); rv = q(:,:,3); E = q(:,:,4);
    p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r); c = sqrt(gamma*p./r);
    maxvelu = max(max(c+abs(ru./r))); maxvelv = max(max(c+abs(rv./r)));
    k = CFL*min(hx,hy)/sqrt(2)/sqrt(maxvelu^2+maxvelv^2);
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution using 3rd SSP-RK
    [dq] = EulerSLrhs2D(x,y,q,gamma,hx,hy,k); q1 = q + k*dq;
    [dq] = EulerSLrhs2D(x,y,q1,gamma,hx,hy,k); q2 = (3*q+q1+k*dq)/4;
    [dq] = EulerSLrhs2D(x,y,q2,gamma,hx,hy,k); q = (q+2*q2+2*k*dq)/3;
    time = time+k; tstep = tstep+1;
end
return
```

---

**Script 10.35. EulerSLrhs2D.m: Slope limited scheme for the two-dimensional Euler equations.**

---

```
function [dq] = EulerSLrhs2D(x,y,q,gamma,hx,hy,k);
% function [dq] = EulerSLrhs2D(x,y,gamma,hx,hy,k);
% Purpose: Evaluate right hand side for the two-dimensional Euler equations
% using a slope limited scheme
Nxy = size(x); Nx = Nxy(2); Ny = Nxy(1);
qex = zeros(Nx+4,4); qey = zeros(Ny+4,4);
dqLx = zeros(Nx+2,4); dqLy = zeros(Ny+2,4);

% Choose slope limiter - 0:LF; 1:minmod; 2:MUSCL; 3:Superbee;
% 4:van Albada; 5:van Leer, 6: TVB
type = 2; c=0; M=10;

% Apply slope limited scheme in the x-direction
for i=1:Ny
```

```

% Extend data and assign boundary conditions in x-direction
[ xe , qex (: , 1) ] = extend(x(i , : ) , q(i , : , 1) , hx , 2 , 'N' , 0.0 , 'N' , 0.0) ;
[ xe , qex (: , 2) ] = extend(x(i , : ) , q(i , : , 2) , hx , 2 , 'N' , 0.0 , 'N' , 0.0) ;
[ xe , qex (: , 3) ] = extend(x(i , : ) , q(i , : , 3) , hx , 2 , 'N' , 0.0 , 'N' , 0.0) ;
[ xe , qex (: , 4) ] = extend(x(i , : ) , q(i , : , 4) , hx , 2 , 'N' , 0.0 , 'N' , 0.0) ;

% Compute element slopes and slope limit
dup = qex (3:Nx+4,:) - qex (2:Nx+3,:);
dum = qex (2:Nx+3,:) - qex (1:Nx+2,:);
dqLx (: , 1) = SlopeLimit (dup (: , 1) , dum (: , 1) , type , c , M , hx) ;
dqLx (: , 2) = SlopeLimit (dup (: , 2) , dum (: , 2) , type , c , M , hx) ;
dqLx (: , 3) = SlopeLimit (dup (: , 3) , dum (: , 3) , type , c , M , hx) ;
dqLx (: , 4) = SlopeLimit (dup (: , 4) , dum (: , 4) , type , c , M , hx) ;
q1 = qex (2:Nx+3,:) - dqLx /2; qr = qex (2:Nx+3,:) + dqLx /2;

% Compute fluxes
[ dq1 ] = EulerLF2Dx (qr (2:Nx+1,:) , q1 (3:Nx+2,:) , gamma) ;
[ dq2 ] = EulerLF2Dx (qr (1:Nx , :) , q1 (2:Nx+1,:) , gamma) ;

% Update residual
dq (i , : , :) = -(dq1-dq2)/hx;
end

% Apply slope limited in the y-direction
for j=1:Nx
    % Extend data and assign boundary conditions in y-direction
    [ ye , qey (: , 1) ] = extend(y(:, j) , q(:, j , 1) , hy , 2 , 'N' , 0.0 , 'N' , 0.0) ;
    [ ye , qey (: , 2) ] = extend(y(:, j) , q(:, j , 2) , hy , 2 , 'N' , 0.0 , 'N' , 0.0) ;
    [ ye , qey (: , 3) ] = extend(y(:, j) , q(:, j , 3) , hy , 2 , 'N' , 0.0 , 'N' , 0.0) ;
    [ ye , qey (: , 4) ] = extend(y(:, j) , q(:, j , 4) , hy , 2 , 'N' , 0.0 , 'N' , 0.0) ;

    % Compute element slopes and slope limit
    dup = qey (3:Ny+4,:) - qey (2:Ny+3,:);
    dum = qey (2:Ny+3,:) - qey (1:Ny+2,:);
    dqLy (: , 1) = SlopeLimit (dup (: , 1) , dum (: , 1) , type , c , M , hy) ;
    dqLy (: , 2) = SlopeLimit (dup (: , 2) , dum (: , 2) , type , c , M , hy) ;
    dqLy (: , 3) = SlopeLimit (dup (: , 3) , dum (: , 3) , type , c , M , hy) ;
    dqLy (: , 4) = SlopeLimit (dup (: , 4) , dum (: , 4) , type , c , M , hy) ;
    q1 = qey (2:Ny+3,:) - dqLy /2; qr = qey (2:Ny+3,:) + dqLy /2;

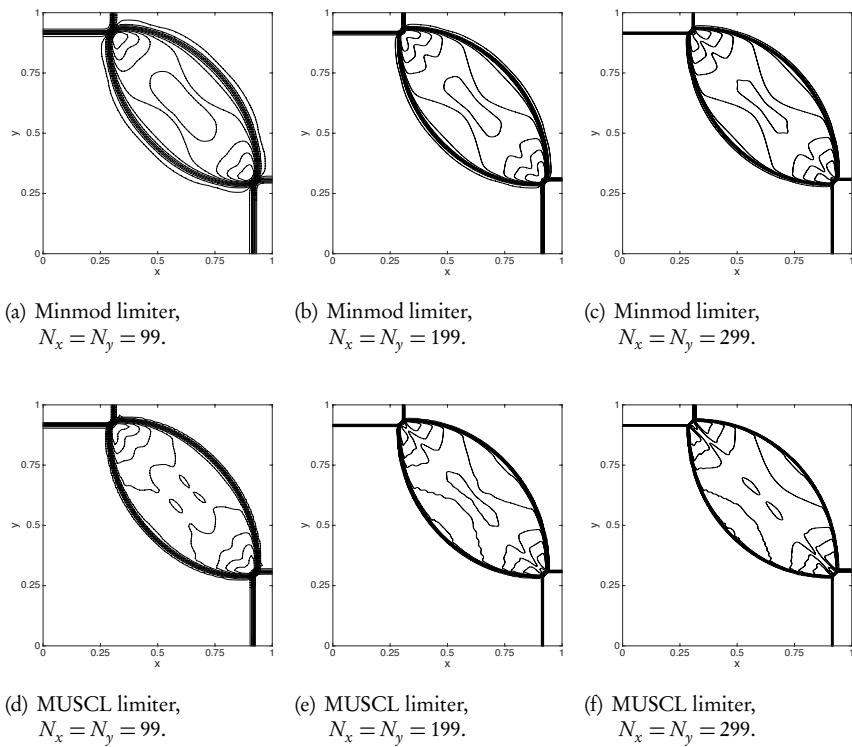
    % Compute fluxes
    [ dq1 ] = EulerLF2Dy (qr (2:Ny+1,:) , q1 (3:Ny+2,:) , gamma) ;
    [ dq2 ] = EulerLF2Dy (qr (1:Ny , :) , q1 (2:Ny+1,:) , gamma) ;

    % Update residual
    dq (: , j , :) = dq (: , j , :) - (reshape (dq1 , Ny , 1 , 4) ...
        - reshape (dq2 , Ny , 1 , 4))/hy;
end
return

```

In Fig. 10.20 we illustrate the results obtained when solving a two-dimensional Riemann problem, comprising four shock waves, for different resolution and for two different choices of the limiter. Comparing with the reference solution in subsection 1.4.2, we observe excellent agreement and convergence towards the solution as well as cleanly resolved shocks, in particular when the more aggressive MUSCL limiter is used.

Similar observations can be made for the more complex problem shown in Fig. 10.21, which features two shocks and two contact waves. In particular, we note the excellent resolution of the contact waves.



**Figure 10.20.** Contours of the computed density for a two-dimensional Riemann problem, comprising four propagating shocks, solved with a slope limited scheme. All results are shown at  $T = 0.25$ , obtained with a  $CFL = 0.5$ .

When turning to the evaluation of the second order central scheme, we recall the discussion of the staggered scheme for Burgers' equation. The central pieces of the scheme are the routines to evaluate the flux as well as flux Jacobians. For the case of the two-dimensional Euler equations the evaluation of the  $x$ - and  $y$ -flux are provided, respectively, by `EulerFlux2Dx.m` and `EulerFlux2Dy.m`.

**Script 10.36.** *EulerFlux2Dx.m: Evaluation of the  $x$ -flux for the two-dimensional Euler equations.*

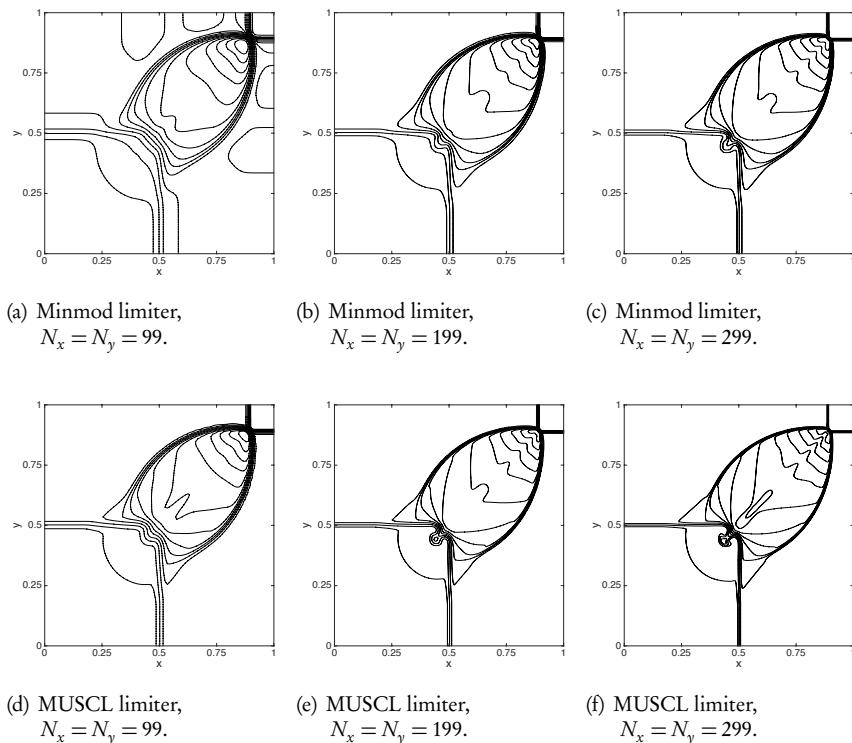
---

```

function [qx] = EulerFlux2Dx(q,gamma);
% function [qx] = EulerFlux2Dx(q,gamma);
% Purpose: Evaluate x-flux for 2D Euler equations
r = q(:,1); ru = q(:,2); rv = q(:,3); E = q(:,4);
p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./ r);
qx(:,1)= ru; qx(:,2) = ru.^2./ r + p;
qx(:,3) = ru.*rv./ r; qx(:,4) = (E+p).*ru./ r;
return

```

---



**Figure 10.21.** Contours of the computed density for a two-dimensional Riemann problem, comprising two propagating shocks and two contact waves, solved with a slope limited scheme. All results are shown at  $T = 0.25$ , obtained with a  $CFL = 0.5$ .

**Script 10.37.** *EulerFlux2Dy.m: Evaluation of the y-flux for the two-dimensional Euler equations.*

---

```

function [qy] = EulerFlux2Dy(q,gamma);
% function [qy] = EulerFlux2Dy(q,gamma);
% Purpose: Evaluate y-flux for 2D Euler equations
r = q(:,1); ru = q(:,2); rv = q(:,3); E = q(:,4);
p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r);

qy(:,1)= rv; qy(:,2) = ru.*rv./r;
qy(:,3) = rv.^2./r+p; qy(:,4) = (E+p).*rv./r;
return

```

---

The flux Jacobians are given as

$$\nabla_q f_x = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{2}\tilde{\gamma}(u^2+v^2)-u^2 & (3-\gamma)u & -\tilde{\gamma}v & \tilde{\gamma} \\ -uv & v & u & 0 \\ (\tilde{\gamma}(u^2+v^2)-\gamma E \rho^{-1})u & \gamma E \rho^{-1} - \frac{1}{2}\tilde{\gamma}(u^2+v^2)-\tilde{\gamma}u^2 & -\tilde{\gamma}uv & \gamma u \end{bmatrix},$$

$$\nabla_q f_y = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ \frac{1}{2}\tilde{\gamma}(u^2+v^2)-v^2 & -\tilde{\gamma}u & (3-\gamma)v & \tilde{\gamma} \\ (\tilde{\gamma}(u^2+v^2)-\gamma E\rho^{-1})v & -\tilde{\gamma}uv & \gamma E\rho^{-1}-\frac{1}{2}\tilde{\gamma}(u^2+v^2)-\tilde{\gamma}v^2 & \gamma v \end{bmatrix},$$

and expressed in `EulerJac2Dx.m` and `EulerJac2Dy.m`, respectively. Here  $\tilde{\gamma} = \gamma - 1$  for simplicity of the expression.

**Script 10.38. *EulerJac2Dx.m*: Evaluation of the  $x$ -flux Jacobian for the two-dimensional Euler equations.**

---

```
function [Ax] = EulerJac2Dx(q,gamma);
% function [Ax] = EulerJac2Dx(q,gamma);
% Purpose: Evaluate x-flux Jacobian for 2D Euler equations
Ax = zeros(4);
r = q(1); ru = q(2); rv = q(3); E = q(4); u = ru/r; v = rv/r;
p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r); pE = 0.5*(gamma-1)*(u.^2+v.^2);

Ax(1,2)=1; Ax(2,1) = pE - u.^2; Ax(2,2) = (3-gamma)*u; Ax(3,1) = -u*v;
Ax(2,3) = -(gamma-1)*v; Ax(2,4) = gamma-1; Ax(3,2) = v; Ax(3,3)=u;
Ax(4,1) = (2*pE - gamma*E/r)*u; Ax(4,2) = gamma*E/r - pE - (gamma-1)*u.^2;
Ax(4,3) = -(gamma-1)*u*v; Ax(4,4)=gamma*u;
return
```

---

**Script 10.39. *EulerJac2Dy.m*: Evaluation of the  $y$ -flux Jacobian for the two-dimensional Euler equations.**

---

```
function [Ay] = EulerJac2Dy(q,gamma);
% function [Ay] = EulerJac2Dy(q,gamma);
% Purpose: Evaluate y-flux Jacobian for 2D Euler equations
Ay = zeros(4);
r = q(1); ru = q(2); rv = q(3); E = q(4); u = ru/r; v = rv/r;
p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r); pE = 0.5*(gamma-1)*(u.^2+v.^2);

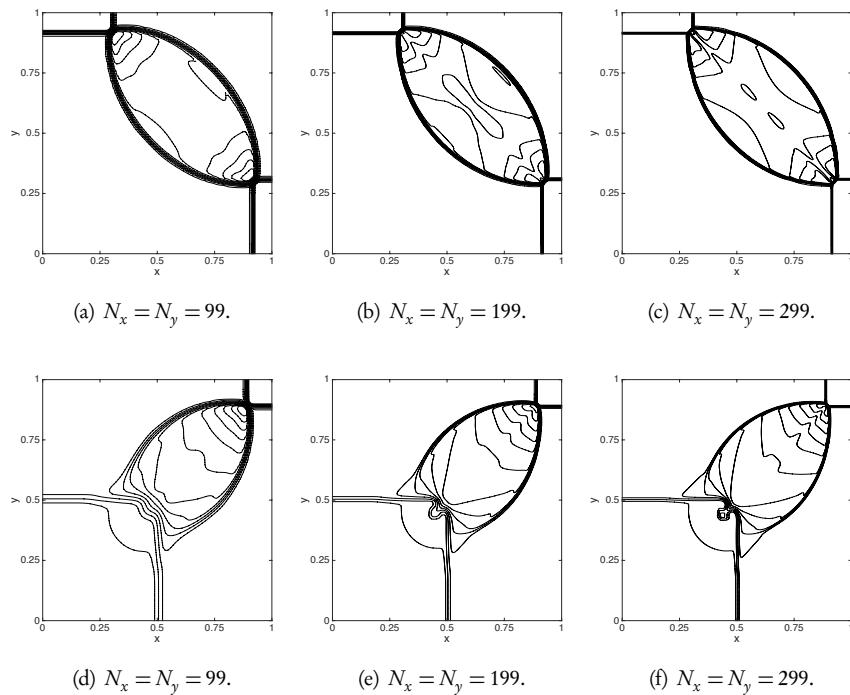
Ay(1,3)=1; Ay(2,1) = -u*v; Ay(2,2) = v; Ay(2,3)=u; Ay(3,1) = pE - v.^2;
Ay(3,2) = -(1-gamma)*u; Ay(3,3) = (3-gamma)*v; Ay(3,4) = gamma-1;
Ay(4,1) = (2*pE - gamma*E/r)*v; Ay(4,2) = -(gamma-1)*u*v;
Ay(4,3) = gamma*E/r - pE - (gamma-1)*v.^2; Ay(4,4)=gamma*v;
return
```

---

Due to the complexity of the implementation of the second order central scheme for the evaluation of the right-hand side of the Euler equations, we refrain from listing the script here. The structure is very similar to `BurgersCrhs2D.m` and nothing special is needed for the system case.

Figure 10.22 illustrates the performance of the second order central scheme for the two Riemann problems considered previously. We observe a very similar performance to that of the slope limited scheme.

Let us finally consider the convergence rate of the two schemes for the test of a smooth isotropic vortex, defined in subsection 1.4.2. The  $L^2$ -error of the density under refinement is shown in Table 10.1, and confirms the second order accuracy of both schemes for this smooth case.



**Figure 10.22.** Contour of the computer solution of two-dimensional Riemann problems, solved with a second order central scheme. Both cases are solved using a MUSCL limiter. All results are shown at  $T = 0.25$  and obtained with a CFL = 0.9.

**Table 10.1.**  $L^2$ -error for the density at  $T = 0.1$  for the isentropic vortex test of the two-dimensional Euler equations. The problem is solved using a slope limited and a central scheme, both with a MUSCL limiter. The error of the density is measured in  $\|\cdot\|_{h,2}$ .

$N_x = N_y$	16	32	64	128	256	512	Order
Slope limit	6.33e-2	1.66e-2	4.57e-3	1.41e-3	4.69e-4	1.38e-4	$\mathcal{O}(h^2)$
Central	2.05e-1	6.10e-2	1.41e-2	4.23e-3	1.38e-3	5.00e-4	$\mathcal{O}(h^2)$

## References

- [1] Richard M. Beam and Robert F. Warming. An implicit finite-difference algorithm for hyperbolic systems in conservation-law form. *Journal of Computational Physics*, 22(1):87–110, 1976.
- [2] Matania Ben-Artzi and Amnon Birman. Application of the “generalized Riemann problem” method to 1-D compressible flows with material interfaces. *Journal of Computational Physics*, 65(1):170–178, 1986.

- [3] Matania Ben-Artzi and Joseph Falcovitz. A second-order Godunov-type scheme for compressible fluid dynamics. *Journal of Computational Physics*, 55(1):1–32, 1984.
- [4] Marsha Berger, Michael J. Aftosmis, and Scott M. Murman. Analysis of slope limiters on irregular grids. In: *Proceedings of the 43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005, paper AIAA 2005-490.
- [5] David L. Book, Jay P. Boris, and K. Hain. Flux-corrected transport II: Generalizations of the method. *Journal of Computational Physics*, 18(3):248–283, 1975.
- [6] Jay P. Boris and David L. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11(1):38–69, 1973.
- [7] Jay P. Boris and D. L. Book. Flux-corrected transport. III. Minimal-error FCT algorithms. *Journal of Computational Physics*, 20(4):397–431, 1976.
- [8] Bernardo Cockburn, San-Yih Lin, and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, 1989.
- [9] Bernardo Cockburn and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework. *Mathematics of Computation*, 52(186):411–435, 1989.
- [10] Phillip Colella. A direct Eulerian MUSCL scheme for gas dynamics. *SIAM Journal on Scientific and Statistical Computing*, 6(1):104–117, 1985.
- [11] Phillip Colella and Paul R. Woodward. The piecewise parabolic method (PPM) for gas-dynamical simulations. *Journal of Computational Physics*, 54(1):174–201, 1984.
- [12] Nicolas Crouseilles, Michel Mehrenberger, and Eric Sonnendrücker. Conservative semi-Lagrangian schemes for Vlasov equations. *Journal of Computational Physics*, 229(6):1927–1953, 2010.
- [13] Kurt O. Friedrichs. Symmetric hyperbolic linear differential equations. *Communications on Pure and Applied Mathematics*, 7(2):345–392, 1954.
- [14] Jacob E. Fromm. A method for reducing dispersion in convective difference schemes. *Journal of Computational Physics*, 3(2):176–189, 1968.
- [15] John Giannakouros and George Em Karniadakis. Spectral element-FCT method for scalar hyperbolic conservation laws. *International Journal for Numerical Methods in Fluids*, 14(6):707–727, 1992.
- [16] John Giannakouros and George Em Karniadakis. A spectral element-FCT method for the compressible euler equations. *Journal of Computational Physics*, 115(1):65–85, 1994.
- [17] John G. Giannakouros, David Sidilkover, and George Em Karniadakis. Spectral element-FCT method for the one- and two-dimensional compressible Euler equations. *Computer Methods in Applied Mechanics and Engineering*, 116(1):113–121, 1994.

- [18] Guang-Shan Jiang, Doron Levy, Chi-Tien Lin, Stanley Osher, and Eitan Tadmor. High-resolution nonoscillatory central schemes with nonstaggered grids for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 35(6):2147–2168, 1998.
- [19] Guang-Shan Jiang and Eitan Tadmor. Nonoscillatory central schemes for multi-dimensional hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 19(6):1892–1917, 1998.
- [20] V. P. Kolgan. Application of the principle of minimum values of the derivative to the construction of finite-difference schemes for calculating discontinuous gasdynamics solutions. *TsAGI Uchenye Zapiski*, 3(6):68–77, 1972.
- [21] Barry Koren. *A Robust Upwind Discretization Method for Advection, Diffusion and Source Terms*. Centrum voor Wiskunde en Informatica Amsterdam, 1993.
- [22] Alexander Kurganov, Guergana Petrova, and Bojan Popov. Adaptive semidiscrete central-upwind schemes for nonconvex hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 29(6):2381–2401, 2007.
- [23] Alexander Kurganov and Eitan Tadmor. New high-resolution central schemes for nonlinear conservation laws and convection–diffusion equations. *Journal of Computational Physics*, 160(1):241–282, 2000.
- [24] Peter D. Lax and Xu-Dong Liu. Solution of two-dimensional Riemann problems of gas dynamics by positive schemes. *SIAM Journal on Scientific Computing*, 19(2):319–340, 1998.
- [25] Xu-Dong Liu and Peter D. Lax. Positive schemes for solving multi-dimensional hyperbolic systems of conservation laws. *CFD Journal*, 5(2):133–156, 1996.
- [26] Xu-Dong Liu and Peter D. Lax. Positive schemes for solving multi-dimensional hyperbolic systems of conservation laws II. *Journal of Computational Physics*, 187(2):428–440, 2003.
- [27] Xu-Dong Liu and Eitan Tadmor. Third order nonoscillatory central scheme for hyperbolic conservation laws. *Numerische Mathematik*, 79(3):397–425, 1998.
- [28] Beth E. McDonald. Flux-corrected pseudospectral method for scalar hyperbolic conservation laws. *Journal of Computational Physics*, 82(2):413–428, 1989.
- [29] Haim Nessyahu and Eitan Tadmor. Non-oscillatory central differencing for hyperbolic conservation laws. *Journal of Computational Physics*, 87(2):408–463, 1990.
- [30] Stanley Osher. Riemann solvers, the entropy condition, and difference. *SIAM Journal on Numerical Analysis*, 21(2):217–235, 1984.
- [31] Stanley Osher and Sukumar Chakravarthy. High resolution schemes and the entropy condition. *SIAM Journal on Numerical Analysis*, 21(5):955–984, 1984.
- [32] Stanley Osher and Sukumar Chakravarthy. Very high order accurate TVD schemes. In: *Oscillation Theory, Computation, and Methods of Compensated Compactness*, volume 2 of The IMA Volumes in Mathematics and Its Applications, pages 229–274, Springer, 1986.

- [33] Jing-Mei Qiu and Chi-Wang Shu. Conservative high order semi-Lagrangian finite difference WENO methods for advection in incompressible flow. *Journal of Computational Physics*, 230(4):863–889, 2011.
- [34] Philip L. Roe. Characteristic-based schemes for the Euler equations. *Annual Review of Fluid Mechanics*, 18(1):337–365, 1986.
- [35] Chi-Wang Shu. TVB uniformly high-order schemes for conservation laws. *Mathematics of Computation*, 49(179):105–121, 1987.
- [36] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.
- [37] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 83(1):32–78, 1989.
- [38] Andrew Staniforth and Jean Côté. Semi-Lagrangian integration schemes for atmospheric models: A review. *Monthly Weather Review*, 119(9):2206–2223, 1991.
- [39] P. K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 21(5):995–1011, 1984.
- [40] Eleuterio F Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer Science+Business Media, 2009.
- [41] Gábor Tóth and Dušan Odstrčil. Comparison of some flux corrected transport and total variation diminishing numerical schemes for hydrodynamic and magnetohydrodynamic problems. *Journal of Computational Physics*, 128(1):82–100, 1996.
- [42] G. Dick Van Albada, Bram Van Leer, and William W. Roberts, Jr. A comparative study of computational methods in cosmic gas dynamics. *Astronomy and Astrophysics*, 108:76–84, 1982.
- [43] Bram van Leer. Towards the ultimate conservative difference scheme I. The quest of monotonicity. In: *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*, pages 163–168. Springer, 1973.
- [44] Bram Van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *Journal of Computational Physics*, 14(4):361–370, 1974.
- [45] Bram Van Leer. Towards the ultimate conservative difference scheme III. Upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23(3):263–275, 1977.
- [46] Bram Van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *Journal of Computational Physics*, 23(3):276–299, 1977.
- [47] Bram Van Leer. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov’s method. *Journal of Computational Physics*, 32(1):101–136, 1979.

- [48] Bram Van Leer. On the relation between the upwind-differencing schemes of Godunov, Engquist–Osher and Roe. *SIAM Journal on Scientific and Statistical Computing*, 5(1):1–20, 1984.
- [49] Bram Van Leer. Upwind and high-resolution methods for compressible flow: From donor cell to residual-distribution schemes. *Communications in Computational Physics*, 1(2):192–206, 2006.
- [50] Robert F. Warming and Richard M. Beam. Upwind second-order difference schemes and applications in aerodynamic flows. *AIAA Journal*, 14(9):1241–1249, 1976.
- [51] Nicholas P. Waterson and Herman Deconinck. A unified approach to the design and application of bounded higher-order convection schemes. *Numerical Methods in Laminar and Turbulent Flow.*, 9:203–214, 1995.
- [52] Wikipedia contributors. *Flux Limiter*. Wikipedia, The Free Encyclopedia. [https://en.wikipedia.org/wiki/Flux\\_limiter](https://en.wikipedia.org/wiki/Flux_limiter) (accessed August 2016).
- [53] Paul Woodward and Phillip Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54(1):115–173, 1984.
- [54] Steven T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31(3):335–362, 1979.
- [55] Steven T. Zalesak. *The Design of Flux-Corrected Transport (FCT) Algorithms for Structured Grids*. Springer, 2005.

# Chapter 11

# Essentially nonoscillatory schemes

In the last chapter we took the initial steps toward the development and analysis of high-order accurate nonlinear schemes for conservation laws. Although the accuracy improvements, when compared to the results obtained with monotone schemes, were often substantial, the majority of these schemes were limited to second order accuracy. In Chapter 8 we investigated the considerable benefits associated with the use of schemes of even higher order accuracy. However, as became clear during the discussion, care must be exercised to avoid the creation of oscillations in the vicinity of steep gradients or shocks.

To construct a scheme of  $m$ th order accuracy, we must seek approximations such that

$$\frac{F_{j+1/2} - F_{j-1/2}}{h} = \frac{\partial f}{\partial x} \Big|_{x_j} + \mathcal{O}(h^m),$$

with a nonoscillatory reconstruction of the solution or the numerical flux at each cell interface, depending on whether a finite volume or a finite difference formulation is considered.

The desire to formulate schemes of arbitrary order accuracy for general conservation laws has led to essentially nonoscillatory (ENO) schemes and later to weighted essentially nonoscillatory (WENO) schemes, both of which have proven to be very powerful and versatile computational methods. Their development was initiated in the pioneering work [14] which introduced the concept of ENO schemes. Later, this approach was substantially enhanced in [33, 34], proposing the methods we shall shortly discuss. The development of the WENO schemes was initiated in [23], with an additional substantial development presented in [17]. Since then, these methods have undergone a rapid development and have been deployed to solve very complex problems across a wide range of application domains. We refer to [31, 32] for extensive reviews of many of these developments.

## 11.1 • Interpolation and reconstruction

Let us take a step back and consider a general function  $v$  represented by its cell averages  $\bar{v}_j$  at the grid points  $x_j$ . This function may either express the solution or the flux, depending on whether a finite volume or a finite difference scheme is considered. Our

objective is to identify an approach that allows the reconstruction of  $v$  at the cell interfaces  $x_{j+1/2}$  with a given accuracy.

The first natural question that arises is whether such a reconstruction is possible for a general piecewise smooth function  $v$ . The answer is offered by the following result [14].

**Theorem 11.1.** *For a piecewise smooth function,  $v$ , with a finite number of jump discontinuities, there exists an  $h_0 > 0$  such that*

$$\Pi(x; v) = v(x) + \mathcal{O}(h^{m+1}) \quad \forall h < h_0,$$

where  $\Pi(x; v)$  is a polynomial of order  $m$  with the property

$$\Pi(x_j; v) = v(x_j), \quad j = -m/2, \dots, m/2,$$

at  $m + 1$  nodal points. Furthermore,

$$\|\Pi(x; v)\|_{TV} \leq \|v\|_{TV} + \mathcal{O}(h^{m+1}).$$

**Proof.** Let us fix  $x$ . Then there exists a sufficiently small  $h < h_0$  such that  $v$  is either smooth in the entire interval  $(x - \frac{mb}{2}, x + \frac{mb}{2})$  or it has at most one point of discontinuity.

If we first assume that  $v$  is smooth, standard polynomial interpolation yields

$$\Pi(x; v) = v(x) + \mathcal{O}(h^{m+1}).$$

Let us now assume that a discontinuity is located at  $y_0$  and express

$$v(x) = \delta H(x - y_0) + w(x),$$

where  $H$  is the Heaviside function and  $\delta$  reflects the strength of the discontinuity. We assume, for simplicity, that  $y_0 \in [x_0, x_0 + h[$ . In this case,  $w(x)$  is at least continuous and  $\Pi(x; w)$  has a bounded derivative.

If we now consider  $\Pi(x; H)$  it interpolates  $H(x)$  at  $m + 1$  points. Furthermore, in all intervals outside of  $[x_0, x_0 + h[$ , it interpolates a constant function. As a consequence of Rolle's theorem,  $\Pi(x; H)$  must therefore have  $m - 1$  extrema in these elements. Since  $\Pi(x; H)$  is a polynomial of order  $m$ , it must therefore be monotone for  $x \in [x_0, x_0 + h[$ . This implies that

$$\|\Pi(x; v)\|_{TV} \leq \|v\|_{TV} + \mathcal{O}(h^{m+1}),$$

hence completing the proof.  $\square$

Thus, we can construct a function that does not increase the total variation, except by a small amount that is proportional to the truncation error [15].

**Example 11.2.** Let us consider the function  $v(x) = x^3$ , and seek an interpolating polynomial of order 2 that interpolates  $v(x)$  at  $x = [0, h, 2h]$ . Simple algebra yields the polynomial

$$\Pi(x) = 3hx^2 - 2h^2x,$$

which has an extrema at  $x^* = h/3$ . Hence, we have

$$v(x^*) - \Pi(x^*) = -h^3/3 = \mathcal{O}(h^3).$$

Theorem 11.1 guarantees that this is the worst case scenario.  $\blacksquare$

If  $\Pi(x; v)$  satisfies the property that

$$\|\Pi(x; v)\|_{TV} \leq \|v\|_{TV} + \mathcal{O}(h^{m+1}),$$

we refer to it as being essentially nonoscillatory (ENO).

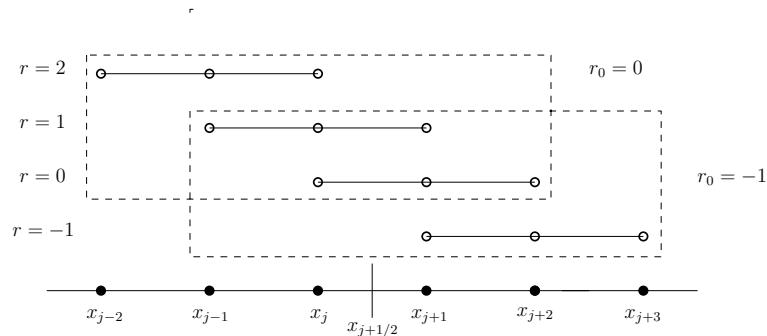
Let us consider a stencil of cell averages  $\{\bar{v}_{j-p}, \dots, \bar{v}_{j+q}\}$ , and seek a polynomial  $\pi$  of order  $m-1$  such that

$$v(x_{j+1/2}) = \pi(x_{j+1/2}) + \mathcal{O}(h^m),$$

to ensure accuracy. Furthermore, we require

$$\frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} \pi(x) dx = \bar{v}_i, \quad \forall i = j-p-r, \dots, j+q-r,$$

to enforce conservation of the cell averages. We introduce  $r$  to enable a shift of the stencil. As we shall see shortly, this is a key element of the scheme.



**Figure 11.1.** Illustration of possible stencils for  $m = 3$ . The stencils are distinguished by starting at different values of  $r$ . Also illustrated are two families of stencils, each with  $2m-1$  nodes, and characterized by  $r_0$ .

Let us generalize this setup slightly and consider a stencil of  $m$  points, selected across  $m$  available stencils, as illustrated in Fig. 11.1. Clearly, the best we can hope for is that  $m = p + q + 1$ , as in this case all points are used.

We seek a general expression for the reconstruction in the form

$$v_{j+1/2}^{(r)} = \sum_{i=0}^{m-1} c_{ir}^m \bar{v}_{j-r+i}, \quad (11.1)$$

where we note the explicit dependence on  $r$ . For consistency, we must require that

$$\sum_{i=0}^{m-1} c_{ir}^m = 1.$$

Furthermore, if we require that all stencils contain either  $\bar{v}_j$  or  $\bar{v}_{j+1}$  then  $-1 \leq r \leq m-1$ , as illustrated in Fig. 11.1.

To recover the coefficients  $c_{ir}^m$ , we generalize the approach developed for the slope limited scheme and consider the primitive

$$\frac{dV(x)}{dx} = v(x) \Rightarrow V(x) = \int_x v(\xi) d\xi.$$

For simplicity, we assume the primitive is defined on  $\mathbb{R}$ . First observe that

$$V(x_{j+1/2}) = \sum_{i=-\infty}^j \int_{x_{i-1/2}}^{x_{i+1/2}} v(\xi) d\xi = h \sum_{i=-\infty}^j \bar{v}_i.$$

Thus, if the cell averages of  $v$  are known, so is  $V$  at all interface points.

Now define a polynomial  $\Pi$  of order  $m$  as the interpolation of  $V$  at the cell interfaces, i.e.,

$$\Pi(x_{j-r+i-1/2}) = V(x_{j-r+i-1/2}), \quad i = 0, \dots, m. \quad (11.2)$$

Let  $\pi = \Pi'$  and express its cell average as

$$\begin{aligned} \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \pi(\xi) d\xi &= \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \Pi'(\xi) d\xi \\ &= \frac{1}{h} [\Pi(x_{j+1/2}) - \Pi(x_{j-1/2})] = \frac{1}{h} [V(x_{j+1/2}) - V(x_{j-1/2})] \\ &= \frac{1}{h} \left[ \sum_{i=-\infty}^j \int_{x_{i-1/2}}^{x_{i+1/2}} v(\xi) d\xi - \sum_{i=-\infty}^{j-1} \int_{x_{i-1/2}}^{x_{i+1/2}} v(\xi) d\xi \right] \\ &= \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} v(\xi) d\xi = \bar{v}_j. \end{aligned}$$

Since  $\Pi$  is an  $\mathcal{O}(h^{m+1})$  approximation to  $V$ , we recover

$$v(x) = \pi(x) + \mathcal{O}(h^m).$$

Hence,  $\pi$  is exactly the polynomial we are after.

To construct  $\pi$  and, ultimately, the coefficients  $c_{ir}^m$  in (11.1), we recall (11.2) to obtain

$$\Pi(x) = \sum_{i=0}^m V(x_{j-r+i-1/2}) \ell_i^{(j-r-1/2)}(x), \quad x \in [x_{j-r-1/2}, x_{j-r+m-1/2}],$$

where

$$\ell_i^{(a)}(x) = \frac{\omega(x)}{\omega'(x_{i+a})(x - x_{i+a})}, \quad \omega(x) = \prod_{i=0}^m (x - x_{i+a}), \quad (11.3)$$

is the  $m$ th order Lagrange polynomial based on  $[x_a, \dots, x_{m+a}]$  and  $\ell_i^{(a)}(x_{k+a}) = \delta_{ik}$ . We recall that the Lagrange polynomial forms a partition of unity as

$$\sum_{i=0}^m \ell_i^{(a)}(x) = 1. \quad (11.4)$$

Now consider

$$\Pi(x) - V(x_{j-r-1/2}) = \sum_{i=0}^m [V(x_{j-r+i-1/2}) - V(x_{j-r-1/2})] \ell_i^{(j-r-1/2)}(x),$$

and recall that

$$V(x_{j-r+i-1/2}) - V(x_{j-r-1/2}) = \int_{x_{j-r-1/2}}^{x_{j-r+i-1/2}} v(\xi) d\xi = b \sum_{q=0}^{i-1} \bar{v}_{q+j-r},$$

to obtain

$$\pi(x) = \Pi'(x) = b \sum_{i=0}^m \sum_{q=0}^{i-1} \bar{v}_{q+j-r} (\ell_i^{(j-r-1/2)})'(x). \quad (11.5)$$

To recover the coefficients  $c_{i,r}^m$  we need to evaluate  $\pi(x_{j+1/2})$ . The identity

$$\sum_{i=0}^m \sum_{j=0}^{i-1} a_j b_i = \sum_{j=0}^{m-1} a_j \sum_{i=j+1}^m b_i$$

allows us to rewrite (11.5) as

$$\pi(x_{j+1/2}) = b \sum_{q=0}^{m-1} \bar{v}_{q+j-r} \sum_{i=q+1}^m (\ell_i^{(j-r-1/2)})'(x_{j+1/2}).$$

Restricting our attention to grids with a constant grid size  $b$ , we recover

$$c_{i,r}^m = \sum_{q=i+1}^m \frac{\sum_{\substack{s=0 \\ s \neq q}}^m \prod_{\substack{t=0 \\ t \neq q}}^m (r+1-t)}{\prod_{\substack{s=0 \\ s \neq q}}^m (s-q)}$$

by evaluating the derivative of the Lagrange polynomials at the grid points. This expression can be further simplified as

$$c_{i,r}^m = \sum_{q=i+1}^m \alpha_q, \quad \alpha_q = \begin{cases} \frac{f(r+1, m)}{f(q, m)} \frac{1}{r+1-q}, & q \neq r+1, \\ -(H_{m-r-1} - H_{r+1}), & q = r+1, \end{cases}$$

where

$$\prod_{\substack{s=0 \\ s \neq q}}^m (s - q) = f(q, m) = (-1)^{q+m} q! (m - q)!$$

and  $H_n$  is the harmonic function. In `ReconstructWeights.m` we illustrate the computation of these coefficients for arbitrary choices of  $m$  and  $-1 \leq r \leq m - 1$ .

**Script 11.1. *ReconstructWeights.m: Computation of the coefficients for reconstruction of interface values from cell averages, returning an  $m$ th order stencil, shifted  $r$  cells to the left of  $j$ .***

---

```
function [c] = ReconstructWeights(m, r)
% Purpose: Compute weights c_ir for reconstruction
%           v_{j+1/2} = \sum_{j=0}^{m-1} c_{ir} v_{i-r+j}
%           with m=order and r=shift (-1 <= r <= m-1).
c = zeros(1, m); fh = @(s) (-1)^(s+m)*prod(1:s)*prod(1:(m-s));
for i=0:m-1
    q = linspace(i+1, m, m-i);
    for q=(i+1):m
        if (q-r+1)
            c(i+1) = c(i+1) + fh(r+1)/fh(q)/(r+1-q);
        else
            c(i+1) = c(i+1) - (harmonic(m-r-1)-harmonic(r+1));
        end
    end
end
return
```

---

To evaluate  $\pi(x_{j-1/2})$ , we use the same stencil, albeit shifted to the left as

$$v(x_{j-1/2}) \simeq v_{j-1/2}^{(r)} \simeq \pi(x_{j-1/2}) = \sum_{i=0}^{m-1} c_{i,r-1}^m \bar{v}_{j-r+i}.$$

Having obtained the coefficients  $c_{i,r}^m$ , we can approximate the value of  $v(x_{j+1/2})$  to an accuracy of  $\mathcal{O}(h^m)$  using  $m$  points. However, as illustrated in Fig. 11.1, there are several ways to accomplish this using a total of  $2m - 1$  points, parameterized by  $r_0$ . It is therefore natural to ask if we can find coefficients  $d_r^{r_0}$  such that

$$v(x_{j+1/2}) \simeq v_{j+1/2} = \sum_{r=r_0}^{m-1} d_r^{r_0} v_{j+1/2}^{(r)} \quad (11.6)$$

expresses an approximation to  $v(x_{j+1/2})$  of order  $\mathcal{O}(h^{2m-1})$ . Here  $r_0$  represents the starting point of  $r$ . If we again require that all stencils contain either  $\bar{v}_j$  or  $\bar{v}_{j+1}$ , then only  $r_0 = -1, 0$  are possible values. Here  $r_0 = 0$  reflects full upwinding and  $r_0 = -1$  a more central scheme.

The coefficients in (11.6) can be recovered by matching orders of accuracy, which results in

$$\forall i = 0 \dots m-1: \sum_{j=0}^{m-1-i} c_{i+j, j+r_0}^m d_{j+r_0}^{r_0} = c_{m-1+i, m-1+r_0}^{2m-1},$$

as  $m$  equations for the  $m$  unknowns,  $d_{r_0}^{r_0}, \dots, d_{m-1+r_0}^{r_0}$ .

In `LinearWeights.m` the computation of these coefficients is illustrated for arbitrary choices of  $m$  and  $r_0 = -1, 0$ .

**Script 11.2. *LinearWeights.m: Computation of the linear weights for the reconstruction of interface values from cell averages, based on  $m-1$  stencils of  $m$  points, yielding a maximal accuracy of  $\mathcal{O}(h^{2m-1})$ .***

---

```
function [d]=LinearWeights (m, r0) ;
% Purpose: Compute linear weights for maximum accuracy 2m-1,
% using stencil shifted $r_0=-1,0$ points upwind.
A = zeros (m,m) ; b = zeros (m,1) ;

% Setup linear system for coefficients
for i=1:m
    col = ReconstructWeights (m, i-1+r0) ;
    A(1:(m+1-i), i) = col (i:m) ;
end

% Setup righthand side for maximum accuracy and solve
crhs = ReconstructWeights (2*m-1,m-1+r0) ;
b = crhs (m:(2*m-1))' ; d = A\ b;
return
```

---

For consistency (11.4) we expect that

$$\sum_{r=r_0}^{m-1+r_0} d_r^{r_0} = 1.$$

Table 11.1 shows the computed coefficients for a few values of  $m$ . We note that all values of  $d_r^{r_0}$  are positive, i.e.,  $v_{j+1/2}$  is recovered as a convex combination of  $v_{j+1/2}^{(r)}$ . Furthermore, we observe that

$$d_r^{-1} = d_{m-2-r}^0$$

as a consequence of the symmetry of the stencils.

Finally, by symmetry we recover that

$$v(x_{j-1/2}) \simeq v_{j-1/2} = \sum_{i=0}^{m-1} d_{i,m-1-r}^{r_0} v_{j-1/2}^{(r)},$$

as an  $\mathcal{O}(h^{2m-1})$  approximation to  $v(x_{j-1/2})$ .

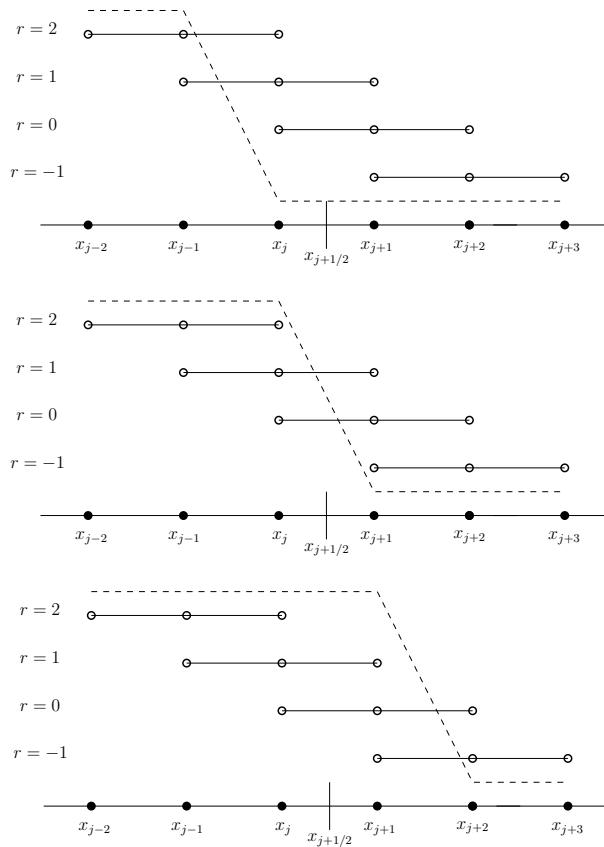
**Table 11.1.** Coefficients to recover maximal linear accuracy of  $\mathcal{O}(h^{2m-1})$  by a convex combination of  $m$  stencils of size  $m$ .

$m$	$r_0$	$d_{-1}^{r_0}$	$d_0^{r_0}$	$d_1^{r_0}$	$d_2^{r_0}$	$d_3^{r_0}$
1	-1	1.0000				
	0		1.0000			
2	-1	0.3333	0.6667			
	0		0.6667	0.3333		
3	-1	0.1000	0.6000	0.3000		
	0		0.3000	0.6000	0.1000	
4	-1	0.0286	0.3429	0.5143	0.1143	
	0		0.1143	0.5143	0.3429	0.0286

## 11.2 • ENO methods

Although the developments in the previous section offer substantial flexibility in the reconstruction at a cell interface, it remains unclear how to best take advantage of this. The answer to this, proposed first in the pioneering works [14, 33, 34], leads to the ENO methods.

To gain insight into this, consider the situation depicted in Fig. 11.2. It is clear that we should seek a reconstruction that is as accurate as possible. As previously discussed, this suggests that we avoid polynomial reconstructions across cells that include a jump. As illustrated in Fig. 11.2, this can be achieved by changing the value of  $r$ , depending on where the discontinuity is located. For instance, in the top figure this is ensured by the stencil with  $r = 0$ , while in the center figure  $r = 2$  is the appropriate choice to reconstruct the value at  $x_{j+1/2}$ .



**Figure 11.2.** Possible stencil choices for  $m = 3$  and a total of  $2m - 1$  points, as the solution (dashed) moves across the cell interface  $x_{j+1/2}$ , where we seek to reconstruct.

If we adopt this strategy, the question of how to systematically select one stencil among the  $m$  possible stencils is left open. Ideally, the best stencil is the one that utilizes the least oscillatory approximation. In other words, we need to identify the smoothest stencil among all the candidates.

Let us take one step back and consider the construction of an interpolating polynomial  $\pi$ , based on the  $m + 1$  nodes,  $x_j$ , and values,  $v_j$ , for  $j = 0, \dots, m$ . Rather than using the Lagrange polynomials we construct  $\pi$  in the Newton form. Hence, as a first step, set  $\pi_0(x) = v_0$ . We proceed to construct  $\pi$  as

$$\pi_1(x) = \pi_0(x) + a_1(x - x_0), \quad a_1 = \frac{v_1 - v_0}{x_1 - x_0},$$

where the second term follows by requiring that  $\pi$  is an interpolant at  $x_0, x_1$ . The general expression for a polynomial of order  $m$  on Newton form is easily recovered as

$$\pi_m(x) = \sum_{i=0}^m a_i \prod_{j=0}^{i-1} (x - x_j).$$

We realize that each coefficient  $a_i$  depends solely on  $(x_j, v_j)$  for  $j = 0, \dots, i$ , which we express as  $a_i = v[x_0, \dots, x_i]$ . To evaluate these coefficients, consider

$$\pi_m(x) = \sum_{i=0}^m a_i \prod_{j=0}^{i-1} (x - x_j) = \sum_{i=0}^m b_i \prod_{j=m-i}^m (x - x_j).$$

The latter expression emerges by reversing the order of the nodes,  $x_j$ . Since both expressions represent the unique interpolation polynomial, their terms must match by order. Hence, we conclude that  $a_m = b_m$ . Considering the next term, we have that

$$a_{m-1} - a_m \sum_{j=0}^{m-1} x_j = b_{m-1} - b_m \sum_{j=1}^m x_j, \Rightarrow a_{m-1} - b_{m-1} = a_m(x_m - x_0).$$

If we further recall that  $a_{m-1} = v[x_0, \dots, x_{m-1}]$  and  $b_{m-1} = v[x_1, \dots, x_m]$ , we recover the recurrence

$$a_m = \frac{v[x_0, \dots, x_{m-1}] - v[x_1, \dots, x_m]}{x_m - x_0}.$$

This allows for the straightforward computation of  $a_m$  and, thus, all coefficients for increasing  $m$ . This expression is known as Newton divided differences and the computation of these coefficients is shown in `ddnewton.m`.

**Script 11.3. `ddnewton.m`: Computation of the table of Newton divided differences from a vector of point values.**

---

```

function [maxDDN,DDNmat] = ddnewton(x,v)
% Purpose: Create the table of Newtons divided differences based on (x,v)
m = length(x); DDNmat = zeros(m,m+1);

% Inserting x into the 1st column and f into 2nd column of table
DDNmat(1:m,1) = x; DDNmat(1:m,2) = v;

% create divided difference coefficients by recurrence
for j = 1:m-1
    for k = 1:m-j
        DDNmat(k,j+2) = ...
            (DDNmat(k+1,j+1)-DDNmat(k,j+1)) / (DDNmat(k+j+1)-DDNmat(k+1));
    end
end

```

```

    end
end

% extract max coefficient
maxDDN = abs (DDNmat(1 ,m+1));
return

```

To realize the importance of the divided differences in the current context, consider

$$\alpha_1 = \frac{v_1 - v_0}{x_1 - x_0}.$$

If  $v$  is a smooth function,  $\alpha_1 = v'(\xi)$  for some  $\xi \in [x_0, x_1]$ . On the contrary, if  $v$  is not smooth, then  $\alpha_1 \simeq h^{-1}$ , which increases as the grid is refined. Using the same argument repeatedly, we realize

$$\alpha_m = \begin{cases} \frac{v^{(m)}(\xi)}{m!}, & v \text{ smooth,} \\ h^{-m}, & v \text{ nonsmooth.} \end{cases}$$

This suggests that the divided differences allow us to understand the smoothness of  $v$ . Furthermore, if we need to consider different functions, we may decide which of the functions exhibit the smoothest behavior. Revisiting Fig. 11.2 this information is exactly what is needed to select the appropriate stencil.

This observation is at the heart of the ENO method [14, 33, 34]. The stencil selection is a bottom-up process in the following sense. Given an initial stencil  $S_l = \{x_l\}$  we construct two new candidate stencils,  $S^a = \{x_l^-, S_l\}$  and  $S^b = \{S_l, x_l^+\}$  by adding one node to the left,  $x_l^-$ , and one to the right,  $x_l^+$ , as well as the corresponding function to the current stencils. We identify the one among these two candidates with the smallest divided difference and rename it as  $S_{l+1}$ . Once the smoothest stencil with  $m$  nodes has been identified, this selects  $r$  for the active cell, and the appropriate coefficients can be computed to recover the  $\mathcal{O}(h^m)$  accurate reconstruction.

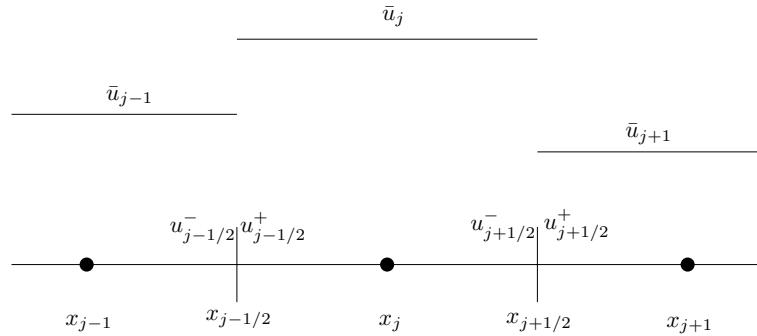
### 11.2.1 • ENO method for conservation laws

Let us now employ this nonoscillatory reconstruction to the solution of conservation laws. Consider the general semidiscrete form

$$\frac{dU_j}{dt} = -\frac{F_{j+1/2} - F_{j-1/2}}{h},$$

where  $F_{j\pm 1/2}$  is the numerical flux. Focusing on finite volume methods the reconstruction allows the evaluation of  $u_{j+1/2}^\pm$  as a left and right state of the generalized Riemann problem; see Fig. 11.3. Hence, we recover the finite volume method directly by  $F_{j+1/2} = F(u_{j+1/2}^-, u_{j+1/2}^+)$ , where  $F(u, v)$  is a monotone flux. The difference to the monotone scheme is the explicit reconstruction of the left and right states of the Riemann problem.

As discussed in the introduction of Chapter 10, the differences between the finite difference and the finite volume schemes can be reduced to the question of which quantities are being reconstructed, i.e., the solution  $u$  is reconstructed in the finite volume method while the flux is reconstructed in the finite difference method. As we



**Figure 11.3.** Illustration of the terminology for the reconstruction of the interface values, needed to advance the conservation law.

have seen on several occasions, some degree of upwinding is needed for stability. While this is ensured by the solution of the Riemann problem in the finite volume scheme, in the case of the finite difference scheme it must be introduced through the numerical flux and, thus, as part of the reconstruction process.

The simplest way to achieve this is to split the flux as

$$f(u) = f^+(u) + f^-(u), \quad f_u^+ \geq 0, \quad f_u^- < 0,$$

and perform the reconstruction with upwinding or downwinding on  $f^+$  or  $f^-$ , respectively. A simple approach to this flux splitting is the Lax-Friedrichs splitting

$$f^\pm(u) = \frac{1}{2}(f(u) \pm \alpha u), \quad \alpha \geq \max_u |f'(u)|.$$

Naturally, other splittings are possible [33].

### Extension to systems

Let us consider the more general case of a system of conservation laws,

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = \frac{\partial u}{\partial t} + A \frac{\partial u}{\partial x} = 0,$$

where  $A = \nabla_u f$  is the flux Jacobian. As usual, we assume that  $A$  is uniformly diagonalizable,  $A = SAS^{-1}$ , with purely real eigenvalues,  $\Lambda_{ii}$ .

If we first consider the finite difference scheme, then flux splitting

$$f(u) = f^+(u) + f^-(u)$$

can be achieved in the same fashion as for the scalar case by setting  $\alpha = \max_u |\Lambda(u)|$ . A possible alternative is the Roe-like flux splitting as

$$f^\pm(u) = \frac{1}{2}(f(u) \pm \tilde{A}u), \quad \tilde{A} = S\tilde{\Lambda}S^{-1}, \quad \tilde{\Lambda} = \text{diag}\left(\max_u |\lambda_1|, \dots, \max_u |\lambda_m|\right).$$

There are two different approaches for the finite volume scheme. In the first one, often referred to as the componentwise approach, the ENO reconstruction is applied to each

of the conserved variables of the system and the numerical flux is computed. While this approach often works well, one may observe that for higher-order reconstructions, e.g.,  $m > 2$ , or for complex and highly nonlinear problems, oscillations may appear, similar to what we discussed in Chapter 10.

As explained in subsection 10.1.4 and discussed in the context of essentially nonoscillatory schemes in [27], the solution to this problem is to perform the reconstruction on the characteristic variables. If we assume that the smoothest stencil has been identified through the divided differences, we form an intermediate value  $\tilde{u}_{j+1/2}$  and evaluate  $S$  and  $\Lambda$  at the interface  $x_{j+1/2}$ . There are several ways to approximate this intermediate value with a simple average  $\tilde{u}_{j+1/2} = \frac{1}{2}(\bar{u}_j + \bar{u}_{j+1})$  often being an adequate choice. More advanced choices, such as the Roe average, are also possible.

Once  $\tilde{u}_{j+1/2}$  is determined, we can recover the characteristic variables in the neighborhood of the cell interface and, subsequently, reconstruct them at the cell interface. Once the characteristic variables are reconstructed at the cell interfaces, the conserved variables can be recovered by  $S$ . We shall discuss the details of this approach in subsection 11.3.4.

Since  $S$  and  $\Lambda$  depend on  $\tilde{u}_{j+1/2}$ , these are different for every grid point in the computational domain. This local nature of the characteristic variables increases the computational cost of the characteristic approach considerably as compared to the approach based on the conservative variables. For the finite difference scheme, a similar approach can be implemented by using a scalar flux splitting for each of the characteristic fluxes.

### 11.2.2 • A bit of theory

Although the ENO schemes were proposed in the mid-1980s and have been used with substantial success since then, a robust theoretical understanding remains elusive. To appreciate the difficulties of such analysis, let us follow [25] and write the numerical scheme as

$$\bar{u}_j^{n+1} = G(\bar{u}^n)_j = \bar{u}_j^n - \lambda \left[ F_{j+1/2}^n - F_{j-1/2}^n \right].$$

This serves as an approximation to the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \Rightarrow u(t) = S(t)u_0,$$

where  $u_0$  is the initial condition and  $S(t)$  is the solution operator. The numerical scheme seeks to approximate

$$\bar{u}_j(t) = \mathcal{A}_b S(t) \pi(x; u_0),$$

where  $\pi(x; u)$  is the reconstruction polynomial and  $\mathcal{A}_b$  represents the cell averaging operator

$$\mathcal{A}_b u(x) = \frac{1}{b} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x) dx.$$

If we consider the total variation, we have

$$\|\mathcal{A}_b S(t) \pi(x; u_0)\|_{TV} \leq \|\pi(x; u_0)\|_{TV},$$

since both the averaging operation and the solution operator are monotone. Hence, if the reconstruction is ENO, we recover

$$\|\mathcal{A}_b S(t) \pi(x; u_0)\|_{TV} \leq \|\pi(x; u_0)\|_{TV} \leq \|\bar{u}_0\|_{TV} + \mathcal{O}(h^m),$$

which suggests TVD-stability for the scalar case by Theorem 11.1. However, this last conclusion relies on an assumption that there exists  $h_0 > 0$  such that  $u$  is smooth and has at most one discontinuity in this interval. This is generally not the case for the numerical solution  $\bar{u}^n$ . Hence, Theorem 11.1 does not apply. Consequently, we are left without any notion of stability and, thus, convergence.

If we were to conjecture ENO TVD-stability of the form

$$\|\bar{u}^{n+1}\|_{TV} \leq \|\bar{u}^n\|_{TV} + \mathcal{O}(h^{m+1}),$$

we could expect a cumulative accuracy  $\mathcal{O}(h^m)$  at a final time  $T$ , provided the exact solution is sufficiently smooth. Naturally, the cell averages can be used to reconstruct the pointwise approximation to the solution at a similar accuracy.

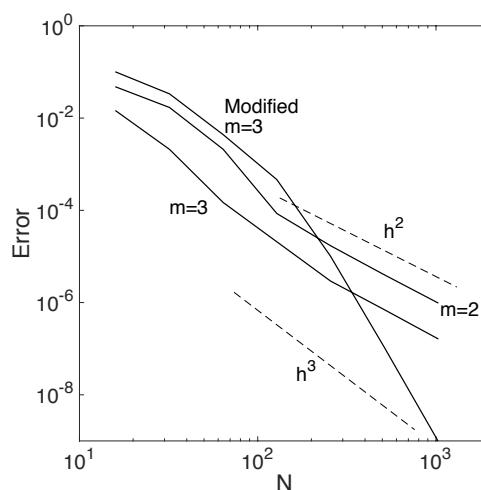
However, the accuracy of the scheme is subtle, as illustrated in the following example [28, 30].

**Example 11.3.** We consider the linear wave equation

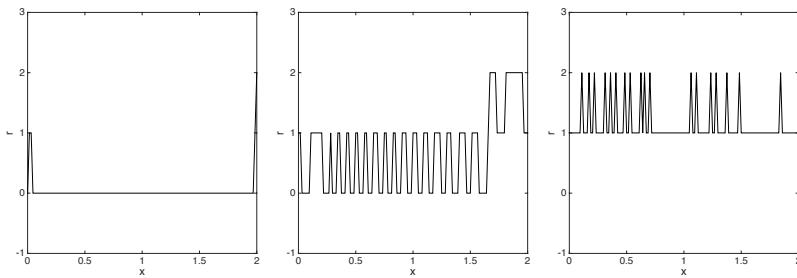
$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [0, 2],$$

subject to periodic boundary conditions. A nonperiodic initial condition  $u_0(x) = \exp(-x)$  is used. This creates an initial shock at  $x = 0$ .

We employ an ENO scheme, a Lax-Friedrichs flux, and a third order strong stability preserving Runge-Kutta (SSPERK) scheme to solve the problem and vary the order of the ENO reconstruction  $m$ . Figure 11.4 shows the error of the computed result, measured at  $T = 4$ . Since the solution is discontinuous, we only measure the error



**Figure 11.4.** Error for the linear advection equation with an ENO reconstruction of order  $m$  and initial condition  $u_0(x) = e^{-x}$ .



**Figure 11.5.** The stencil selection value  $r$  for the initial condition (left) at  $T = 4$  using a standard ENO reconstruction (center), and when using a slightly modified upwind biased stencil selection process (right).

over  $[0.5, 1.5]$ . As expected, we observe an accuracy of  $\mathcal{O}(h^2)$  for  $m = 2$ . However, for  $m = 3$ , the accuracy deteriorates to a suboptimal second order of convergence.

To understand this phenomenon, we plot the value of  $r$  chosen by the ENO reconstruction of the initial condition in Fig. 11.5 and observe that it is zero throughout the computational domain, except very close to the point of discontinuity. We recall that  $r = 0$  implies a maximally downwinded stencil which results in a linearly unstable scheme [35]. However, rather than becoming unstable, the nonlinear ENO reconstruction overcomes the linear instability by constantly switching the stencil to add stability to the scheme through upwinding, as illustrated in Fig. 11.5.

Guided by this insight, it is natural to seek a slightly modified stencil selection procedure that is more biased towards a central stencil. A simple strategy is to allow downwinding only when the divided differences indicate that this stencil is strongly preferred and, otherwise, prefer upwinding. Naturally, extreme upwinding can be controlled in the same way.

The error of the scheme with  $m = 3$  and an upwind biased stencil selection procedure is shown in Fig. 11.4. As expected the order of accuracy is more than recovered. The behavior of the stencil selection parameter  $r$ , shown in Fig. 11.5, illustrates the shift of the stencil in the upwind direction, resulting in increased stability and improved accuracy. ■

The only significant result that is known for the properties of the ENO reconstruction is the sign property [9].

**Theorem 11.4.** Consider the ENO reconstruction of order  $m$ , based on cell averages. Let  $\bar{u}_j^n$  and  $\bar{u}_{j+1}^n$  be the cell averages in cells  $x_j$  and  $x_{j+1}$ , respectively, and also let  $u_{j+1/2}^\pm$  refer to the reconstructed values of  $u$  at the interface (see Fig. 11.3 for notation). Then it holds

$$0 \leq \frac{u_{j+1/2}^+ - u_{j+1/2}^-}{\bar{u}_{j+1}^n - \bar{u}_j^n} \leq C,$$

where the constant  $C$  depends only on the order of reconstruction  $m$  and the local mesh.

Furthermore, if  $\bar{u}_{j+1}^n = \bar{u}_j^n$ , the ENO reconstruction is continuous across the interface,  $u_{j+1/2}^+ = u_{j+1/2}^-$ .

The proof of this result is lengthy and relies on a very careful study of the properties of the stencils and their selection. Hence, this is not a universal result, but is tied closely to the stencil selection procedure discussed above. However, the interpretation of the result is simple, as it guarantees that the sign of the jump in  $\bar{u}_j$  to  $\bar{u}_{j+1}$  is maintained in the reconstructed interface values. Moreover, the relative size of the jumps is uniformly bounded.

The sign property has been explored in [8, 44] to develop entropy stable, high-order accurate ENO schemes. As this is the only known substantial stability result for ENO schemes, let us briefly discuss how the sign property enters as a key element of this result. For simplicity, we restrict attention to the scalar case.

As discussed in subsection 8.2.2, we seek methods that satisfy

$$\eta(U^{n+1})_j - \eta(U^n)_j + \frac{k}{h} (\Psi(U^n)_{j+1/2} - \Psi(U^n)_{j-1/2}) \leq 0,$$

to ensure entropy stability. Here  $(\eta, \Psi)$  represents the convex entropy and the entropy flux, respectively.

For the scalar case we recall that

$$\Delta^+ U_j F_{j+1/2}^* = \zeta_{j+1} - \zeta_j, \quad F_{j+1/2}^* = \int_0^1 f(U_j + \xi \Delta^+ U_j) d\xi$$

which yields the entropy conservative flux

$$\Psi_{j+1/2} = \frac{1}{2} (U_j + U_{j+1}) F_{j+1/2}^* - \frac{1}{2} (\zeta_j + \zeta_{j+1}).$$

This scheme is second order accurate. Its generalization to arbitrarily high-order schemes was discussed in Theorem 8.16. For smooth solutions, the use of this flux suffices. When shocks appear, we need to introduce sufficient dissipation to guarantee entropy dissipation, as discussed subsection 8.2.2.

For this, we define a new entropy flux

$$\tilde{\Psi}_{j+1/2} = \Psi_{j+1/2} + \frac{1}{2} (U_j + U_{j+1}) D_{j+1/2} (U_{j+1} - U_j).$$

As established in Theorem 8.12, this ensures entropy stability and second order accuracy of the scheme, provided  $D_{j+1/2}$  is symmetric and positive definite.

Extending this to higher order requires an entropy flux like

$$\tilde{\Psi}_{j+1/2} = \Psi_{j+1/2} + \frac{1}{2} (U_j + U_{j+1}) D_{j+1/2} (U_{j+1/2}^+ - U_{j+1/2}^-),$$

where we have introduced the reconstructed interface values. Clearly, if we can ensure that there exists a semipositive constant,  $B_{j+1/2}$ , such that

$$(U_{j+1} - U_j) = B_{j+1/2} (U_{j+1/2}^+ - U_{j+1/2}^-),$$

the result on second order accurate entropy stability carries over to the high-order case. However, this is exactly guaranteed by the sign property, stated in Theorem 11.4.

This result, validated extensively in [8] for a variety of problems, carries over to systems provided they are symmetrized, following the discussion in section 3.3. In this case, the diffusion operator  $D_{j+1/2}$  is defined as

$$D_{j+1/2} = S_{j+1/2} \tilde{\Lambda} S_{j+1/2}^T.$$

Here  $S_{j+1/2}$  collects the eigenvectors of the flux Jacobian  $\nabla_u f$  of the system, and  $\tilde{\Lambda}$  is a positive definite diagonal matrix associated with the eigenvalues of the flux Jacobian, e.g.,

$$\tilde{\Lambda} = \text{diag}(|\lambda_1|, \dots, |\lambda_m|) \text{ or } \tilde{\Lambda} = \max(|\lambda_1|, \dots, |\lambda_m|)I.$$

The latter choice can be recognized as being similar to a Lax–Friedrichs flux while the former reflects a Roe flux.

### 11.2.3 • ENO methods in action

With the development of ENO methods and an understanding of their key properties in place, let us now consider the implementation of these techniques to confirm the key results of the analysis.

As we have already discussed, the ENO reconstruction builds the stencils in a bottom-up fashion by comparing two candidate stencils at a time and choosing the one that oscillates the least. This process continues until the stencil contains  $m$  cells as required for accuracy. A concrete implementation is shown in ENO.m.

**Script 11.4. ENO.m: Routine that builds stencil by ENO procedure and returns cell interface values.**

---

```

function [um,up] = ENO(xloc ,uloc ,m,Crec);
% Purpose: Reconstruct the left (um) and right (up) cell interface values
%           using an ENO reconstruction based on 2m-1 long vectors.

% Treat special case of m=1 - no stencil to select
if (m==1)
    um = uloc(1); up = uloc(1);
else
% Apply ENO procedure to build stencil
    S = zeros(m,1); S(1) = m;
    for mm=1:m-1
        % Left stencil
        Sindxl = [S(1)-1; S(1:mm)];
        [Vl,DDNmat] = ddnewton(xloc(Sindxl),uloc(Sindxl));

        % Right stencil
        Sindxr = [S(1:mm); S(mm)+1];
        [Vr,DDNmat] = ddnewton(xloc(Sindxr),uloc(Sindxr));

        % Choose stencil by divided differences
        S(1:mm+1) = Sindxl;
        if (Vl>Vr) S(1:mm+1) = Sindxr; end;
    end

    % Compute stencil shift 'r' and cell interface values
    r = m - S(1);
    up = Crec(r+2,:)*uloc(S); um = Crec(r+1,:)*uloc(S);
end
return

```

---

With higher-order accuracy being the goal, we need a careful definition of the initial conditions and their cell average. Clearly, if the initial condition is piecewise constant, no special attention is needed. However, for a more general initial condition, we must evaluate

$$\bar{u}_j(0) = \frac{1}{b} \int_{x_{j-1/2}}^{x_{j+1/2}} u_0(x) dx = \frac{1}{2} \int_{-1}^1 u_0\left(x_j + \frac{b}{2}y\right) dy,$$

where  $u_0$  is the initial condition.

The cell average is approximated with a Legendre–Gauss quadrature

$$\bar{u}_j(0) = \frac{1}{2} \sum_{i=0}^{N_{GQ}} u_0\left(x_j + x_i^{GQ} \frac{b}{2}\right) \omega_i^{GQ},$$

where  $(x^{GQ}, \omega^{GQ})$  are the  $N_{GQ} + 1$  quadrature nodes and weights, respectively. Classic analysis [6] shows that this quadrature is exact for polynomials up to order  $2N_{GQ} + 1$  and it is the optimal quadrature for smooth functions.

`LegendreGQ.m` implements the computation of the  $N_{GQ} + 1$  Legendre–Gauss quadrature nodes and weights, based on the classic approach in [12].

**Script 11.5. *LegendreGQ.m: Routine for computing nodes and weights for a Legendre–Gauss quadrature of order  $m$ .***

---

```
function [x,w] = LegendreGQ(m);
% function [x,w] = LegendreGQ(m)
% Purpose: Compute the m'th order Legendre Gauss quadrature points , x,
%           and weights , w
if (m==0) x(1)=0; w(1) = 2; return; end;

% Form symmetric matrix from recurrence.
J = zeros(m+1); h1 = 2*(0:m);
J = diag(2./(h1(1:m)+2).*...
sqrt((1:m).*((1:m)).*((1:m)).*((1:m))./(h1(1:m)+1)./(h1(1:m)+3)),1);
J(1,1)=0; J = J + J';

%Compute quadrature by eigenvalue solve
[V,D] = eig(J); x = diag(D); w = 2*(V(1,:)').^2;
return
```

---

## Burgers' equation

We begin by considering Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to the piecewise-constant initial condition

$$u^0(x) = \begin{cases} 2 & x < 0.2, \\ 1 & x \geq 0.2. \end{cases}$$

The exact solution is recovered from the Rankine–Hugoniot condition as  $u(x, t) = u^0(x - 0.2 - 3t)$ .

In BurgersENO1D.m we show the driver for the solver which sets the order  $m$  of the ENO solver and defines the initial condition.

**Script 11.6. BurgersENO1D.m: Driver routine for solving Burgers' equation using an ENO scheme.**

---

```
% Driver script for solving the 1D Burgers equations using an ENO scheme
clear all

% Order of method
m=2;

% Set problem parameters
L = 1; FinalTime = 0.2; N = 128; CFL = 0.9; h = L/N;

% Define domain and initial conditions
x = [0:h:L]';
fic = @(x,t) sin(2*pi*x); %periodic BC needed
%fic = @(x,t) (1-sign(x-0.2-3*t))/2+1; % Constant BC needed

% Compute cell averages using Legendre Gauss rule of order NGQ
u = zeros(N+1,1);
NGQ = 10; [xGQ,wGQ] = LegendreGQ(NGQ);
for i=1:NGQ+1
    u = u + wGQ(i)*fic(x+xGQ(i)*h/2,0);
end;
u = u/2;

% Solve Problem
[u] = BurgersENO1D(x,u,h,m,CFL,FinalTime);
```

---

In BurgersENO1D.m the coefficients  $c_{rj}$  are precomputed, as they depend only on  $m$ , and a third order SSPRK method is used to advance the solution in time.

**Script 11.7. BurgersENO1D.m: Time integrator for solving Burgers' equation using an ENO scheme.**

---

```
function [u] = BurgersENO1D(x,u,h,m,CFL,FinalTime)
% function [u] = BurgersENO1D(x,u,h,m,r0,CFL,FinalTime)
% Purpose : Integrate 1D Burgers equation until FinalTime using an ENO
%             scheme and 3rd order SSP-RK method
time = 0; tstep = 0;

% Initialize reconstruction weights
Crec = zeros(m+1,m);
for r=-1:m-1;
    Crec(r+2,:) = ReconstructWeights(m,r);
end;

% integrate scheme
while (time<FinalTime)
    % Decide on timestep
    maxvel = max(2*abs(u)); k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    rhsu = BurgersENOrhs1D(x,u,h,k,m,Crec,maxvel);
    u1 = u + k*rhsu;
    rhsu = BurgersENOrhs1D(x,u1,h,k,m,Crec,maxvel);
    u2 = (3*u + u1 + k*rhsu)/4;
```

---

---

```

rhsu = BurgersENOrhs1D(x,u2,h,k,m,Crec,maxvel);
u = (u + 2*u2 + 2*k*rhsu)/3;
time = time+k; tstep = tstep+1;
end
return

```

---

Finally, routine `BurgersENOrhs1D.m` implements the ENO solver for Burgers' equation. The current solution is first extended with  $m - 1$  points at each end to allow the implementation of boundary conditions. This is similar to what was done for the monotone scheme, albeit the extension is wider to allow for the extended stencil. For each node  $x_j$ , the  $2m - 1$  surrounding points are extracted and the ENO reconstruction computes the approximation to the solution at the cell interfaces,  $x_{j\pm 1/2}$ . Finally, these are applied to evaluate the numerical flux in the conservative scheme.

**Script 11.8. *BurgersENOrhs1D.m: Evaluation of the right-hand side for solving Burgers' equation using an ENO scheme.***

---

```

function [du] = BurgersENOrhs1D(x,u,h,k,m,Crec,maxvel)
% function [du] = BurgersENOrhs1D(x,u,h,k,m,Crec,maxvel)
% Purpose : Evaluate the RHS of Burgers equations using
%           an ENO reconstruction
N = length(x); du = zeros(N,1);

% Extend data and assign boundary conditions
[xe,ue] = extend(x,u,h,m,'P',0,'P',0);

% define cell left and right interface values
um = zeros(N+2,1); up = zeros(N+2,1);

for i=1:N+2
    [um(i),up(i)] = ENO(xe(i:(i+2*(m-1))),ue(i:(i+2*(m-1))),m,Crec);
end;

% Compute residual
du = - (BurgersLF(up(2:N+1),um(3:N+2),0,maxvel) ...
          - BurgersLF(up(1:N),um(2:N+1),0,maxvel))/h;
return

```

---

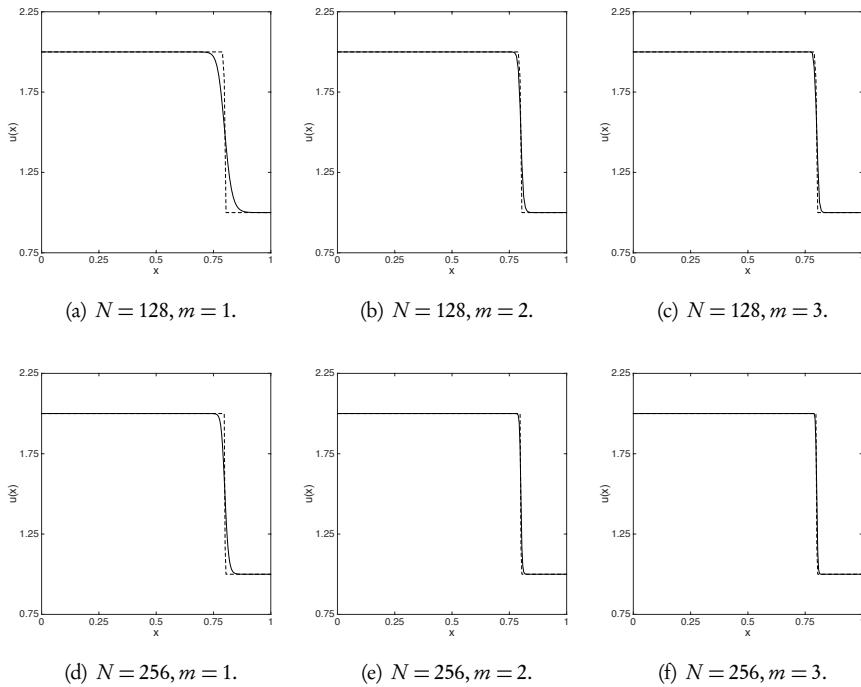
As a first test, we show in Fig. 11.6 the solution for different values of  $N$  and the orders of the approximation  $m$ . Even though the solution is very simple, we observe a clear improvement when increasing the order of approximation from  $m = 1$  to  $m = 3$ . This is particularly evident in the reduced smearing of the shock. As should be expected, no oscillations are observed even at higher order.

When considering the accuracy, we observe that  $\|u - u_b\|_{h,1}$  is  $\mathcal{O}(h)$  while  $\|u - u_b\|_{h,2}$  is  $\mathcal{O}(h^{1/2})$  in Fig. 11.7. This is consistent with expectations for a scalar equation [24, 29], when errors are measured globally.

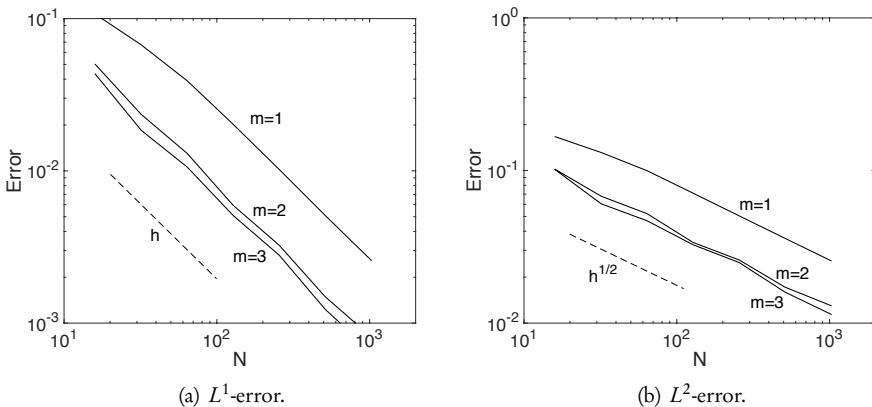
## Maxwell's equations

Let us turn to Maxwell's equations

$$\varepsilon(x) \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x}, \quad x \in [-1, 1],$$



**Figure 11.6.** Burgers' equation at  $T = 0.2$ , solved with  $N$  elements and an ENO scheme of order  $m$ . The solid lines represent the computed solution while the dashed line reflects the exact solution.



**Figure 11.7.** The error of the ENO scheme of order  $m$  when solving Burgers' equation with a propagating shock using  $N$  elements.

where the coefficient,  $\varepsilon(x)$ , is assumed to be piecewise constant

$$\varepsilon(x) = \begin{cases} \varepsilon_1, & x \leq 0, \\ \varepsilon_2, & x > 0, \end{cases}$$

and  $\varepsilon_i$  is positive.

In `MaxwellENODriver1D.m` we show the driver for the solver, which sets the order of the ENO solver and defines the initial condition.

**Script 11.9.** *MaxwellENODriver1D.m: Driver routine for solving the one-dimensional Maxwell's equations using an ENO scheme.*

---

```
% Driver script for solving the 1D Maxwell's equations using an ENO scheme
clear all

% Order of method
m=2;

% Set problem parameters
L = 2; FinalTime = pi/2; N = 256; CFL = 0.90; h = L/N;
ep1 = 1.0; mul = 1.0; epr = 2.25; mur = 1.0;

% Define domain, materials and initial conditions
x = [-1:h:1];
NGQ = 16; [xGQ,wGQ] = LegendreGQ(NGQ);
Ef = zeros(N+1,1); Hf = zeros(N+1,1);
for i=1:NGQ+1
    [Ef,h, ep, mu] = CavityExact(x+xGQ(i)*h/2, ep1, epr, mul, mur, 0);
    Ef = Ef + wGQ(i)*Ef; Hf = Hf + wGQ(i)*Hf;
end
Ef = Ef/2; Hf = Hf/2;

% Solve Problem
q = [Ef Hf];
[q] = MaxwellENO1D(x,q,ep,mu,h,m,CFL,FinalTime);
```

---

In `MaxwellENO1D.m` the coefficients  $c_{rj}$  are precomputed, and a third order SSPERK method is used to advance the solution in time.

**Script 11.10.** *MaxwellENO1D.m: Time integrator for solving the one-dimensional Maxwell's equations using an ENO scheme.*

---

```
function [q] = MaxwellENO1D(x,q,ep,mu,h,m,CFL,FinalTime)
% function [q] = MaxwellENO1D(x,q,ep,mu,h,m,CFL,FinalTime)
% Purpose : Integrate 1D Maxwell's equation until FinalTime using an ENO
% scheme and 3rd order SSP-RK method.
time = 0; tstep = 0;

% Initialize reconstruction weights
Crec = zeros(m+1,m);
for r=-1:m-1;
    Crec(r+2,:) = ReconstructWeights(m, r);
end;

% Set timestep
cvel = 1./sqrt(ep.*mu); k = CFL*h/max(cvel); maxvel = max(cvel);

% integrate scheme
while (time<FinalTime)
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    rhsq = MaxwellENOrhs1D(x,q,ep,mu,h,k,m,Crec,maxvel);
    q1 = q + k*rhsq;
    rhsq = MaxwellENOrhs1D(x,q1,ep,mu,h,k,m,Crec,maxvel);
    q2 = (3*q + q1 + k*rhsq)/4;
```

```

rhsq = MaxwellENOrhs1D(x,q2,ep, mu, h, k, m, Crec, maxvel);
q = (q + 2*q2 + 2*k*rhsq)/3;
time = time+k; tstep = tstep+1;
end
return

```

Finally, `MaxwellENOrhs1D.m` implements the solver with an ENO reconstruction of the fields and applies an appropriately chosen numerical flux.

**Script 11.11. *MaxwellENOrhs1D.m: Evaluation of the right-hand side for the one-dimensional Maxwell's equations solved using an ENO scheme.***

```

function [dq] = MaxwellENOrhs1D(x,q,ep, mu, h, k, m, Crec, maxvel);
% function [dE] = MaxwellENOrhs1D(x,q,ep, mu, h, k, m, Crec, maxvel);
% Purpose: Evaluate right-hand side for Maxwell's equation using ENO method

N = length(x); dq = zeros(N,2);
ql = zeros(N,2); qr = zeros(N,2); qm = zeros(N,2); qp = zeros(N,2);

% Extend data and assign boundary conditions
[xe,Ee] = extend(x,q(:,1),h,m,'D',0,'D',0);
[xe,He] = extend(x,q(:,2),h,m,'N',0,'N',0);

% define cell left and right interface values
Em = zeros(N+2,1); Ep = zeros(N+2,1); Hm = zeros(N+2,1); Hp = zeros(N+2,1);
for i=1:N+2
    [Em(i),Ep(i)] = ENO(xe(i:(i+2*(m-1))),Ee(i:(i+2*(m-1))),m,Crec);
    [Hm(i),Hp(i)] = ENO(xe(i:(i+2*(m-1))),He(i:(i+2*(m-1))),m,Crec);
end;

% Extract interface values
qr = [Ep(2:N+1) Hp(2:N+1)]; ql = [Em(2:N+1) Hm(2:N+1)];
qm = [Ep(1:N) Hp(1:N)]; qp = [Em(3:N+2) Hm(3:N+2)];

% Lax Friedrich flux - or something simple
% dEH = - (MaxwellLF(qr,qp,ep, mu, k/b, maxvel) - ...
%           MaxwellLF(qm,ql,ep, mu, k/b, maxvel))/b;

% Exact upwinding
[xe,epe] = extend(x,ep,h,1,'N',0,'N',0);
[xe,mue] = extend(x, mu, h, 1, 'N', 0, 'N', 0);
ep0 = epe(2:N+1); epm = epe(1:N); epp = epe(3:N+2);
mu0 = mue(2:N+1); mum = mue(1:N); mup = mue(3:N+2);

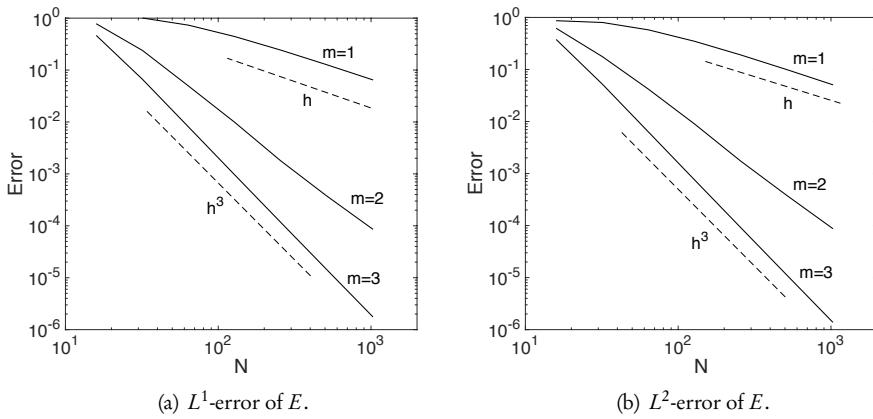
dEH = - (MaxwellUpwind(qr,qp,ep0,epp, mu0, mup, ep0, mu0) - ...
          MaxwellUpwind(qm,ql,epm,ep0,mum, mu0, ep0, mu0))/h;
return

```

As a first example, discussed at length in subsection 1.4.1, we consider a closed metallic cavity with  $\varepsilon_1 = \varepsilon_2 = 1$ . This results in a smooth solution, comprised of simple waves. We use a Lax–Friedrichs numerical flux.

Figure 11.8 shows the error of the computed electric field  $E$  evaluated at final time  $\pi/2$  for decreasing  $h$  and increasing order of the ENO scheme. Since the solution is smooth, we expect the error to behave as  $\mathcal{O}(h^m)$ . This is clearly confirmed by the results.

We now turn to the more interesting case of a cavity with different materials, resulting in a solution with limited regularity. In the case  $\varepsilon_1 \neq \varepsilon_2$ , the magnetic field  $H$  remains smooth but the electric field  $E$  is only continuous, as illustrated in Fig. 1.6.



**Figure 11.8.** The error of the ENO scheme of order  $m$  when solving Maxwell's equations with a smooth solution using  $N$  elements.

While we shall return to the use of a Lax–Friedrichs flux for this case shortly, let us first develop an exact upwinded flux by solving the Riemann problem.

To simplify the discussion, write Maxwell's equations as

$$Q \frac{\partial \mathbf{q}}{\partial t} + A \frac{\partial \mathbf{q}}{\partial x} = 0,$$

where

$$Q = \begin{bmatrix} \varepsilon(x) & 0 \\ 0 & \mu(x) \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} E \\ H \end{bmatrix}$$

represent the spatially varying material coefficients, the one-dimensional rotation operator, and the vector of state variables, respectively.

The flux is given as  $A\mathbf{q}$  and the eigenvalues of  $Q^{-1}A$  are  $\pm(\varepsilon\mu)^{-1/2}$ , which reflect the two light waves, counter-propagating at the local speed of light  $c = (\varepsilon\mu)^{-1/2}$ .

We proceed by using the Riemann conditions to obtain

$$\begin{aligned} c^- Q^- (\mathbf{q}^* - \mathbf{q}^-) + (\Pi \mathbf{q})^* - (\Pi \mathbf{q})^- &= 0 \\ -c^+ Q^+ (\mathbf{q}^* - \mathbf{q}^+) + (\Pi \mathbf{q})^* - (\Pi \mathbf{q})^+ &= 0, \end{aligned}$$

where  $\mathbf{q}^*$  refers to the intermediate state,  $(\Pi \mathbf{q})^*$  is the associated numerical flux, and

$$(\Pi \mathbf{q})^\pm = \hat{\mathbf{n}} \cdot (A\mathbf{q})^\pm = \hat{\mathbf{n}} \cdot \begin{bmatrix} H^\pm \\ E^\pm \end{bmatrix}.$$

Simple manipulations yield

$$(c^+ Q^+ + c^- Q^-)(\Pi \mathbf{q})^* = c^+ Q^+ (\Pi \mathbf{q})^- + c^- Q^- (\Pi \mathbf{q})^+ + c^- c^+ Q^- Q^+ (\mathbf{q}^- - \mathbf{q}^+)$$

or

$$H^* = \frac{1}{\{\{Z\}\}} \left( \{\{ZH\}\} + \frac{1}{2} [\![E]\!] \right), \quad E^* = \frac{1}{\{\{Y\}\}} \left( \{\{YE\}\} + \frac{1}{2} [\![H]\!] \right).$$

Here

$$Z^\pm = \sqrt{\frac{\mu^\pm}{\varepsilon^\pm}} = (Y^\pm)^{-1}$$

represents the impedance of the medium. We have introduced the notation of

$$\{\{u\}\} = \frac{u^- + u^+}{2}, \quad [\![u]\!] = \hat{n}^- u^- + \hat{n}^+ u^+,$$

with  $u^-$  being the left internal state with the corresponding outward pointing normal vector  $\hat{n}^-$ . Similarly,  $u^+$  represents the right external state with the corresponding outward pointing normal vector  $\hat{n}^+$ . In one dimension,  $\hat{n}^- = -\hat{n}^+$ .

Considering the simplest case of a continuous medium, things simplify since

$$H^* = \{\{H\}\} + \frac{Y}{2} [\![E]\!], \quad E^* = \{\{E\}\} + \frac{Z}{2} [\![H]\!].$$

This we recognize as the Lax–Friedrichs flux, since

$$\frac{Y}{\varepsilon} = \frac{Z}{\mu} = \frac{1}{\sqrt{\varepsilon \mu}} = c$$

is the speed of light. `MaxwellUpwind.m` implements the corresponding numerical flux.

**Script 11.12. *MaxwellUpwind.m*: Exactly upwinded numerical flux for Maxwell's equations with discontinuous coefficients.**

---

```
function [ numflux ] = MaxwellUpwind(u, v, epu, evp, muu, muv, ep, mu)
% function [numflux] = MaxwellUpwind(u, v, epu, evp, muu, muv, ep, mu)
% Purpose: Evaluate exact upwind numerical flux for Maxwell's equation with
% discontinuous coefficients

Zm = sqrt(muu./epu); Zp = sqrt(muv./evp); Zavg = (Zm+Zp)./2;
Ym = 1./Zm; Yp = 1./Zp; Yavg = (Ym+Yp)./2;

Hs = ( (Zm.*u(:,2) + Zp.*v(:,2))/2 + (u(:,1)-v(:,1))/2 )./Zavg;
Es = ( (Ym.*u(:,1) + Yp.*v(:,1))/2 + (u(:,2)-v(:,2))/2 )./Yavg;
numflux = [Hs./ep Es./mu];
end
```

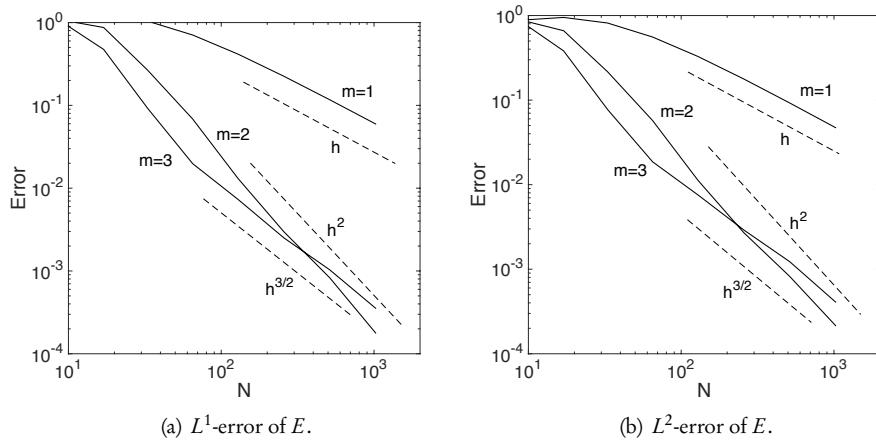
---

We use this numerical flux to solve the case of  $\varepsilon_1 = 1, \varepsilon_2 = 2.25$ . In Fig. 11.9 we show the error of the computed electric field  $E$  evaluated at final time  $\pi/2$  for decreasing  $h$  and for different orders of the ENO scheme. Since the solution is no longer smooth, we expect a reduced order of convergence, as is confirmed. Relying on standard polynomial approximation theory, to be discussed in more detail in Chapter 12, we can conjecture an error estimate as

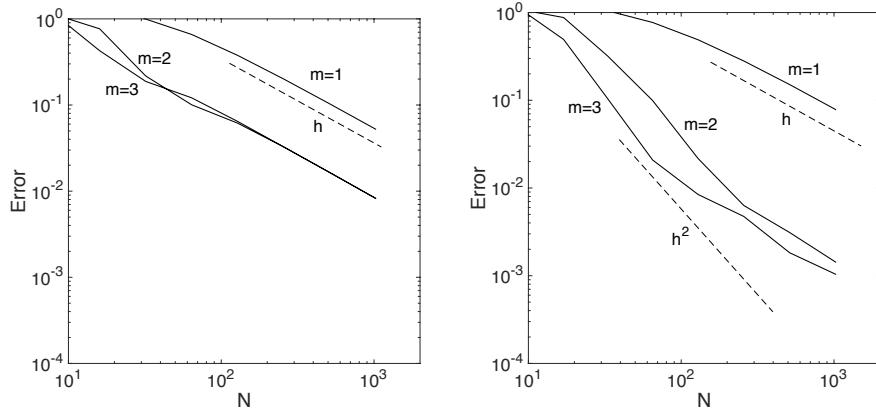
$$\|u - u_h\|_{h,2} \leq Ch^\sigma \|u^{(p)}\|_{h,2},$$

where  $\sigma = \min(m, p + 1/2)$  for  $u^{(p)} \in L^2$ . In this particular case  $u^{(1)} \in L^2$ .

Recovering this result computationally is, however, a delicate matter. Figure 11.10 illustrates the results of two slightly different experiments. In the first case, the point of discontinuity is placed inside a cell, rather than at a cell interface as was the case for the optimal results shown in Fig. 11.9. As a result, the accuracy reduces to first order.



**Figure 11.9.** The error of the ENO scheme of order  $m$  when solving Maxwell's equations with a nonsmooth solution using  $N$  elements.



**Figure 11.10.** The error of the ENO scheme of order  $m$  when solving Maxwell's equations with a nonsmooth solution. On the left we show the error with the point of discontinuity inside a cell, while the right shows the error when using a simple Lax-Friedrichs flux. In both cases we show the  $L^1$ -error of the electric field.

Similarly, if the flux is changed to the Lax-Friedrichs flux, as in the second panel in Fig. 11.10, the global accuracy is also reduced to first order.

### The Euler equations

Finally, let us consider the solution of the one-dimensional Euler equations. We refer to subsection 1.4.1 for a detailed discussion of the problem.

In EulerENODriver1D.m we show the driver for the solver, setting the order of the ENO solver and defining the initial condition.

**Script 11.13. EulerENO1D.m: Driver routine for solving the one-dimensional Euler equations using an ENO scheme.**

---

```
% Driver script for solving the 1D Euler equations using an ENO scheme
clear all

% Order of method
m=2;

% Set problem parameters
L = 1; FinalTime = 0.2; N = 256; CFL = 0.90; gamma = 1.4; h = L/N;

% Define domain, materials and initial conditions
r = zeros(N+1,1); ru = zeros(N+1,1); E = zeros(N+1,1);

% Initialize for Sod's problem - piecewise constant so no integration
x = [0:h:1]';
for i=1:N+1;
    if x(i)<0.5
        r(i)=1.0; E(i) = 1/(gamma-1);
    else
        r(i) = 0.125; E(i) = 0.1/(gamma-1);
    end
end

% Initialize for shock entropy problem
% x = [-5:h:5]';
% NGQ = 10; [xGQ,wGQ] = LegendreGQ(NGQ);
% for i=1:N+1;
%     if x(i)<-4
%         rh = 3.857143; u = 2.629369; p = 10.33333;
%     else
%         rh = 0;
%         for j=1:NGQ+1
%             rh = rh + wGQ(j)*(1+0.2*sin(pi*(x(i)+h/2*xGQ(j))));%
%         end
%         rh = rh/2; u = 0; p = 1;
%     end
%     r(i) = rh; ru(i) = rh*u; E(i) = p/(gamma-1)+0.5*rh*u^2;
% end

% Solve Problem
q = [r ru E];
[q] = EulerENO1D(x,q,h,m,CFL,gamma,FinalTime);
```

---

In EulerENO1D.m the coefficients  $c_{r,j}$  are precomputed, and a third order SSPERK method is used to advance the Euler equations in time.

**Script 11.14. EulerENO1D.m: Time integrator for solving the one-dimensional Euler equations using an ENO scheme**

---

```
function [q] = EulerENO1D(x,q,h,m,CFL,gamma,FinalTime)
% function [q] = EulerMDriver1D(x,q,h,m,CFL,gamma,FinalTime)
% Purpose : Integrate 1D Euler equation until FinalTime using an ENO
% scheme and a 3rd order SSP-RK
time = 0; tstep = 0;

% Initialize reconstruction weights
Crec = zeros(m+1,m);
for r=-1:m-1;
    Crec(r+2,:) = ReconstructWeights(m,r);
```

```

end;

% integrate scheme
while (time<FinalTime)

    % Set timestep
    p = (gamma-1)*(q(:,3) - 0.5*q(:,2).^2./q(:,1)); c = sqrt(gamma*p./q(:,1));
    maxvel = max(c+abs(q(:,2)./q(:,1))); k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    rhsq = EulerENOrhs1D(x, q, h, k, m, Crec, gamma, maxvel);
    q1 = q + k*rhsq;
    rhsq = EulerENOrhs1D(x, q1, h, k, m, Crec, gamma, maxvel);
    q2 = (3*q + q1 + k*rhsq)/4;
    rhsq = EulerENOrhs1D(x, q2, h, k, m, Crec, gamma, maxvel);
    q = (q + 2*q2 + 2*k*rhsq)/3;
    time = time+k; tstep = tstep+1;
end
return

```

Finally, EulerENOrhs1D.m implements the ENO scheme for the Euler equations with the reconstruction done on the conserved variables. The current solution is extended with  $m - 1$  points at each end to allow the implementation of boundary conditions, as also used for the monotone scheme. For each node  $x_j$ , the  $2m - 1$  neighboring points are extracted and the ENO reconstruction computes the approximation to the solution at the cell interfaces,  $x_{j\pm 1/2}$ . Finally, these are used to evaluate the numerical flux in the conservative scheme. We shall primarily use a global Lax–Friedrichs flux, but also consider alternatives.

---

**Script 11.15. EulerENOrhs1D.m: Evaluation of the right-hand side for the one-dimensional Euler equations using an ENO scheme.**

---

```

function [dq] = EulerENOrhs1D(x, q, h, k, m, Crec, gamma, maxvel)
% function [dq] = EulerENOrhs1D(x, q, h, k, m, Crec, gamma, maxvel)
% Purpose: Evaluate right hand side for Euler equations using an ENO method
N = length(x); dq = zeros(N,3);
q1 = zeros(N,3); qr = zeros(N,3); qp = zeros(N,3); qm = zeros(N,3);

% Extend data and assign boundary conditions for Sod's problem
[xe, re] = extend(x, q(:,1), h, m, 'D', 1.0, 'D', 0.125);
[xe, me] = extend(x, q(:,2), h, m, 'D', 0, 'N', 0);
[xe, Ee] = extend(x, q(:,3), h, m, 'D', 2.5, 'N', 0);

% Extend data and assign boundary conditions for shock entropy
% [xe, re] = extend(x, q(:,1), h, m, 'D', 3.857143, 'N', 0);
% [xe, me] = extend(x, q(:,2), h, m, 'D', 10.141852, 'D', 0);
% [xe, Ee] = extend(x, q(:,3), h, m, 'D', 39.166661, 'N', 0);

% define cell left and right interface values
rm = zeros(N+2,1); mm = zeros(N+2,1); Em = zeros(N+2,1);
rp = zeros(N+2,1); mp = zeros(N+2,1); Ep = zeros(N+2,1);

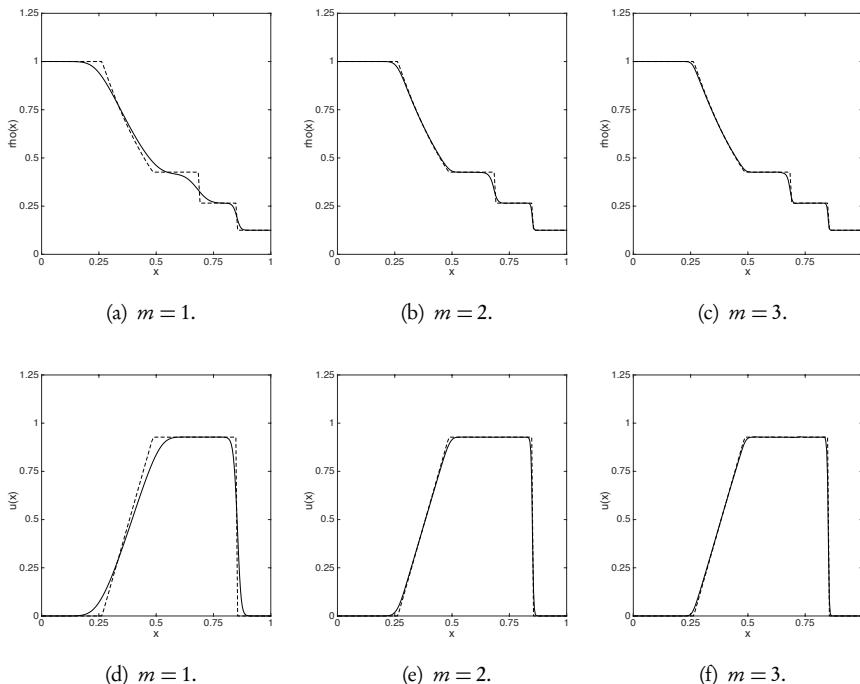
for i=1:N+2
    [rm(i), rp(i)] = ENO(xe(i:(i+2*(m-1))), re(i:(i+2*(m-1))), m, Crec);
    [mm(i), mp(i)] = ENO(xe(i:(i+2*(m-1))), me(i:(i+2*(m-1))), m, Crec);
    [Em(i), Ep(i)] = ENO(xe(i:(i+2*(m-1))), Ee(i:(i+2*(m-1))), m, Crec);
end;

```

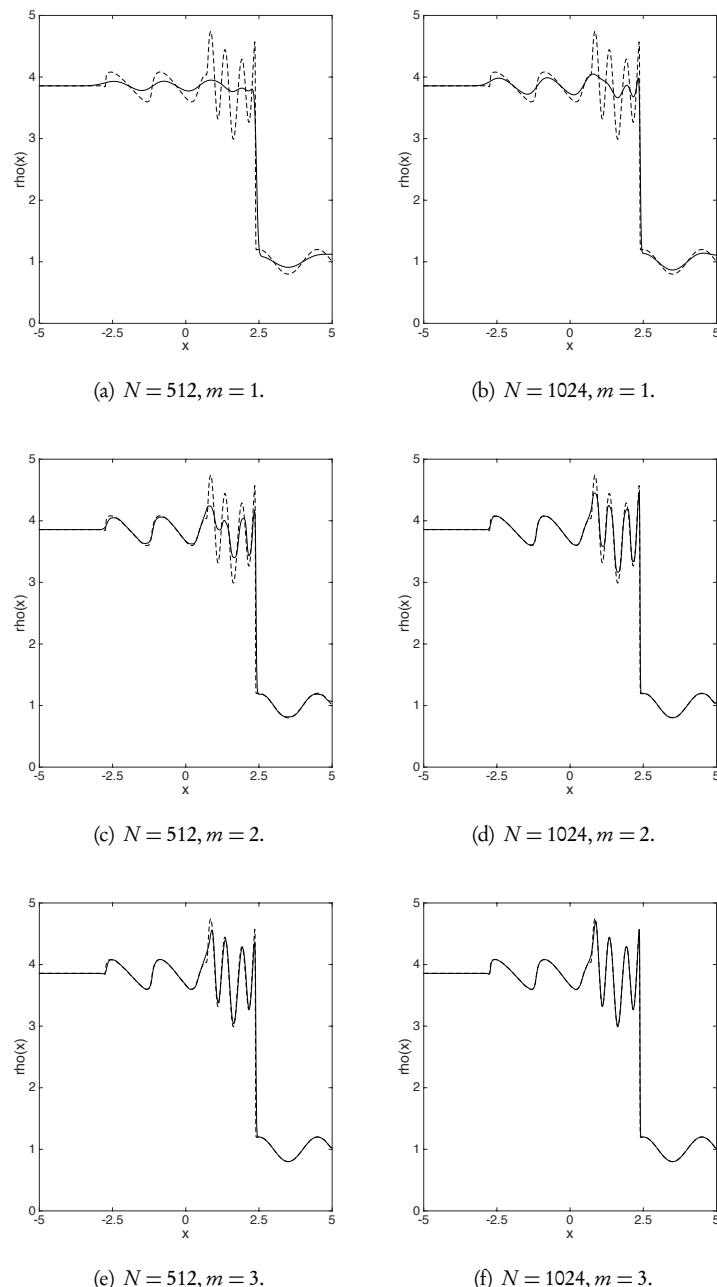
```
% Compute rhs - also change numerical flux here
ql = [rm(2:N+1) mm(2:N+1) Em(2:N+1)]; qr = [rp(2:N+1) mp(2:N+1) Ep(2:N+1)];
qp = [rm(3:N+2) mm(3:N+2) Em(3:N+2)]; qm = [rp(1:N) mp(1:N) Ep(1:N)];
dq = - (EulerLF(qr, qp, gamma, k/h, maxvel)) - ...
      EulerLF(qm, ql, gamma, k/h, maxvel))/h;
return
```

As a first test case, we revisit Sod's shock tube problem. Its solution for different orders of accuracy is shown in Fig. 11.11. We observe a dramatic improvement in the overall accuracy of the solution with increasing order, in particular in the resolution of the rarefaction wave and, most pronounced, for the contact discontinuity located around  $x = 0.7$ . Since the contact discontinuity is essentially a linear wave, this agrees well with the discussion in section 8.1 on the benefits of high-order methods for linear wave problems. However, we also observe a decreased smearing of the shock as a result of the increased order of accuracy. Comparing visually with the results in Fig. 5.5, it is arguable that the results obtained with  $N = 8192$  and a first order monotone scheme are inferior to the results in Fig. 11.11 for  $N = 256$  and  $m = 3$ . This serves as a powerful illustration of the substantial potential for using high-order accurate methods, even for problems with shocks.

Sod's problem is, in many ways, well suited for a first order method since its solution is predominately piecewise constant. Let us therefore also consider the shock-entropy wave problem discussed in subsection 1.4.1. Figure 11.12 shows the results obtained for



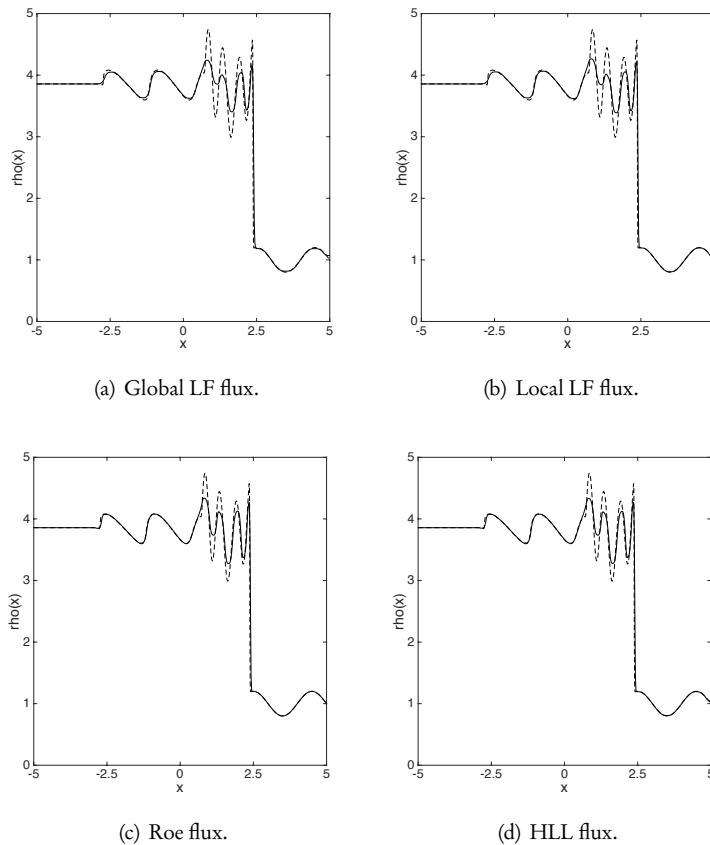
**Figure 11.11.** Computed solution to Sod's shock tube problem at  $T = 0.2$  solved with  $N = 256$  cells and different orders  $m$  of the ENO scheme. The top row shows the density  $\rho$  while the bottom row shows the velocity  $u$ . The solid line refers to the computed solution, while the dashed line reflects the exact solution.



**Figure 11.12.** The computed density for the shock entropy problem at  $T = 1.8$  solved with an  $m$ th order ENO scheme, using  $N$  cells. The solid lines represent the computed solution, while the dashed line reflects the reference solution.

different values of  $N$  and  $m$ . Again, we observe a dramatic improvement in the quality of the solution when increasing the order of the scheme. This is particularly evident in the high-frequency oscillations upstream of the shock in the reference solution. These oscillations are effectively destroyed by the dissipation of the first order scheme, while increasing the order of the scheme quickly recovers the oscillations and only minor details of the solution are left unresolved. Comparing the results obtained with a monotone scheme, shown in Fig. 5.7, we observe that  $N = 1024$  and  $m = 3$  is, qualitatively, at least as good as the one obtained with  $N = 32768$  and a Lax–Friedrichs scheme, and it remains superior to the results in Fig. 6.5, which rely on the use of more advanced numerical fluxes.

Let us finally consider the question of the impact of different choices of the flux. In Fig. 6.5, we observed that the use of more advanced fluxes, albeit at added cost, can improve the overall accuracy of the computed results for monotone schemes. In Fig. 11.13 we present a similar comparison for the shock entropy problem, solved using a second order ENO scheme and  $N = 512$  cells. While the Lax–Friedrichs flux shows signs of dissipation, the remaining fluxes produce results that are visually similar.



**Figure 11.13.** The computed density for the shock entropy problem at  $T = 1.8$  solved with  $N = 512$ , a second order ENO scheme and different choices of the numerical flux. The solid lines represent the computed solution, while the dashed line reflects the reference solution.

This suggests that, for higher-order schemes, the choice of the flux is less critical and, consequently, can often be made to minimize the overall computational cost.

### 11.3 • WENO methods

While the success of the ENO methods is undisputed, it is easy to identify a number of shortcomings:

- Although  $m$  different stencils are evaluated, only one is finally used.
- In smooth regions, the union of the  $m$  stencils covers a total of  $2m - 1$  points, yet the scheme attains  $\mathcal{O}(h^m)$  accuracy only.
- The nonlinear nature of the stencil selection suggests that even small changes in the solution can result in a different stencil choice. Hence, even for smooth problems, the error behavior can be noisy and uneven.

Let us attempt to gain some insight into how to overcome these shortcomings. If we first assume that the solution is smooth we may, as discussed in section 11.1, seek coefficients  $d_r^{r_0}$  such that the linear combination of the  $m$  stencils

$$v_{j+1/2} = \sum_{r=r_0}^{m-1+r_0} d_r^{r_0} v_{j+1/2}^{(r)},$$

is  $\mathcal{O}(h^{2m-1})$  accurate. However, this approach reduces the scheme to a linear scheme and, thus, we lose accuracy at points of limited regularity.

Let us therefore consider a slightly more general formulation,

$$v_{j+1/2} = \sum_{r=r_0}^{m-1+r_0} \omega_r^{r_0} v_{j+1/2}^{(r)},$$

and seek to define the weights  $\omega_r^{r_0}$  so that a linear scheme is recovered in a smooth region of the solution, while the ENO scheme is retained close to points of discontinuity.

For consistency and stability, we require that

$$\sum_{r=r_0}^{m-1+r_0} \omega_r^{r_0} = 1, \quad \omega_r^{r_0} \geq 0.$$

Furthermore, to maintain optimal linear accuracy in smooth parts of the solution, we must require that

$$\omega_r^{r_0} = d_r^{r_0} + \mathcal{O}(h^{m-1}), \quad (11.7)$$

since

$$\sum_{r=r_0}^{m-1+r_0} (\omega_r^{r_0} - d_r^{r_0}) v_{j+1/2}^{(r)} = \sum_{r=r_0}^{m-1+r_0} (\omega_r^{r_0} - d_r^{r_0}) [v_{j+1/2}^{(r)} - v(x_{j+1/2})] = \mathcal{O}(h^{2m-1}).$$

Here we have used the consistency relations

$$\sum_{r=r_0}^{m-1+r_0} \omega_r^{r_0} = \sum_{r=r_0}^{m-1+r_0} d_r^{r_0} = 1.$$

Clearly, the path to success lies in a careful definition of  $\omega_r^{r_0}$ .

Following [17], let us consider the following choice

$$\omega_r^{r_0} = \frac{\alpha_r}{\sum_{s=0}^{m-1} \alpha_s}, \quad \alpha_r = \frac{d_r^{r_0}}{(\varepsilon + \beta_r)^{2p}}, \quad (11.8)$$

where  $\varepsilon \ll 1$ ; typically  $\varepsilon = 10^{-6}$ . The appropriate value of  $\varepsilon$  is closely related to the magnitude of  $\beta_r$  and, for very accurate solutions, a smaller value may be needed. For a discussion of this matter we refer to [16, 1]. The value of  $p$  can be changed to modify the impact of the smoothness indicator, although a value of  $p = 1$  is often used.

We first consider the smooth case. In this case, (11.7) is satisfied provided

$$\beta_r = C(h)(1 + \mathcal{O}(h^{m-1})).$$

On the other hand, if  $\beta_r = \mathcal{O}(1)$ , we recover  $\omega_r^{r_0} = \mathcal{O}(h^{2p})$ , which severely reduces the impact of this stencil. This is precisely the behavior we are looking for.

The final step in the design of the algorithm is the definition of the smoothness indicator  $\beta_r$ . Intuitively, we seek a cheap and smooth indicator related to the total variation of the solution. To avoid the complexity of evaluating the  $L^1$ -norm, [17] proposes

$$\beta_r = \sum_{l=1}^{m-1} \int_{x_{j-1/2}}^{x_{j+1/2}} b^{2l-1} \left( \frac{d^l \pi_r}{dx^l} \right)^2 dx. \quad (11.9)$$

We recall that  $\pi_r$  is the reconstruction polynomial for the specific stencil, while the factor of  $b^{2l-1}$  ensures grid-scale independence. Since we assume an equidistant grid, the specific reference to  $r_0$  is omitted.

The smoothness indicators can be expressed in explicit form as

$$\beta_0 = (\bar{v}_{j+1} - \bar{v}_j)^2, \quad \beta_1 = (\bar{v}_j - \bar{v}_{j-1})^2,$$

for  $m = 2$ , and

$$\beta_0 = \frac{13}{12} (\bar{v}_j - 2\bar{v}_{j+1} + \bar{v}_{j+2})^2 + \frac{1}{4} (3\bar{v}_j - 4\bar{v}_{j+1} + \bar{v}_{j+2})^2, \quad (11.10)$$

$$\beta_1 = \frac{13}{12} (\bar{v}_{j-1} - 2\bar{v}_j + \bar{v}_{j+1})^2 + \frac{1}{4} (\bar{v}_{j-1} - \bar{v}_{j+1})^2,$$

$$\beta_2 = \frac{13}{12} (\bar{v}_{j-2} - 2\bar{v}_{j-1} + \bar{v}_j)^2 + \frac{1}{4} (\bar{v}_{j-2} - 4\bar{v}_{j-1} + 3\bar{v}_j)^2,$$

for  $m = 3$ . For a smooth function, one easily verifies that

$$\beta_r = Ch^2 (1 + \mathcal{O}(h^{m-1})), \quad (11.11)$$

which ensures that (11.7) holds. The actual computation of the smoothness indicators for general values of  $m$  is somewhat involved, as outlined in the following example.

**Example 11.5.** The evaluation of the smoothness indicator  $\beta_r$  for a general order is complicated by the need to evaluate the integral of the square of the reconstruction polynomial  $\pi_r$ .

Let us consider the smoothness indicator

$$\beta_r = \sum_{l=1}^{m-1} \int_{x_{j-1/2}}^{x_{j+1/2}} b^{2l-1} \left( \frac{d^l \pi_r}{dx^l} \right)^2 dx,$$

where the reconstruction polynomial  $\pi_r$ , embedding the dependence on the grid and the local solution, is defined as

$$\pi_r(x) = b \sum_{q=0}^{m-1} \bar{v}_{q+j-r} \left( \sum_{i=q+1}^m (\ell_i^{(j-r-1/2)})'(x) \right).$$

We recall that  $\ell_i^{(j-r-1/2)}(x)$  is the Lagrange polynomial, defined in (11.3). As a first step, we need to evaluate

$$\frac{d^l \pi}{dx^l} = b \sum_{q=0}^{m-1} \bar{v}_{q+j-r} \left( \sum_{i=q+1}^m (\ell_i^{(j-r-1/2)})^{(l+1)}(x) \right) = b \sum_{q=0}^{m-1} \bar{v}_{q+j-r} Q_{q,r}^l(x).$$

We proceed to express  $Q_{q,r}^l(x)$  by its Taylor expansion by first considering

$$\ell_i(x) = \sum_{k=0}^m \frac{c_{k,i}}{k!} x^k.$$

As discussed in [11], the coefficients  $c_{k,i}$  can be found through a forward recurrence for any order  $m$  and distribution of grid points  $x_j$ . The algorithm is illustrated in `lagrangeweights.m`

**Script 11.16. *lagrangeweights.m*: Weights for the Taylor expansion of the Lagrange polynomial.**

---

```

function [cw] = lagrangeweights(x)
% Purpose: Compute weights for Taylor expansion of Lagrange polynomial based
%           on x and evaluated at 0.
%           Method due to Fornberg (SIAM Review, 1998, 685–691)
np = length(x); cw=zeros(np,np);
cw(1,1)=1.0; c1 = 1.0; c4 = x(1);
for i=2:np
    mn = min(i,np-1)+1;
    c2 = 1.0; c5 = c4; c4 = x(i);
    for j=1:i-1
        c3 = x(i)-x(j); c2 = c2*c3;
        if (j==i-1)
            for k=mn:-1:2
                cw(i,k) = c1*((k-1)*cw(i-1,k-1)-c5*cw(i-1,k))/c2;
            end
            cw(i,1) = -c1*c5*cw(i-1,1)/c2;
        end
    end
end

```

---

```

for k=mn:-1:2
    cw(j,k) = (c4*cw(j,k)-(k-1)*cw(j,k-1))/c3;
end
cw(j,1) = c4*cw(j,1)/c3;
end
c1=c2;
end
return

```

---

The derivative follows immediately as

$$\frac{d^l}{dx^l} \ell_i(x) = \sum_{k=0}^{m-l} \frac{c_{k+l,i}}{k!} x^k,$$

from which we recover

$$Q_{q,r}^l(x) = \sum_{i=q+1}^m \left( \sum_{k=0}^{m-l} \frac{c_{k+l,i}}{k!} x^k \right) = \sum_{k=0}^{m-l} \frac{d_{k,q}^l}{k!} x^k, \quad d_{k,q}^l = \sum_{i=q+1}^m c_{k+l,i}. \quad (11.12)$$

Evaluating the square of the derivatives of the reconstruction polynomial amounts to

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \left( \frac{d^l}{dx^l} \pi \right)^2 dx = \int_{x_{j-1/2}}^{x_{j+1/2}} \left( \sum_{q=0}^{m-1} \bar{v}_{q+j-r} Q_{q,r}^l(x) \right)^2 dx = \bar{v}^T \tilde{Q}^l \bar{v},$$

where the  $m \times m$  matrix operator  $\tilde{Q}^l(x)$  has the entries

$$\tilde{Q}_{ij}^l = \int_{x_{j-1/2}}^{x_{j+1/2}} Q_{i,r}^l(x) Q_{j,r}^l(x) dx.$$

Recalling (11.12), we evaluate this by considering

$$Q_{i,r}^l(x) Q_{j,r}^l(x) = \tilde{x}^T D \tilde{x},$$

with

$$\tilde{x} = [1, x, x^2/2, \dots, x^{m-l}/(m-l)!]^T, \quad D_{ij} = d_{k,i}^l d_{k,j}^l,$$

to recover

$$\tilde{Q}_{ij}^l = \int_{x_{j-1/2}}^{x_{j+1/2}} \tilde{x}^T D \tilde{x} dx.$$

Since this involves a polynomial expression of order  $2(m-l)$ , it can be integrated exactly by a Gaussian quadrature, as illustrated in *Qcalc.m*.

---

**Script 11.17. *Qcalc.m*: Evaluation of entries to matrix operator  $\tilde{Q}^l$  needed for smoothness indicator.**

---

```

function [Qelem] = Qcalc(D,m,l);
% Purpose: Evaluate entries in the smoothness indicator for WENO
[x,w] = LegendreGQ(m); xq = x./2; Qelem = 0;
for i=1:m+1

```

---

```

xvec = zeros(m-1+1,1);
for k=0:m-1 xvec(k+1) = xq(i)^k./prod(1:k); end;
Qelem = Qelem + (xvec'*D*xvec)*w(i)/2;
end
return

```

---

Once these operators are computed for all values of  $l$ , the smoothness indicator can be evaluated directly. Assuming equidistant grids with spacing  $h$ , we recover

$$\beta_r = \bar{v}^T \left( \sum_{l=1}^{m-1} \tilde{Q}^l \right) \bar{v}.$$

The entire routine is shown in `betarcalc.m`.

**Script 11.18. *betarcalc.m*: Computation of matrix operator to evaluate smoothness indicator.**

---

```

function [errmat] = betarcalc(x,m)
% Purpose: Compute matrix to allow evaluation of smoothness indicator in
%           WENO based on stencil [x] of length m+1.
%           Returns sum of operators for l=1..m-1

% Evaluate Lagrange polynomials
[cw] = lagrangeweights(x);

% Compute error matrix for l=1..m-1
errmat = zeros(m,m);
for l=2:m
    % Evaluate coefficients for derivative of Lagrange polynomial
    dw = zeros(m,m-1+1);
    for k=0:(m-1)
        for q=0:m-1
            dw(q+1,k+1) = sum(cw((q+2):m+1,k+l+1));
        end
    end

    % Evaluate entries in matrix for order 'l'
    Qmat = zeros(m,m);
    for p=0:m-1
        for q=0:m-1
            D = dw(q+1,:)'*dw(p+1,:);
            Qmat(p+1,q+1) = Qcalc(D,m,l);
        end
    end
    errmat = errmat + Qmat;
end
return

```

---

To exemplify the evaluation of the smoothness indicator, let us first consider the case of  $m = 2$  and  $r = 0, 1$ , characterized by the grid  $[x_{-1/2-r}, x_{1/2-r}, x_{3/2-r}]$ . The resulting operator is

$$Q^0 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad Q^1 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

and the smoothness indicators for  $m = 2$  are given as

$$\beta_0 = (\bar{v}_{j+1} - \bar{v}_j)^2, \quad \beta_1 = (\bar{v}_j - \bar{v}_{j-1})^2.$$

For  $m = 3$  and  $r = 0, 1, 2$ , we obtain

$$Q^0 = \frac{1}{6} \begin{bmatrix} 20 & -31 & 11 \\ -31 & 50 & -19 \\ 11 & -19 & 8 \end{bmatrix}, \quad Q^1 = \frac{1}{6} \begin{bmatrix} 8 & -13 & 5 \\ -13 & 26 & -13 \\ 5 & -13 & 8 \end{bmatrix},$$

$$Q^2 = \frac{1}{6} \begin{bmatrix} 8 & -19 & 11 \\ -19 & 50 & -31 \\ 11 & -31 & 20 \end{bmatrix}.$$

We note the close relationship between  $Q^0$  and  $Q^2$ . Note also that row and column sums are zero, as is expected by consistency, since these operators are associated with the derivative of a polynomial.

In agreement with (11.10), we can rewrite these smoothness indicators as

$$\beta_0 = \frac{13}{12} (\bar{v}_j - 2\bar{v}_{j+1} + \bar{v}_{j+2})^2 + \frac{1}{4} (3\bar{v}_j - 4\bar{v}_{j+1} + \bar{v}_{j+2})^2,$$

$$\beta_1 = \frac{13}{12} (\bar{v}_{j-1} - 2\bar{v}_j + \bar{v}_{j+1})^2 + \frac{1}{4} (\bar{v}_{j-1} - \bar{v}_{j+1})^2,$$

$$\beta_2 = \frac{13}{12} (\bar{v}_{j-2} - 2\bar{v}_{j-1} + \bar{v}_j)^2 + \frac{1}{4} (\bar{v}_{j-2} - 4\bar{v}_{j-1} + 3\bar{v}_j)^2. \quad \blacksquare$$

While the accuracy of the WENO scheme is designed to be  $\mathcal{O}(h^{2m-1})$  in smooth regions of the solution and its accuracy reduces to  $\mathcal{O}(h^m)$  near discontinuities, one can question if an intermediate accuracy is possible, i.e., whether the nonlinear weights take values that improve the accuracy in the neighborhood of a discontinuity.

To gain some insight into this, let us express the nonlinear weights (11.8) as

$$\omega_r^{r_0} = d_r^{r_0} \left( \sum_{s=0}^{m-1} d_s^{r_0} \left( \frac{\beta_r + \varepsilon}{\beta_s + \varepsilon} \right)^{2p} \right)^{-1}.$$

If all stencils are smooth, then (11.11) immediately yields the linear weights and the optimal order of accuracy is recovered.

Let us instead assume that one stencil, e.g.,  $r = 0$ , is nonsmooth. In this case,  $\beta_0 = \mathcal{O}(1)$  and we observe that  $\omega_0^{r_0} = \mathcal{O}(h^{2p})$  while

$$\omega_r^{r_0} = \frac{d_r^{r_0}}{\sum_{s=1}^{m-1} d_s^{r_0}},$$

for  $r \neq 0$ . Hence, the  $m-2$  stencils are combined linearly away from the nonsmooth stencil but with weights that are different from the optimal linear weights. Hence, the WENO scheme will switch abruptly in accuracy from  $\mathcal{O}(h^{2m-1})$  in smooth regions of the solution to  $\mathcal{O}(h^m)$  in areas where at least one stencil encounters a discontinuity.

### 11.3.1 • WENO variants

The above development leads to the classic WENO scheme [17]. However, since the choice of the nonlinear weights and the smoothness indicator is nonunique, several

alternatives have been proposed. Most efforts focus on improving the accuracy of the WENO methods at critical points, i.e., points where the derivatives of the solution vanish and, due to its widespread use, most work focuses on the fifth order WENO method, i.e.,  $m = 3$ . However, the central ideas of these developments are general and can be extended to higher-order WENO methods.

To realize the potential for an accuracy reduction at a critical point, let us consider the situation at a local extrema. We recall that optimal accuracy of  $\mathcal{O}(h^{2m-1})$  requires

$$\omega_r^{r_0} = d_r^{r_0} + \mathcal{O}(h^{m-1}),$$

which is guaranteed provided

$$\beta_r = C(h)(1 + \mathcal{O}(h^{m-1})). \quad (11.13)$$

This result, based on Taylor expansions, assumes that all derivatives are nonzero.

If we now restrict the attention to  $m = 3$ , an expansion of the smoothness indicators (11.10) yields

$$\beta_r = (v'(x_j)h)^2(1 + \mathcal{O}(h^2)),$$

which establishes the required accuracy. However, if we consider a neighborhood of a critical point where  $v'(x_j) \ll 1$ , we recover

$$\beta_0 = \beta_2 = \frac{13}{12}(v''(x_j)h^2)^2(1 + \mathcal{O}(h)), \quad \beta_1 = \frac{13}{12}(v''(x_j)h^2)^2(1 + \mathcal{O}(h^2)).$$

Thus, the accuracy condition is violated for  $(\beta_0, \beta_2)$ . This could lead to a local reduction of accuracy to  $\mathcal{O}(h^3)$  as the nonlinear weights approximate the linear weights as

$$\omega_r^{r_0} = d_r^{r_0} + \mathcal{O}(h).$$

A natural solution to this problem is to seek a different definition of the nonlinear weights or, as a minimum, modify the weights in the neighborhood of critical points to recover values closer to the ideal linear weights.

In [16], it is proposed to consider the mapping

$$g_r(\omega) = \omega \frac{\omega^2 - 3\omega d_r^{r_0} + d_r^{r_0} + (d_r^{r_0})^2}{(d_r^{r_0})^2 + \omega(1 - 2d_r^{r_0})},$$

where  $\omega \in [0, 1]$ . The new weights,  $\omega_r^M$ , are then obtained as

$$\alpha_r^* = g_r(\omega_r^{r_0}), \quad \omega_r^M = \frac{\alpha_r^*}{\alpha_0^* + \alpha_1^* + \alpha_2^*}.$$

The motivation behind the construction of  $g_r(\omega)$  is that weights in the neighborhood of a critical point are mapped closer to the optimal linear value. Indeed, one shows by a Taylor expansion that

$$\alpha_r^* = d_r^{r_0} + \mathcal{O}(h^3),$$

so that

$$\omega_r^M = d_r^{r_0} + \mathcal{O}(h^3),$$

which ensures that optimal accuracy is recovered everywhere, including at critical points. Note, however, that if also the second derivative vanishes at the critical point, the accuracy of this mapped approach reduces to second order, as does the classic WENO method.

The implementation of these new mapped weights is straightforward. However, since the mapping has to be evaluated for all weights in every cell, the cost is considerable. This observation has motivated the development of a different approach to ensure full accuracy at critical points, leading to WENO-Z methods, introduced in [1, 5].

Let us define a new parameter

$$\tau = |\beta_0 - \beta_2|,$$

intended to provide a global measure of the smoothness of the solution across the stencil. Using a Taylor expansion it is easy to show that

$$\tau = \frac{13}{3} |v''(x_j)v'''(x_j)|h^5 + \mathcal{O}(h^6),$$

for a smooth solution. This allows the definition of new nonlinear weights as

$$\omega_r^z = \frac{\alpha_r^z}{\alpha_0^z + \alpha_1^z + \alpha_2^z}, \quad \alpha_r^z = d_r^{r_0} \left( 1 + \left( \frac{\tau}{\beta_r + \varepsilon} \right)^q \right). \quad (11.14)$$

If we assume that  $\beta_r = \mathcal{O}(h^2)$  in smooth parts of the solution, we recover

$$\omega_r^z = d_r^{r_0} + \mathcal{O}(h^{3q}),$$

hence ensuring optimal formal accuracy. In [1], it is shown that for  $q = 1$ , (11.14) ensures fourth order accuracy at critical points with a vanishing first derivative, while  $q = 2$  ensures recovery of optimal accuracy at such points. Furthermore, at least second order accuracy is obtained at all higher-order critical points. It is noteworthy that these modified nonlinear weights can be applied at minimal additional cost as compared to the classic WENO method, i.e., there is no clear argument in favor of the classic WENO method over the more accurate WENO-Z method.

In [5], this approach has been generalized to include any order of the WENO-Z scheme through the nonlinear weights

$$\omega_r^z = \frac{\alpha_r^z}{\sum_{s=0}^{m-1} \alpha_s^z}, \quad \alpha_r^z = d_r^{r_0} \left( 1 + \left( \frac{\tau_{2m-1}}{\beta_r + \varepsilon} \right)^q \right),$$

where ( $m > 2$ ):

$$\tau_{2m-1} = \begin{cases} |\beta_0 - \beta_{m-1}|, & m \text{ odd}, \\ |\beta_0 - \beta_1 - \beta_{m-2} + \beta_{m-1}|, & m \text{ even}. \end{cases}$$

Typically, one takes  $q = m - 1$ , although this is not required. Through a careful analysis, this definition of weights has been shown to ensure optimal accuracy, even at

higher-order critical points and for WENO-Z schemes up to 11th order ( $m = 5$ ). We refer to [5] for the details.

At the end of the previous section, we observed that the WENO scheme switches abruptly in accuracy from  $\mathcal{O}(h^{2m-1})$  to  $\mathcal{O}(h^m)$  as soon as one stencil encounters a discontinuity. In the case of a large stencil in which a discontinuity is encountered at the edges, one can question if it is possible to combine the available smooth stencils to enhance the local accuracy, i.e., seek to change some of the weights in the reconstruction in parts of the stencil to enhance the overall accuracy. Since we already realized that the classic WENO weights do not allow this, a modification of the weights is needed.

Such schemes, known as embedded WENO schemes, have recently been proposed. The basic idea is to redefine the nonlinear weights such that smooth stencils are combined in a hierarchical manner to yield an order of accuracy between  $\mathcal{O}(h^m)$  and  $\mathcal{O}(h^{2m-1})$ , when possible. This approach is similar in spirit to the WENO method, albeit now applied only on subsets of the stencils in the reconstruction.

In [40] this is achieved by redefining the nonlinear weights as

$$\tilde{\omega}_r^{r_0} = \omega_r^{r_0} \left( a_{rr} + \sum_{\substack{s=0 \\ s \neq r}}^{m-1} \frac{a_{rs} \beta_s}{\beta_r + \varepsilon} \right), \quad (11.15)$$

where the coefficients  $a_{ij}$  need to be specified. By consistency, one recovers that

$$\sum_{s=0}^{m-1} a_{rs} = 1, \quad k = 0, \dots, m-1.$$

The motivation for the modification in (11.15) is found in the observation that

$$\frac{\beta_r}{\beta_s} = \begin{cases} 1 + \mathcal{O}(h^{m-1}) & \text{if } \beta_r \sim \beta_s \sim C(h)(1 + \mathcal{O}(h^{m-1})), \\ C(h) & \text{if } \beta_r \sim C(h)(1 + \mathcal{O}(h^{m-1})), \beta_s \sim \mathcal{O}(1), \\ (C(h))^{-1} & \text{if } \beta_r \sim \mathcal{O}(1), \beta_s \sim C(h)(1 + \mathcal{O}(h^{m-1})), \end{cases}$$

by (11.13). We recall that generally  $C(h) \sim Ch^{m-1}$ . Hence, for smooth stencils, the maximum order of accuracy is maintained, whereas in the neighborhood of discontinuities where  $\beta_r \sim \mathcal{O}(1)$ , the modifications can be substantial. The remaining constants can then be determined, depending on the stencil in which the discontinuity is found, and with the goal of increasing the linear accuracy in the smooth stencils. With such a scheme one expects improved accuracy for smooth structures in the neighborhood of discontinuities, albeit at increased computational cost for the nonlinear weights. We refer to [40] for details and computational results that confirm these initial considerations.

Let us finally discuss central WENO schemes (CWENO), introduced in [19] and developed further in [20, 3, 4]. We consider a central stencil of size  $2m-1$  in the neighborhood of  $x_j$  and  $m$  substencils of size  $m$ , similar to that of a standard WENO technique. A global polynomial of order  $2m-2$ , interpolating the  $2m-1$  cell averages, is given as

$$\frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} \pi_G(x) dx = \bar{u}_i \quad \forall i = j-(m-1), \dots, j+(m-1).$$

Similarly, we define polynomials based on the  $m$  stencils of  $m$  points as

$$\frac{1}{b} \int_{x_{i-1/2}}^{x_{i+1/2}} \pi_r(x) dx = \bar{u}_i \quad \forall i = j - (m-1) + (r-1), \dots, j + (r-1),$$

for  $r = 1 \dots m$ . The construction of these two sets of polynomials follows the discussion in section 11.1. Let us now define another global polynomial:

$$\pi_0(x) = \frac{1}{d_0} \left( \pi_G(x) - \sum_{r=1}^m d_r \pi_r(x) \right),$$

where  $d_r \in [0, 1]$  represents a set of linear coefficients. Consistency requires that

$$\sum_{r=0}^m d_r = 1.$$

While  $\pi_0$  is a polynomial of order  $2m-1$ , the accuracy of the approximation is only  $m$ . To see this, consider

$$\begin{aligned} u(x) - \pi_0(x) &= \frac{1}{d_0} \left[ d_0 u(x) - \pi_G(x) + \sum_{r=1}^m d_r \pi_r(x) \right] \\ &= \frac{1}{d_0} \left[ \left( 1 - \sum_{r=1}^m d_r \right) u(x) - \pi_G(x) + \sum_{r=1}^m d_r \pi_r(x) \right] \\ &= \frac{1}{d_0} [u(x) - \pi_G(x)] - \sum_{r=1}^m d_r [u(x) - \pi_r(x)] \\ &= \mathcal{O}(h^{2m-1}) + \mathcal{O}(h^m). \end{aligned}$$

We now follow the development of the WENO method and define

$$\omega_r = \frac{\alpha_r}{\sum_{s=0}^m \alpha_s}, \quad \alpha_r = \frac{d_r}{(\varepsilon + \beta_r)^{2p}}, \quad r = 0, \dots, m,$$

where  $\beta_r$  is a smoothness indicator, e.g., (11.9). The reconstruction polynomial is then given as

$$\pi(x) = \sum_{r=0}^m \omega_r \pi_r(x).$$

The main difference between the CWENO scheme and the traditional WENO schemes is that the global polynomial  $\pi_G$  is a natural part of the former scheme whereas the linear weights must be chosen carefully in the latter scheme to ensure optimal accuracy. Hence, in the CWENO scheme, one is free to choose  $d_r \in [0, 1]$  to optimize the scheme, and the choice does not depend on the underlying grid.

In [4] it is suggested that one chooses  $d_0$ , define the remaining coefficients by

$$\tilde{d}_r = \tilde{d}_{m+1-r} = r, \quad 1 \leq r \leq \frac{m+1}{2},$$

and then update these as

$$d_r = (1 - d_0) \frac{\tilde{d}_r}{\sum_{i=1}^m \tilde{d}_i}.$$

It is advocated to choose  $d_0 = \frac{1}{2}$  or  $d_0 = \frac{2}{3}$ . Note that taking  $d_0 = 0$  renders  $\pi_0$  unbounded, while  $d_0 = 1$  implies that  $\pi_0(x) = \pi_G(x)$  and leaves limited room to control oscillations.

To realize that this approach is as accurate as a standard WENO scheme, consider

$$\begin{aligned} u(x) - \pi(x) &= u(x) - \pi_G(x) + \sum_{r=0}^m (d_r - \omega_r)(\pi_r(x) - u(x)) \\ &= \mathcal{O}(h^{2m-1}) + \mathcal{O}(h^m) \sum_{r=0}^m (d_r - \omega_r). \end{aligned}$$

Hence, as for the WENO scheme, we must require that  $|d_r - \omega_r| \simeq \mathcal{O}(h^{m-1})$  to ensure optimal convergence for the smooth case. This is established in [4], and a discussion of TVD-/TVB-stability of the CWENO method is presented in [21].

The main benefits of the CWENO scheme is the flexibility in the choice of  $d_r$  and their grid independence, which is particularly beneficial when variable or time-dependent grids are needed. We refer to [20, 3] for further discussions and numerical tests of the central WENO schemes.

### 11.3.2 • Well balanced schemes

Let us now consider conservation laws of the form

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = s(u, x), \quad (11.16)$$

where  $s(u, x)$  is a source. Everything we have discussed so far can be applied directly to solve this problem, although one has to be careful when considering behavior close to the equilibrium:

$$\frac{\partial f(u)}{\partial x} = s(u, x).$$

In such cases, the scheme must be able to maintain steady-state solutions which exactly satisfy the balance law. Schemes with this property are known as well balanced schemes and play an important role in certain classes of application, such as shallow water equations or problems with gravity. While it may appear to be important only for steady state problems, there is substantial evidence that this is also a concern when small dynamic perturbations are being studied [43].

Let us write the finite volume scheme for (11.16) as

$$\frac{d\bar{u}_j}{dt} + \frac{1}{h} [F_{j+1/2} - F_{j-1/2}] = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} s(u, x) dx.$$

Following [41] we make two assumptions. Given the exact steady-state solution  $u$ , we assume that there exist two functions  $q(x)$  and  $p(x)$  such that

$$a(u, x) \equiv \frac{u(x) + p(x)}{q(x)} = c, \quad (11.17)$$

where  $c$  is a constant. Furthermore, we assume that the source term can be decomposed as

$$s(u, x) = \sum_{l=1}^L s_l(a(u, x)) t_l'(x), \quad (11.18)$$

where  $s_l$  and  $t_l$  are problem-dependent functions.

First observe that since the cell averaging operator is a linear operation, we recover

$$\bar{u}_j + \bar{p}_j = c \bar{q}_j,$$

which satisfies (11.17).

The first step in an ENO/WENO scheme is to reconstruct the interface values from the cell averages, leading to a general expression

$$u_{j+1/2}^{\pm} = S^{\pm}(\bar{u})_j,$$

where  $S^{\pm}(\bar{u})_j$  reflects the nonlinear ENO/WENO reconstruction and its dependence on the local stencil. Let us now use the same reconstructions for  $p_{j+1/2}^{\pm}$  and  $q_{j+1/2}^{\pm}$ :

$$p_{j+1/2}^{\pm} = S^{\pm}(\bar{p})_j, \quad q_{j+1/2}^{\pm} = S^{\pm}(\bar{q})_j,$$

where  $\bar{p}_j$  and  $\bar{q}_j$  represent the local cell averages of the two known functions  $p(x)$  and  $q(x)$ . While the scheme that defines  $S^{\pm}(\bar{u})_j$  is nonlinear, its subsequent application to evaluate  $p_{j+1/2}^{\pm}$  and  $q_{j+1/2}^{\pm}$  is similar to that of a linear scheme. Hence, we immediately recover that

$$u_{j+1/2}^{\pm} + p_{j+1/2}^{\pm} = c q_{j+1/2}^{\pm},$$

which also implies that

$$a(u, x)_{j+1/2}^{\pm} = c. \quad (11.19)$$

Given (11.18) it is clear that

$$\begin{aligned} \frac{d}{dx} \left[ f(u) - \sum_{l=1}^L s_l(a(u, x)) t_l(x) \right] &= f_x(u) - \sum_{l=1}^L s_l(a(u, x)) t_l'(x) \\ &= f_x(u) - s(u, x) = 0 \end{aligned}$$

such that

$$f(u) - \sum_{l=1}^L s_l(a(u, x)) t_l(x)$$

is a constant for the steady-state solution  $u$ . Hence, we need to define  $t_l^{\pm}(x_{j+1/2})$  such that

$$f(u_{j+1/2}^{\pm}) - \sum_{l=1}^L s_l(a_{j+1/2}^{pm}) t_l^{\pm}(x_{j+1/2}) \quad (11.20)$$

remains constant at the cell interfaces.

The cell average of the source function can be expressed as

$$\begin{aligned}
\int_{x_{j-1/2}}^{x_{j+1/2}} s(u, x) dx &= \sum_{l=1}^L \int_{x_{j-1/2}}^{x_{j+1/2}} s_l(a(u, x)) t_l'(x) dx \\
&= \sum_{l=1}^L \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} \int_{x_{j-1/2}}^{x_{j+1/2}} t_l'(x) dx \\
&\quad + \sum_{l=1}^L \int_{x_{j-1/2}}^{x_{j+1/2}} \left( s_l(a(u, x)) - \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} \right) t_l'(x) dx \\
&= \sum_{l=1}^L \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} (t_l(x_{j+1/2}) - t_l(x_{j-1/2})) \\
&\quad + \sum_{l=1}^L \int_{x_{j-1/2}}^{x_{j+1/2}} \left( s_l(a(u, x)) - \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} \right) t_l'(x) dx.
\end{aligned}$$

This results in the scheme

$$\frac{d\bar{u}_j}{dt} + \frac{1}{h} [F_{j+1/2} - F_{j-1/2}] = \frac{1}{h} S_j,$$

where

$$S_j = \sum_{l=1}^L \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} (t_l(x_{j+1/2}) - t_l(x_{j-1/2})) + \hat{S}_j,$$

with

$$\hat{S}_j = \sum_{l=1}^L \int_{x_{j-1/2}}^{x_{j+1/2}} \left( s_l(a(u, x)) - \frac{s_l(a_{j-1/2}^+) + s_l(a_{j+1/2}^-)}{2} \right) t_l'(x) dx.$$

For the general problem, this integral must be evaluated with adequate accuracy. If, however, we focus on the steady-state problem, then (11.19) implies that  $s_j$  vanishes and the balance law is

$$F_{j+1/2} - F_{j-1/2} = \sum_{l=1}^L s_l(c) (t_l(x_{j+1/2}) - t_l(x_{j-1/2})). \quad (11.21)$$

If we now take

$$F_{j+1/2} = \frac{f(u_{j+1/2}^+) + f(u_{j+1/2}^-)}{2}, \quad t_l(x_{j+1/2}) = \frac{t_l^+(x_{j+1/2}) + t_l^-(x_{j+1/2})}{2},$$

then (11.20) implies that (11.21) is satisfied, resulting in a well balanced scheme.

The Lax-Friedrichs flux is

$$F_{j+1/2} = \frac{f(u_{j+1/2}^+) + f(u_{j+1/2}^-)}{2} - \alpha (u_{j+1/2}^+ - u_{j+1/2}^-).$$

However, this does not reduce to the central flux for the steady solution as the dissipative term is needed for stability. To address this, [41] proposes to redefine the numerical flux as

$$F_{j+1/2} = \frac{f(u_{j+1/2}^+) + f(u_{j+1/2}^-)}{2} - \tilde{\alpha} \text{sign}(q(x_{j+1/2})) (a_{j+1/2}^+ - a_{j+1/2}^-),$$

which vanishes at steady state. To ensure sufficient dissipation, one uses  $\tilde{\alpha} = \alpha / \min q(x)$ , to recover a well balanced stable scheme.

**Example 11.6.** Consider the shallow water equations

$$\frac{\partial h}{\partial t} + \frac{\partial(hu)}{\partial x} = 0, \quad \frac{\partial hu}{\partial t} + \frac{\partial(hu^2 + \frac{1}{2}gb^2)}{\partial dx} = -ghb_x,$$

where  $(h, hu)$  represent the water height and momentum, respectively, while  $g$  is the gravitational constant and  $b(x)$  reflects the bottom topography.

The steady-state solution is given as

$$u = 0, \quad h + b = c.$$

Following the above development, it is natural to define

$$a_1 = h + b, \quad a_2 = 0,$$

and

$$s_1 = s_1(a_1) = -g(h + b), \quad s_2 = s_2(a_2) = \frac{1}{2}g,$$

which leads to

$$-ghb_x = -g(h + b) + \frac{1}{2}g(b^2)_x.$$

By defining

$$t_1(x) = b, \quad t_2(x) = b^2,$$

we recover the form (11.18). This allows the development of a well balanced scheme for the shallow water equations. ■

The development of well balanced schemes for more general steady states, e.g., constant nonzero solutions, is more complicated and discussed at length in [42]. The generalization and development of well balanced schemes using finite difference schemes or discontinuous Galerkin methods are reviewed in [26].

### 11.3.3 • A little more theory

As is the case for the ENO scheme, the inherently nonlinear nature of the WENO scheme makes the development of rigorous stability and convergence results difficult, and such questions generally remain open. In contrast to the results on the sign property of the ENO schemes, such results are not known for the standard WENO scheme, although special schemes with such properties have been proposed [10].

However, if we restrict attention to scalar conservation laws with smooth solutions, the smoothness of the WENO scheme, ensured by the nonlinear weights, allows for some results on accuracy.

Consider the semidiscrete scheme

$$\frac{dU_j}{dt} = G(U)_j = G(U_{j-r-1}, \dots, U_{j+r+1})$$

as a semidiscrete approximation to the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = \frac{\partial u}{\partial t} + f'(u) \frac{\partial u}{\partial x} = 0,$$

where we assume a smooth solution and flux. The starting point is the classic result [36].

**Theorem 11.7.** *Assume that  $G(U)_j$  is a consistent operator of order  $p$  and its first variation is  $L^2$ -stable. If  $u$ ,  $f'(u)$ , and  $G(U)_j$  have  $p + q_0 + 3$  continuous derivatives for all points  $(x_j, t_0)$  then*

$$u(x_j, t_0) = U_j(t_0) + \mathcal{O}(h^p),$$

where  $q_0$  is a constant small integer.

This result also holds for  $d$ -dimensional scalar equations, albeit with a slight dimension-dependent increase in the smoothness requirement.

Hence, once we establish the required smoothness and  $L^2$ -stability of the first variation of  $G(U)_j$ , then accuracy follows. Because of the nonsmooth nature of the ENO scheme, it is clear that this approach does not carry over to this scheme.

By restricting attention to a scheme with a smooth flux, smoothness of  $u$ ,  $f'(u)$ , and  $G(U)_j$  follows. Furthermore, the WENO reconstruction is smooth, ensured by the definition of the nonlinear weights and the smoothness indicator.

Before attempting a more complete discussion, let us first consider an example.

**Example 11.8.** We consider the simple 3-point scheme

$$\frac{dU_j}{dt} = -\frac{F_{j+1/2} - F_{j-1/2}}{h} = -\frac{F(U_j, U_{j+1}) - F(U_{j-1}, U_j)}{h},$$

with the Lax-Friedrichs flux

$$F(u, v) = \frac{1}{2}(f(u) + f(v) - \alpha(v - u)).$$

Here  $\alpha > 0$  is sufficiently large to ensure that the flux is monotone.

Let us now define

$$G(U)_j = -\frac{F(U_j, U_{j+1}) - F(U_{j-1}, U_j)}{h},$$

and compute its first variation

$$\tilde{G}(U)_j = \sum_{i=j-1}^{j+1} \frac{\partial G}{\partial U_i}(U_j) U_i.$$

We recover

$$\begin{aligned}\frac{\partial G}{\partial U_{j-1}}(U_j) &= \frac{1}{2b} (f'(U_j) + \alpha) U_{j-1}, \\ \frac{\partial G}{\partial U_j}(U_j) &= -\frac{1}{2b} 2\alpha U_j, \\ \frac{\partial G}{\partial U_{j+1}}(U_j) &= -\frac{1}{2b} (f'(U_j) - \alpha) U_{j+1},\end{aligned}$$

which, when combined with the previous result, yields

$$\tilde{G}(U)_j = -f'(U_j)D_b^0 U_j + \alpha b D_b^+ D_b^- U_j.$$

As we have already discussed in Ex. 5.6, this scheme is stable since its eigenvalue spectrum is entirely in the left half plane. Hence, a Runge–Kutta scheme of third order guarantees discrete stability and, by Theorem 11.7, we have consistency of the scheme. This is consistent with the analysis in subsection 5.1.1. ■

Following this example, we focus on establishing stability of the first variation, given as

$$\tilde{G}(U)_j = \sum_{l=j-m}^{j+m} \frac{\partial G}{\partial U_l}(U_j) U_l.$$

Assume that we employ a Lax–Friedrichs flux and recall that in the case of a smooth solution, we need  $\omega_r^{r_0} = d_r^{r_0} + \mathcal{O}(b^{m-1})$  to ensure optimal linear accuracy. The linear nature of the reconstruction allows us to recover

$$\begin{aligned}\tilde{G}(U)_j &= -\frac{f'(U_j)}{2b} (U_{j+1/2}^+ + U_{j+1/2}^- - (U_{j-1/2}^+ + U_{j-1/2}^-)) \\ &\quad + \frac{\alpha}{2} (U_{j+1/2}^+ - U_{j+1/2}^- - (U_{j-1/2}^+ - U_{j-1/2}^-)).\end{aligned}$$

To further simplify matters, we introduce the notation

$$\begin{aligned}\beta^+ &= U_{j+1/2}^+ - U_{j-1/2}^+ = \sum_{r=0}^{m-1} d_{m-1-r}^0 \sum_{i=0}^{m-1} c_{i,r-1} \Delta^+ U_{j+i-r}, \\ \beta^- &= U_{j+1/2}^- - U_{j-1/2}^- = \sum_{r=0}^{m-1} d_r^0 \sum_{i=0}^{m-1} c_{i,r} \Delta^+ U_{j+i-r-1},\end{aligned}$$

so that the first variation can be written as

$$\tilde{G}(U)_j = -\frac{f'(U_j)}{2b} (\beta^- + \beta^+) + \frac{\alpha}{2} (\beta^+ - \beta^-).$$

A straightforward, but lengthy, algebraic manipulation shows that this can be rewritten as

$$\tilde{G}(U)_j = -f'(U_j)D_b^{1,2m} U_j + (-1)^{m+1} b^{2m-1} \alpha C(m) D_b^{2m,2m} U_j,$$

where the operator  $D_b^{p,q} U_j$  is the  $q$ th order accurate  $x_j$ -centered finite difference approximation to the  $p$ th order derivative, e.g.,  $D_b^{1,2m} U_j$  is the  $2m$ th order centered approximation of the first order derivative. Furthermore  $C$  is a positive constant that depends on  $m$ .

Proceeding as in the previous example, we recall that the central finite difference approximation to the first order operator has a purely imaginary spectrum, while the second term is an operator with a negative real spectrum. Hence, using a time-stepping methods such as a third or fourth order Runge–Kutta method guarantees that the scheme is stable. We have thus established the following result [17].

**Theorem 11.9.** *Consider the scalar, initial value problem. If at all points  $(x_0, t_0) \in R^d \times R^+$ , the solution  $u$ , the flux Jacobian  $f'$ , and the reconstruction have  $p + [(d+1)/2] + q_0 + 2$  continuous derivatives, then the WENO scheme with a smooth flux and an  $n$ th order Runge–Kutta method ( $n \geq \max(p, 3)$ ) satisfies*

$$u(x_0, t_0) = U(x_0, t_0) + \mathcal{O}(h^p),$$

for an appropriately chosen CFL condition. Here  $q_0$  is a constant small integer.

Hence, when using a WENO method to solve a scalar conservation law with a smooth solution, we expect stability and optimal accuracy.

While this result is encouraging, it is also relatively narrow. However, no general results on stability and convergence beyond this are currently known. Generally, one observes high-order accuracy in smooth parts of the solution when scalar problems are considered. The situation is less clear for systems due to the counter propagating characteristics that may mix numerical errors and cause the scheme to fail to achieve optimal order of convergence. Nevertheless, even though the formal order of accuracy may not be achieved in such cases, one typically observes a substantial improvement in overall accuracy, as compared to that of a lower-order scheme.

As we shall discuss further in Chapter 13, lack of pointwise accuracy may be overcome through a post-processing approach. While primarily developed for spectral methods, the successful application of such ideas to solutions, obtained with WENO schemes, has been demonstrated [13].

### 11.3.4 • WENO methods in action

A first step towards bringing WENO methods to life is the implementation of the WENO reconstruction based on a list of  $2m - 1$  values, as shown in `WENO.m`. With this in place, the differences between the implementation of the ENO and the WENO reconstruction are minimal.

**Script 11.19. *WENO.m: Reconstruction of function values at cell interfaces by a WENO approach.***

---

```

function [um,up] = WENO(xloc ,uloc ,m,Crec ,dw,beta);
% Purpose: Compute the left and right cell interface values using an WENO
%           approach based on 2m-1 long vectors uloc with cell
% Set WENO parameters
p = 1; q = m-1; vareps = 1e-6;

% Treat special case of m=1 - no stencil to select
if (m==1)
    um = uloc(1); up = uloc(1);

    % Treat special case of m=1 - no stencil to select
    if (m==1)
        um = uloc(1); up = uloc(1);
    end
end

```

```

else
    alpham = zeros(m,1); alphap = zeros(m,1);
    upl = zeros(m,1); uml = zeros(m,1); betar = zeros(m,1);

    % Compute um and up based on different stencils and
    % smoothness indicators and alpha
    for r=0:m-1;
        umh = uloc(m-r+[0:m-1]);
        upl(r+1) = Crec(r+2,:)*umh; uml(r+1) = Crec(r+1,:)*umh;
        betar(r+1) = umh'*beta(:,:,r+1)*umh;
    end;

    % Compute alpha weights - classic WENO
    alphap = dw./(vareps+betar).^(2*p);
    alpham = flipud(dw)./(vareps+betar).^(2*p);

    % % Compute alpha weights - WENO-Z
    % tau = abs(betar(1) - betar(m));
    % if mod(m,2)==0
    %     tau = abs(betar(1)-betar(2) - betar(m-1) + betar(m));
    % end
    % alphap = dw.* (1 + (tau./(vareps+betar)).^q);
    % alpham = flipud(dw).* (1 + (tau./(vareps+betar)).^q);

    % Compute nonlinear weights and cell interface values
    um=alpham'*uml/sum(alpham); up=alphap'*upl/sum(alphap);
end
return

```

## Maxwell's equations

As a first example, we consider the solution of Maxwell's equations. The implementation of this solver, as compared to the ENO method, require only minimal changes, comprising the precomputation of the linear weights and the smoothness indicators, as illustrated in `MaxwellWENO1D.m`. The other routines, `MaxwellWENODriver1D.m` and `MaxwellWENOrhs1D.m` are essentially identical to `MaxwellENODriver1D.m` and `MaxwellENOrhs1D.m`, respectively, and are not shown.

**Script 11.20. `MaxwellWENO1D.m`: Time integration routine for Maxwell's equations, including the initializations needed for the WENO method.**

```

function [q] = MaxwellWENO1D(x,q,ep, mu, h,m, CFL, FinalTime)
% function [q] = MaxwellWENO1D(x,q,ep, mu, h,m, CFL, FinalTime)
% Purpose : Integrate 1D Maxwell's equation until FinalTime using a WENO
%             scheme and a 3rd order SSP-RK method.
time = 0; tstep = 0;

% Initialize reconstruction weights
Crec = zeros(m+1,m);
for r=-1:m-1;
    Crec(r+2,:) = ReconstructWeights(m,r);
end;

% Initialize linear weights
dw = LinearWeights(m,0);

% Compute smoothness indicator matrices
beta = zeros(m,m,m);
for r=0:m-1

```

```

x1 = -1/2 + [-r:1:m-r];
beta (:,:,r+1) = betarcalc (x1,m);
end

% Set timestep
cvel = 1./sqrt(ep.*mu); k = CFL*h/max(cvel); maxvel = max(cvel);

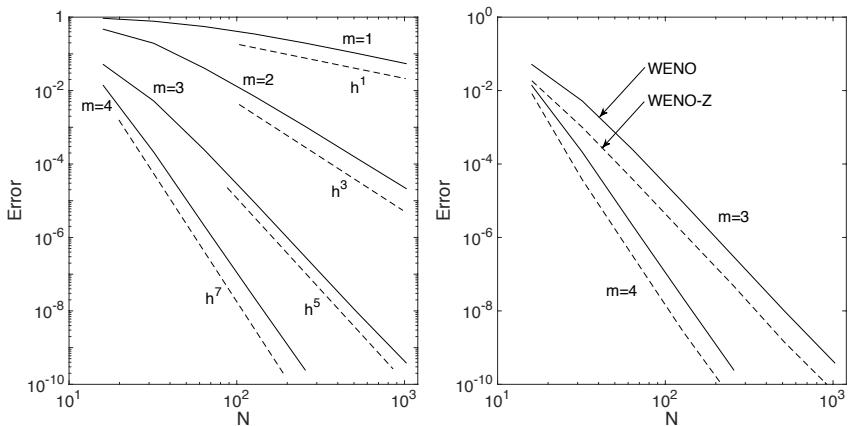
% integrate scheme
while (time<FinalTime)
  if (time+k>FinalTime) k = FinalTime-time; end
  % Update solution
  rhsq = MaxwellWENOrhs1D(x, q, ep, mu, h, k, m, Crec, dw, beta, maxvel);
  q1 = q + k*rhsq;
  rhsq = MaxwellWENOrhs1D(x, q1, ep, mu, h, k, m, Crec, dw, beta, maxvel);
  q2 = (3*q + q1 + k*rhsq)/4;
  rhsq = MaxwellWENOrhs1D(x, q2, ep, mu, h, k, m, Crec, dw, beta, maxvel);
  q = (q + 2*q2 + 2*k*rhsq)/3;
  time = time+k; tstep = tstep+1;
end
return

```

We consider the case of a smooth solution. In Fig. 11.14 we show the results for different values of  $m$  and increasing resolution  $N$ . As expected, we observe optimal order of convergence  $\mathcal{O}(h^{2m-1})$ .

Figure 11.14 also illustrates the impact of using the WENO-Z weights, rather than the classic weights. Although the WENO-Z weights do not change the actual order of convergence they clearly lead to an improvement of the overall accuracy, often by more than an order of magnitude, achieved at minimal additional cost.

Solving a problem with a nonsmooth material coefficient using a WENO scheme yields results similar to those in Fig. 11.9, provided an upwind flux is used and sufficient care is taken to ensure grid alignment.



**Figure 11.14.** The left panel shows the  $L^2$ -error of the WENO scheme of order  $m$  for Maxwell's equations with a smooth solution, measured at  $T = \sqrt{2}$  and a sufficiently small time step chosen to ensure negligible temporal errors. The right panel shows the  $L^2$ -error for  $m = 3$  and  $m = 4$ , computed using a classic WENO scheme and the WENO-Z scheme. In both cases, we use a Lax-Friedrichs flux.

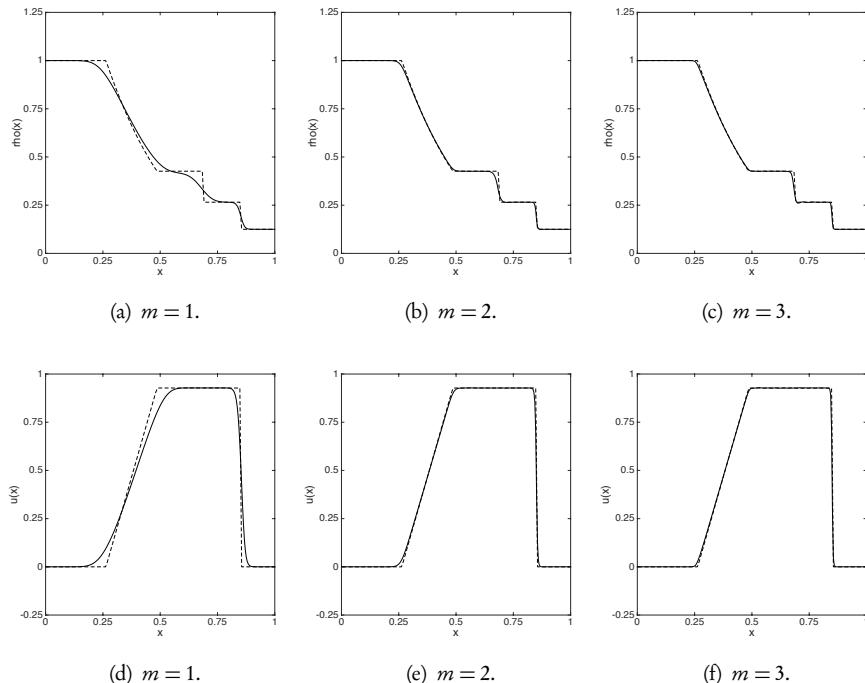
## The Euler equations

Let us now investigate the performance of the WENO scheme for the more challenging Euler equations. As for Maxwell's equations, the modifications of the ENO scheme are minimal.

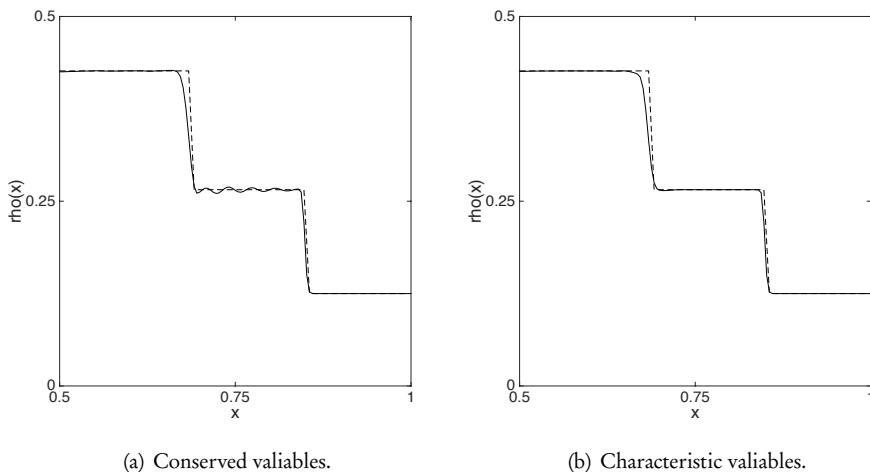
First, we consider Sod's problem; see subsection 1.4.1. Figure 11.15 shows the results for different orders  $m$ . As expected, increasing the order of approximation has a dramatic impact on the quality of the solution, in particular on the resolution of the rarefaction wave and the linear contact wave.

Careful inspection of the solution in the neighborhood of the shock for  $m = 3$  indicates minor oscillations. This behavior is highlighted in Fig. 11.16, where the solution, obtained with  $m = 4$  is shown in the neighborhood of the shock. The oscillations are now clear.

As discussed previously, the source of such oscillations is that the reconstruction is based on the conserved variables, rather than on the characteristic variables which are more naturally associated with upwinding [27]. To enable this, we need EulerChar.m, introduced in Chapter 10. Once the characteristic variables are computable, the reconstruction of the cell-interface values can be based on these quantities and we recover the reconstructed approximations to the interface values of the conserved variables. The entire process is illustrated in EulerWENOcharrhs1D.m.



**Figure 11.15.** Sod's shock tube problem at  $T = 0.2$ . We employ  $N = 256$  cells and different orders  $m$  of the WENO scheme. The solid lines represent the computed solution, while the dashed line reflects the exact solution. The top row shows the density  $\rho$ , while the bottom row shows the velocity  $u$ .



**Figure 11.16.** Blowup of the solution computed with a WENO scheme with  $m = 4$  and  $N = 256$ . The reconstruction is done on the conserved variables (left) or based on the characteristic variables (right).

**Script 11.21. EulerWENOcharrhs1D.m:** WENO reconstruction for Euler equations based on the characteristic variables.

```

function [dq] = EulerWENOcharrhs1D(x,q,h,k,m,Crec,dw,beta, gamma, maxvel)
% function [dq] = EulerWENOcharrhs1D(x,q,h,k,m,Crec,dw,beta, gamma, maxvel)
% Purpose: Evaluate right hand side for Euler equations using a WENO method
%           on the characteristic variables

N = length(x); dq = zeros(N,3);
q1 = zeros(N,3); qr = zeros(N,3); qp = zeros(N,3); qm = zeros(N,3);

% Extend data and assign boundary conditions for Sod's problem
[xe,re] = extend(x,q(:,1),h,m,'D',1.0,'D',0.125);
[xe,me] = extend(x,q(:,2),h,m,'D',0,'N',0);
[xe,Ee] = extend(x,q(:,3),h,m,'D',2.5,'N',0);

% Extend data and assign boundary conditions for shock entropy problem
% [xe,re] = extend(x,q(:,1),h,m,'D',3.857143,'N',0);
% [xe,me] = extend(x,q(:,2),h,m,'D',10.141852,'D',0);
% [xe,Ee] = extend(x,q(:,3),h,m,'D',39.166661,'N',0);

% define cell left and right interface values
qe = [re me Ee]; Rlh = zeros(1,3); Rrh= zeros(1,3);
rm = zeros(N+2,1); mm = zeros(N+2,1); Em = zeros(N+2,1);
rp = zeros(N+2,1); mp = zeros(N+2,1); Ep = zeros(N+2,1);
for i=1:N+2
    qloc = qe(i:(i+2*(m-1)),:); q0 = qloc(m,:);
    [S, iS, Lam] = EulerChar(q0,gamma); Rloc = (iS*qloc)';
    [Rlh(1),Rrh(1)] = WENO(xe(i:(i+2*(m-1))),Rloc(:,1),m,Crec,dw,beta);
    [Rlh(2),Rrh(2)] = WENO(xe(i:(i+2*(m-1))),Rloc(:,2),m,Crec,dw,beta);
    [Rlh(3),Rrh(3)] = WENO(xe(i:(i+2*(m-1))),Rloc(:,3),m,Crec,dw,beta);
    qlh = (S*Rlh)';
    qrh = (S*Rrh)';
    rm(i) = qlh(1); mm(i) = qlh(2); Em(i) = qlh(3);
    rp(i) = qrh(1); mp(i) = qrh(2); Ep(i) = qrh(3);
end;

```

```
% Compute rbs - also change numerical flux here
q1 = [rm(2:N+1) mm(2:N+1) Em(2:N+1)]; qr = [ rp(2:N+1) mp(2:N+1) Ep(2:N+1)];
qp = [rm(3:N+2) mm(3:N+2) Em(3:N+2)]; qm = [ rp(1:N) mp(1:N) Ep(1:N) ];
dq = - (EulerLF(qr, qp, gamma, k/h, maxvel)) - ...
       EulerLF(qm, q1, gamma, k/h, maxvel))/h;
return
```

Figure 11.16 shows the computed solution with  $m = 4$  and confirms that a reconstruction based on the characteristic variables eliminates the unphysical oscillations. While it is tempting to conclude that a reconstruction based on the characteristic variables is always the preferred approach, the added computational cost needs to be considered carefully. As the examples show, the use of the characteristic variables is beneficial only when  $m$  is high, e.g., greater than 2, or if the shocks are strong.

Results for the more challenging case of the shock-entropy problem are shown in Fig. 11.17. When compared to the similar results, obtained with the ENO method in Fig. 11.12, we observe a substantial improvement of the quality of the solution at fixed resolution. As one would expect, this is particularly clear for the highly oscillatory region upstream of the shock.

A careful inspection of the results in Fig. 11.17 also reveals signs of oscillations or overshoots for  $m = 4$ . Figure 11.17 also shows the results for  $m = 4$  with the characteristic reconstruction in which this artificial effect is eliminated.

To further highlight the performance of the WENO scheme and its different variants, a comparison of the solution for the shock-entropy wave problem is shown in Fig. 11.18. In all cases, we use  $m = 3$  and  $N = 256$  grid points. We use a Roe flux and all reconstructions are done on the characteristic variables. As expected, we observe a substantial improvement when compared to the results obtained with the ENO scheme. Furthermore, we also see an improvement in the accuracy of the solution computed with the WENO-Z weights, as compared to the classic WENO weights, in full agreement with the previous discussion.

## 11.4 • Dealing with nonuniform grids

So far we have based all discussions on an assumption of an equidistant grid, utilizing the simplifications that come with this assumption. However, for many problems, i.e., problems with strongly localized behavior, the use of a nonuniform grid is appealing.

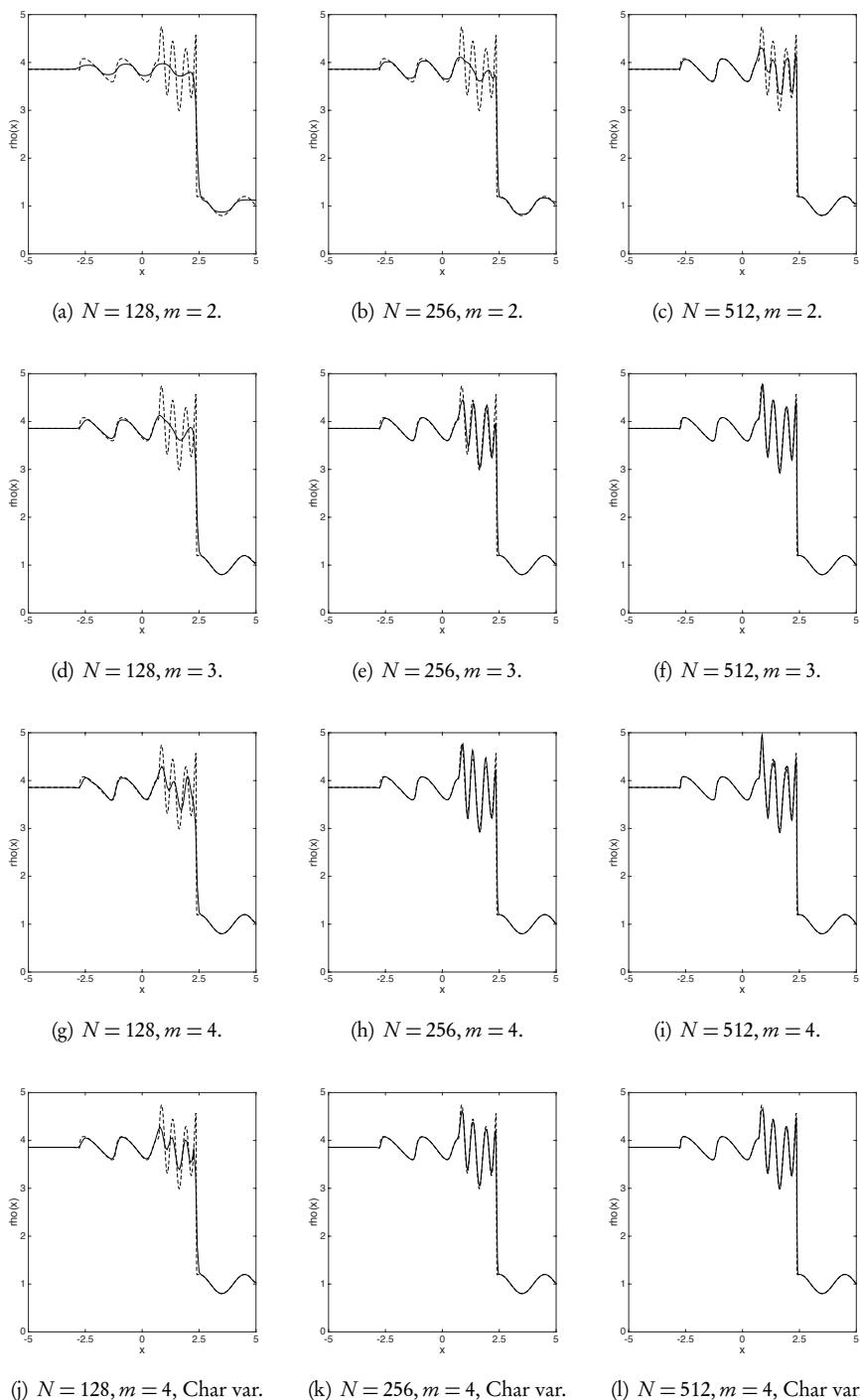
In order to understand the potential challenges in pursuing such a development, loosely following [25], let us define the generic cell averaging operator:

$$\mathcal{A}_h u(x) = \frac{1}{\Delta_\alpha x(\alpha)} \int_{x(\alpha-h/2)}^{x(\alpha+h/2)} u(y) dy = \frac{1}{\Delta_\alpha x(\alpha)} \int_{\alpha-h/2}^{\alpha+h/2} u(x(\beta)) x'(\beta) d\beta.$$

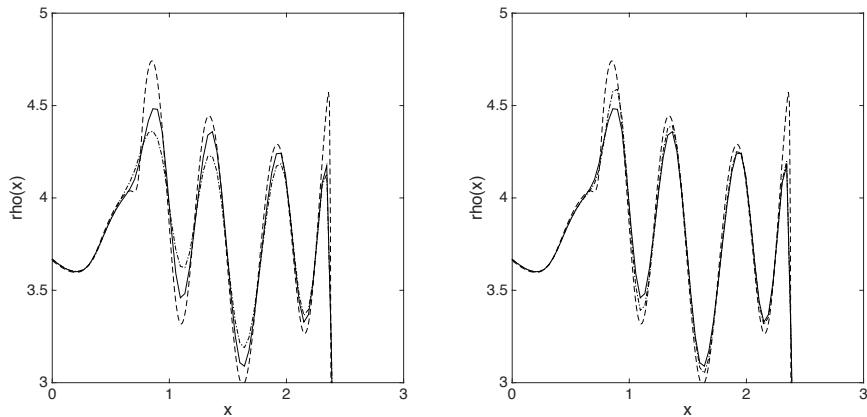
Here  $x(\alpha)$  represents the physical grid, smoothly mapped from  $\alpha$ , which we assume discretized in an equidistant manner. The mapping is reflected in the Jacobian,  $x'(\alpha)$ . We note that if the mapping is affine, the Jacobian is constant. We also introduce the difference operator  $\Delta_\alpha g(\alpha) = g(\alpha + h/2) - g(\alpha - h/2)$ .

Now consider the conservation law,

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0,$$



**Figure 11.17.** Computed density for the shock-entropy problem at  $T = 1.8$ , obtained for different orders  $m$  and number of grid points  $N$ . For the top three rows, the reconstruction is done on the conserved variables. In the last row, we use  $m = 4$  and the reconstruction is done on the characteristic variables.



**Figure 11.18.** Detail of the computed density of the shock-entropy solution, obtained with different variations of the WENO scheme. For all tests we use  $N = 256$ ,  $m = 3$ , and a Roe flux. In both figures, the full line represents the classic WENO scheme and the dashed line is the high-resolution reference solution. On the left, the dash-dotted line represents the third order ENO scheme, while the dash-dotted result on the right is obtained with a WENO-Z reconstruction. In all cases, the reconstruction is based on the characteristic variables.

and express the finite volume scheme as

$$\frac{d}{dt} \mathcal{A}_b u(x(\alpha)) + \frac{f(u(x(\alpha + b/2))) - f(u(x(\alpha - b/2)))}{\Delta_\alpha x(\alpha)} = 0,$$

where  $u(x(\alpha \pm b/2))$  must be reconstructed. This is Godunov's scheme and nothing suggests that a nonuniform grid raises any issue, provided that we can perform the reconstruction with adequate accuracy. In other words, the formulation of high-order finite volume schemes on nonuniform grids is straightforward.

We rewrite the finite volume scheme as

$$\frac{d}{dt} \mathcal{A}_b u(x(\alpha)) + \frac{\Delta_\alpha f(u(x(\alpha)))}{\Delta_\alpha x(\alpha)} = 0$$

and recover a pointwise finite difference formulation in the form

$$\frac{d}{dt} u(x(\alpha)) + \mathcal{A}_b^{-1} \frac{\Delta_\alpha f(u(x(\alpha)))}{\Delta_\alpha x(\alpha)} = 0.$$

We now define  $F$  through  $\mathcal{A}_b F = f$  or, equivalently,

$$\mathcal{A}_b F(x) = \frac{1}{\Delta_\alpha x(\alpha)} \int_{x(\alpha-b/2)}^{x(\alpha+b/2)} F(y) dy = f(u(x(\alpha))).$$

If we now assume that  $\mathcal{A}_b^{-1}$  and  $\Delta_\alpha$  commute such that

$$\mathcal{A}_b^{-1} \Delta_\alpha f(u(x(\alpha))) = \Delta_\alpha \mathcal{A}_b^{-1} f(u(x(\alpha))), \quad (11.22)$$

we recover the finite difference scheme

$$\frac{d}{dt} u(x(\alpha)) + \frac{\Delta_\alpha F(x(\alpha))}{\Delta_\alpha x(\alpha)} = 0.$$

This highlights the connection between the finite volume and the finite difference scheme, as already discussed at the beginning of Chapter 10.

This result, however, relies on the assumption that  $\mathcal{A}_b^{-1}$  and  $\Delta_\alpha$  commute, as expressed in (11.22). To understand the implications of this assumption, let us introduce  $\mathcal{A}_b g = f$  and consider the commutation error

$$\mathcal{A}_b^{-1} \Delta_\alpha f - \Delta_\alpha \mathcal{A}_b^{-1} f = \mathcal{A}_b^{-1} [\Delta_\alpha \mathcal{A}_b - \mathcal{A}_b \Delta_\alpha] g.$$

If we further assume that  $g(x)$  is arbitrarily smooth, we have

$$\begin{aligned} \Delta_\alpha \mathcal{A}_b g(x(\alpha)) &= \int_{\alpha-h/2}^{\alpha+h/2} [g(x(\beta+h/2))x'(\beta+h/2) - g(x(\beta-h/2))x'(\beta-h/2)] d\beta, \\ \mathcal{A}_b \Delta_\alpha g(x(\alpha)) &= \int_{\alpha-h/2}^{\alpha+h/2} [g(x(\beta+h/2)) - g(x(\beta-h/2))] x'(\beta) d\beta, \end{aligned}$$

which allows us to express the commutator as

$$\begin{aligned} [\Delta_\alpha \mathcal{A} - \mathcal{A} \Delta_\alpha] g(x(\alpha)) &= \int_{\alpha-h/2}^{\alpha+h/2} g(x(\beta+h/2)) [x'(\beta+h/2) - x'(\beta)] \\ &\quad - g(x(\beta-h/2)) [x'(\beta-h/2) - x'(\beta)] d\beta. \end{aligned}$$

For this to vanish for all functions  $g$ , we must require that

$$x'(\beta+h/2) \pm x'(\beta) = 0,$$

for all values of  $\beta$  and  $h$ . This is clearly possible only when  $x'(\alpha)$  is a constant. Hence, assumption (11.22) holds only for equidistant grids.

The unfortunate consequence of this result is that we cannot pursue the development of high-order accurate finite difference formulations on uneven grids as we will lose either conservation or accuracy. In [25], examples of special smoothly varying grids that allow one to overcome this restriction are discussed. However, for general nonsmooth grids, conservative finite difference schemes with a local flux cannot exceed second order accuracy [32].

Let us now return to the details of the finite volume scheme on uneven grids. To enable this, we only need to modify the expression of three sets of coefficients:  $c_{ir}^m$  to allow the reconstruction in both ENO and WENO schemes, and  $d_r^{\gamma_0}$  and  $\beta_r$  to enable the computation of the nonlinear weights in the WENO scheme. These coefficients become locally varying functions and depend on the grid size of all neighboring elements. The approach discussed previously generalizes directly to this case, and we shall not pursue the details, some of which can be found in [31].

## 11.5 ▪ Beyond one dimension

In the following we discuss the extension of essentially nonoscillatory schemes to the case of two spatial dimensions. To keep things simple, we restrict ourselves to equidistant Cartesian grids. In this case, as discussed at length in Chapter 7, the extension to problems beyond one dimension is particularly simple.

As we are now aiming at methods of arbitrary accuracy, we need to revisit the developments in Chapter 7. In general, from the integral form of the conservation laws, we recover that

$$\bar{u}_{ij}^{n+1} = \bar{u}_{ij}^n - \frac{1}{h_x h_y} \int_{t^n}^{t^{n+1}} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} \nabla \cdot f(u) dx dy,$$

where the cell average is defined as

$$\bar{u}_{ij}^n = \frac{1}{h_x h_y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} u(x, y, t^n) dx dy.$$

We rewrite the flux term as

$$\int_{\Omega} \nabla \cdot f(u) dx dy = \int_{\partial\Omega} \hat{n} \cdot f(u) dx dy,$$

where  $\Omega$  represents the control volume  $[x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{j+1/2}]$ .

In Chapter 7, we used the midpoint rule to approximate the four surface integrals as

$$\int_{x_{i-1/2}}^{x_{i+1/2}} f(u(x, y)) dx \simeq h_x f(x_i, y),$$

which is generally  $\mathcal{O}(h_x^2)$  accurate. If we were to pursue this same dimensional approach, leading to the simple scheme outlined in Chapter 7, we could not, in general, expect an accuracy better than  $\mathcal{O}(h^2)$ , regardless of the accuracy of the one-dimensional reconstruction.

The natural solution to this problem is to apply a higher-order integration rule to approximate the surface integrals

$$\int_{x_{i-1/2}}^{x_{i+1/2}} f(u(x, y)) dx \simeq \sum_{i=1}^m f(x_i, y) \omega_i.$$

More specifically, one could use an integration formula whose accuracy is equal to that of the reconstruction. While this may appear straightforward, its computational cost is considerable as the number of required reconstructions in each cell grows quickly. A simple estimate of the number of reconstructions needed for a scheme in  $d$ -dimensions with  $m$  stencils and a quadrature with  $m$  points yields  $m^{d+1}/(d-1)!$  reconstructions. Clearly, for high values of  $m$  or  $d$ , this becomes both expensive and complex.

In [2] this is addressed by assuming local smoothness of the solution. The one-dimensional reconstruction at  $(x_{i+1/2}, j)$  recovers

$$\bar{u}_{i+1/2,j}^+ = \frac{1}{b_y} \int_{-b_y/2}^{b_y/2} u_b^n(x_{i+1/2}, y_j + s) ds,$$

as the approximation to the edge-averaged solution. By assuming that the solution is sufficiently smooth along  $y$ , we can express it as

$$\begin{aligned}\bar{u}_{i+1/2,j}^+ &= \frac{1}{b_y} \int_{-b_y/2}^{b_y/2} \sum_{l=0}^p u_y^{(l)}(x_{i+1/2}, y_j) \frac{s^l}{l!} ds \\ &= \sum_{l=0}^{p/2} \frac{1}{(2l)!} \left(\frac{b_y}{2}\right)^{2l} \frac{1}{2l+1} u_y^{(2l)}(x_{i+1/2}, y_j) \\ &= u(x_{i+1/2}, y_j) + \frac{b_y^2}{24} u_y^{(2)}(x_{i+1/2}, y_j) + \frac{b_y^4}{1920} u_y^{(4)}(x_{i+1/2}, y_j) + \dots,\end{aligned}$$

where  $u_y^{(l)}$  represents the  $l$ th derivative along  $y$ . Restricting attention to  $\mathcal{O}(b^5)$  order schemes, we recover

$$u_b(x_{i+1/2}, y_j)^+ = \bar{u}_{i+1/2,j}^+ - \frac{b_y^2}{24} u_y^{(2)}(x_{i+1/2}, y_j) - \frac{b_y^4}{1920} u_y^{(4)}(x_{i+1/2}, y_j).$$

The spatial derivatives can be approximated using finite difference schemes along  $y$ , e.g.,

$$u_y^{(2)}(x_{i+1/2}, y_j) \simeq \frac{1}{b_y^2} (\bar{u}_{i+1/2,j-1}^+ - 2\bar{u}_{i+1/2,j}^+ + \bar{u}_{i+1/2,j+1}^+).$$

In [2] this is shown to yield an  $\mathcal{O}(b^5)$  scheme and extensions to higher order are also discussed. The modification to finite difference schemes can be pursued in a similar manner by directly modifying the numerical flux.

In the following, we sacrifice the formal accuracy for the simplicity of the scheme, and retain the simple dimensional structure, while acknowledging that the formal order of the scheme is  $\mathcal{O}(b^2)$ .

### 11.5.1 • Interlude on non-Cartesian boundaries

Before continuing with the evaluation of essentially nonoscillatory schemes for two-dimensional conservation laws, let us discuss techniques to impose boundary conditions along nonconforming boundaries. Since we continue to rely on a Cartesian grid, the simple extension of solutions used so far will not suffice, and the question arises of how to generate the ghost values needed for the essentially nonoscillatory reconstruction.

To pursue this, let us consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad x \in [a, b],$$

subject to appropriate boundary conditions. Without loss of generality, we assume that  $x = a$  is an inflow boundary, subject to the boundary condition  $u(a, t) = g(t)$ , while  $x = b$  represents an outflow boundary at which no explicit boundary condition is required.

Let us furthermore assume that we have a grid such that

$$a = -\gamma^- b + x_0, \quad x_0, \dots, x_N, \quad \gamma^+ b + x_N = b, \quad \gamma^\pm \geq 0,$$

i.e., there are no grid points located direct at the boundary points.

If we first consider the situation at the inflow boundary, let us make the assumption that both the solution and the flux are smooth at the boundary. We can express the solution  $u(x, t)$  by its Taylor expansion:

$$u(x_j, t) = \sum_{l=0}^{m-1} \frac{(j + \gamma^-)^l h^l}{l!} \left. \frac{\partial^l u(x, t)}{\partial x^l} \right|_{x=a} + \mathcal{O}(h^m),$$

where  $j = -1, -2, \dots$ . To evaluate this at the required order, we need to recover spatial derivatives of the solution at the boundary point. For  $l = 0$ , we have the boundary condition  $u(a, t) = g(t)$ . To recover the next term, we explore the smoothness of the problem at the boundary to write the conservation law as

$$\frac{\partial u}{\partial t} + f'(u) \frac{\partial u}{\partial x} = 0,$$

from which we directly recover

$$\left. \frac{\partial u}{\partial x} \right|_{x=a} = -\frac{g'(t)}{f'(g(t))}.$$

Repeating this process yields [37]

$$\left. \frac{\partial^2 u(x, t)}{\partial x^2} \right|_{x=a} = \frac{f'(g(t))g''(t) - 2f''(g(t))g'(t)^2}{f'(g(t))^3}.$$

Albeit at considerable computational complexity, this process can be continued to arbitrary order, leading to what is known as the inverse Lax–Wendroff method [37], recognizing the close connection to the Lax–Wendroff method, derived in section 5.2. A systematic approach follows by the use of the Cauchy–Kovalevskaya procedure to interchange spatial and temporal derivatives by using the equation and the generalized Leibniz rules. This is an approach also used in high-order time-stepping in a class of methods known as ADER methods. Details on this can be found in [39, 7].

Let us now consider the outflow. We can again express it as

$$u(x_{N+j}, t) = \sum_{l=0}^{m-1} \frac{(j - \gamma^+)^l h^l}{l!} \left. \frac{\partial^l u(x, t)}{\partial x^l} \right|_{x=b} + \mathcal{O}(h^m),$$

where  $j = 1, 2, \dots$ . If the solution is smooth, we can replace the solution with its interpolating polynomial  $\pi$  based on  $[x_{N-(m-1)}, \dots, x_N]$  and differentiate this to recover the extrapolated values.

If the solution lacks smoothness, more care is needed in the extrapolation. Let us refer to the polynomial  $p_r(x)$  as the interpolating polynomial based on  $[x_{N-r}, x_N]$ , i.e.,  $p_0 = u_N$ . Using the elements of the WENO method, we express the extrapolation polynomial  $\pi(x)$  as

$$\left. \frac{\partial^l \pi}{\partial x^l} \right|_{x=b} = \sum_{r=0}^{m-1} \omega_r \left. \frac{\partial^l p_r}{\partial x^l} \right|_{x=b},$$

where  $\omega_r$  are the nonlinear weights defined in (11.8). The smoothness indicator is

defined as

$$\beta_r = \sum_{l=1}^{m-1} \int_{x_N}^{x_N+b} b^{2l-1} \left( \frac{d^l p_r}{dx^l} \right)^2 dx.$$

These can either be expressed explicitly, as in [37] for  $m = 3$ , or evaluated at general order by the process discussed in Ex. 11.5. As discussed in detail in [37], this process maintains the maximum accuracy at the boundary and recovers the centered extrapolation in case of a smooth solution. Stability of this procedure in the linear case is discussed in [37, 22].

The extension of the inverse Lax–Wendroff procedure to systems requires the use of the characteristic variables to properly define inflow and outflow directions. Otherwise, the procedure extends directly with the caveat that the transfer of spatial to temporal derivatives may become algebraically very complex. A modification to address this is discussed in [38] and the Cauchy–Kovalevskaya procedure is likewise applicable [39, 7].

To extend the inverse Lax–Wendroff procedure to deal with multidimensional problems with nonconforming boundaries is, in principle, straightforward. The inflow/outflow direction must now be defined along the normal to the boundary and the corresponding characteristic speeds and variables computed to impose the appropriate boundary conditions at inflow and enable extrapolation at outflow. While conceptionally simple, its implementation is made complicated by the need to perform WENO interpolations along the normal, which generally is not aligned with the grid. This process and its successful use is discussed in some detail in [37, 38].

### 11.5.2 • Scalar equations

We first consider the two-dimensional, nonconvex problem

$$\frac{\partial u}{\partial t} + \nabla \cdot f(u) = 0,$$

with  $f(u) = (\cos(u), \sin(u))$ . We use a discontinuous initial condition, as discussed in subsection 1.4.2.

As compared to Burgers' equation, discussed in subsection 11.2.3, the modifications needed to extend the computations from one to several dimensions are minimal. In fact, the only substantial change is found in the evaluation of the right-hand side, illustrated in `KPPWENOrhs2D.m`. We note in particular the directional application of the WENO reconstruction as the primary new element.

**Script 11.22. `KPPWENOrhs2D.m`: WENO reconstruction for the two-dimensional KPP equations.**

---

```

function [du] = KPPWENOrhs2D(x,y,u,hx,hy,k,m,Crec,dw,beta,maxvel);
% function [du] = KPPWENOrhs2D(x,y,u,hx,hy,k,maxvel);
% Purpose: Evaluate right hand side for 2D KPP equation using
%           a WENO method
Nxy = size(x); Nx = Nxy(2); Ny = Nxy(1); du = zeros(Ny,Nx);

% Extend data and assign boundary conditions in x-direction
for i=1:Ny
    [xe,ue] = extend(x(i,:),u(i,:),hx,m,'D',pi/4,'D',pi/4);

    % define cell left and right interface values

```

```

ul = zeros(Nx+2,1); ur = zeros(Nx+2,1);
for j=1:Nx+2
    [ul(j),ur(j)] = WENO(xe(j:(j+2*(m-1))),ue(j:(j+2*(m-1))),m,Crec,dw,beta);
end;

% Update residual
du(i,:) = - (KPPxLF(ur(2:Nx+1),ul(3:Nx+2),0,maxvel) ...
              - KPPxLF(ur(1:Nx),ul(2:Nx+1),0,maxvel))/hx;
end

% Extend data and assign boundary conditions in y-direction
for j=1:Nx
    [xe,ue] = extend(y(:,j),u(:,j),hy,m,'D',pi/4,'D',pi/4);

    % define cell left and right interface values
    ul = zeros(Ny+2,1); ur = zeros(Ny+2,1);
    for i=1:Ny+2
        [ul(i),ur(i)] = WENO(xe(i:(i+2*(m-1))),ue(i:(i+2*(m-1))),m,Crec,dw,beta);
    end;

    % Update residual
    du(:,j) = du(:,j) - (KPPyLF(ur(2:Ny+1),ul(3:Ny+2),0,maxvel) ...
                          - KPPyLF(ur(1:Ny),ul(2:Ny+1),0,maxvel))/hy;
end
return

```

Figure 11.19 illustrates the performance of the two-dimensional WENO scheme for the KPP problem, highlighting the convergence towards the reference solution for increasing order  $m$ , and spatial resolution  $N_x$  and  $N_y$ .

### 11.5.3 • The Euler equations

Consider the two-dimensional Euler equations

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f} = 0,$$

where the conserved variables,  $\mathbf{q}$ , and the flux,  $\mathbf{f}$ , are given as

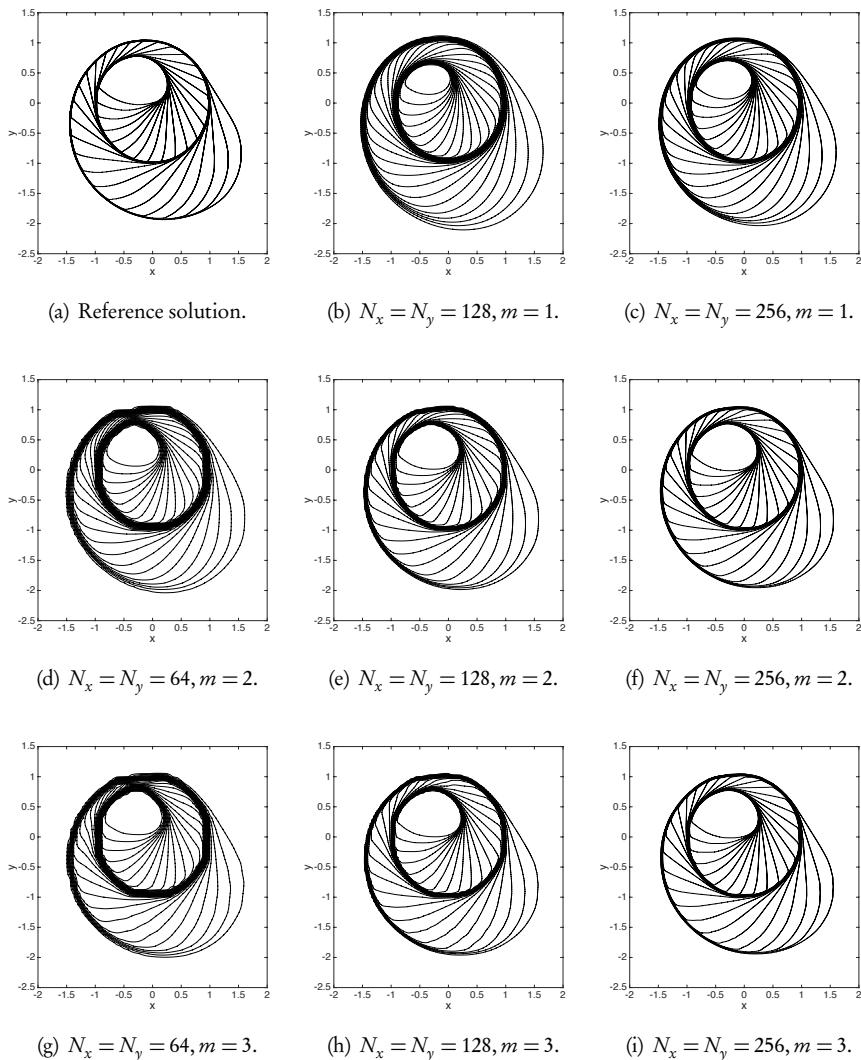
$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \quad \mathbf{f}_1 = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (E + p)u \end{bmatrix}, \quad \mathbf{f}_2 = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (E + p)v \end{bmatrix},$$

and  $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2]$ . The equations are closed by the ideal gas law as

$$p = (\gamma - 1) \left( E - \frac{1}{2} \rho (u^2 + v^2) \right), \quad c = \sqrt{\frac{\gamma p}{\rho}},$$

where  $c$  represents the local speed of sound and  $\gamma$  is a fluid-dependent constant. We take it to be  $\gamma = 7/5$ .

We develop the software in the usual fashion, with a driver routine that sets up the initial condition and defines the computational domain and the parameters of the computation, see EulerWENODriver2D.m.



**Figure 11.19.** Solution to the scalar conservation law with a nonconvex flux using a WENO scheme of linear order  $2m - 1$ . In the top left corner is a high-resolution reference solution.

**Script 11.23.** *EulerWENO2D.m*: Driver routine for the two-dimensional Euler equations solved using a WENO method.

---

```
% Driver script for solving the 2D Euler equations using a WENO scheme
clear all
```

```
% Order of method
m=2;
```

```
% Set problem parameters for isentropic vortex
% Lx = 10; Ly = 10; Nx = 64; Ny = 64; FinalTime = 1.0; CFL = 0.5; gamma = 1.4;
```

```

%  $hx = Lx/Nx$ ;  $hy = Ly/Ny$ ;

% Set up Legendre quadrature grid for cell average
%  $NGQ = 10$ ;  $[xGQ, wGQ] = \text{LegendreGQ}(NGQ)$ ;
%  $xQ1 = hx/2*xGQ$ ;  $yQ1 = hy/2*xGQ$ ;  $[xQ, yQ] = \text{meshgrid}(xQ1, yQ1)$ ;

% Define domain and initial conditions
%  $xv = [-5:hx:5]$ ;  $yv = [-5:hy:5]$ ;  $[x, y] = \text{meshgrid}(xv, yv)$ ;
%  $r = \text{zeros}(Ny+1, Nx+1)$ ;  $ru = \text{zeros}(Ny+1, Nx+1)$ ;
%  $rv = \text{zeros}(Ny+1, Nx+1)$ ;  $E = \text{zeros}(Ny+1, Nx+1)$ ;
%  $x0=0.0$ ;  $y0=0.0$ ;  $u0=1.0$ ;  $v0=0$ ;  $beta=5.0$ ;
% for  $j=1:Nx+1$ 
%   for  $i=1:Ny+1$ 
%      $xL = xQ + xv(j)$ ;  $yL = yQ + yv(i)$ ;
%      $[rL, ruL, rvL, EL] = \dots$ 
%     IsentropicVortex2D( $xL, x0, u0, yL, y0, v0, gamma, beta, 0.0$ );
%      $r(i, j) = wGQ^*rL*wGQ/4$ ;  $ru(i, j) = wGQ^*ruL*wGQ/4$ ;
%      $rv(i, j) = wGQ^*rvL*wGQ/4$ ;  $E(i, j) = wGQ^*EL*wGQ/4$ ;
%   end
% end

% Set problem parameters for 2D Riemann problems
Lx = 1; Ly = 1; Nx = 99; Ny = 99; CFL = 0.5; gamma = 1.4;
RiemannProbCase = 4;
hx = Lx/Nx; hy = Ly/Ny;

% Define domain and initial conditions
xv = [0:hx:Lx]; yv = [0:hy:Ly]; [x, y] = meshgrid(xv, yv);
[r, ru, rv, E, FinalTime] = Riemann2D(x, y, gamma, RiemannProbCase);

% Solve Problem
q = zeros(Ny+1, Nx+1, 4);
q(:,:,1) = r; q(:,:,2) = ru; q(:,:,3) = rv; q(:,:,4) = E;
[q] = EulerWENO2D(x, y, q, hx, hy, m, gamma, CFL, FinalTime);

```

The driver calls EulerWENO2D.m to perform the temporal integration of the problem by evaluating the right-hand side in EulerWENOrhs2D.m. Note that we apply the one-dimensional WENO reconstruction on each conserved variable. This could also be done on the characteristic variables as demonstrated for the one-dimensional case in subsection 11.3.4. Furthermore, the use of an ENO reconstruction or a different definition of the nonlinear WENO weights is straightforward.

**Script 11.24. EulerWENO2D.m: Integration routine for the two-dimensional Euler equations using a WENO method.**

```

function [q] = EulerWENO2D(x, y, q, hx, hy, m, gamma, CFL, FinalTime)
% function [q] = EulerWENO2D(x, y, q, hx, hy, m, gamma, CFL, FinalTime)
% Purpose : Integrate 2D Euler equation until FinalTime using a WENO scheme.

time = 0; tstep = 0;

% Initialize reconstruction weights
Crec = zeros(m+1, m);
for rr=-1:m-1;
    Crec(rr+2,:) = ReconstructWeights(m, rr);
end;

```

```

% Initialize linear weights
dw = LinearWeights(m,0);

% Compute smoothness indicator matrices
beta = zeros(m,m,m);
for rr=0:m-1
    xl = -1/2 + [-rr:1:m-rr];
    beta (:,:,rr+1) = betarcalc (xl ,m);
end

while (time<FinalTime)
    % Set timestep
    r = q (:,:,1); ru = q (:,:,2); rv = q (:,:,3); E = q (:,:,4);
    p = (gamma-1)*(E - 0.5*(ru.^2 + rv.^2)./r); c = sqrt(gamma*p./r);
    maxvelu = max(max(c+abs(ru./r))); maxvelv = max(max(c+abs(rv./r)));
    k = CFL*min(hx,hy)/sqrt(2)/sqrt(maxvelu.^2+maxvelv.^2);
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    [rhsq] = EulerWENOrhs2D(x,y,q,hx,hy,k,m,Crec,dw,beta,gamma);
    q1 = q + k*rhsq;
    [rhsq] = EulerWENOrhs2D(x,y,q1,hx,hy,k,m,Crec,dw,beta,gamma);
    q2 = (3*q + q1 + k*rhsq)/4;
    [rhsq] = EulerWENOrhs2D(x,y,q2,hx,hy,k,m,Crec,dw,beta,gamma);
    q = (q + 2*q2 + 2*k*rhsq)/3;
    time = time+k; tstep = tstep+1;
end
return

```

---

**Script 11.25. EulerWENOrhs2D.m: Definition of right-hand side for the two-dimensional Euler equations using a WENO reconstruction.**

---

```

function [dq] = EulerWENOrhs2D(x,y,q,hx,hy,k,m,Crec,dw,beta,gamma);
% function [dq] = EulerWENOrhs2D(x,y,q,hx,hy,k,m,Crec,dw,beta,gamma);
% Purpose: Evaluate right hand side for the two-dimensional Euler equations
%           using a WENO method
Nx = size(x); Nx = Nx(2); Ny = Nx(1); dq = zeros(Ny,Nx,4);

% Apply WENO in the x-direction
for i=1:Ny
    % Extend data and assign boundary conditions in x-direction
    [xe, re] = extend(x(i,:),q(i,:,1),hx,m,'N',0.0,'N',0.0);
    [xe, rue] = extend(x(i,:),q(i,:,2),hx,m,'N',0.0,'N',0.0);
    [xe, rve] = extend(x(i,:),q(i,:,3),hx,m,'N',0.0,'N',0.0);
    [xe, Ee] = extend(x(i,:),q(i,:,4),hx,m,'N',0.0,'N',0.0);

    % define cell left and right interface values
    ql = zeros(Nx+2,4); qr = zeros(Nx+2,4);
    for j=1:Nx+2
        [ql(j,1),qr(j,1)] = WENO(xe(j:(j+2*(m-1))), re(j:(j+2*(m-1))),m,Crec,dw,
        beta);
        [ql(j,2),qr(j,2)] = WENO(xe(j:(j+2*(m-1))),rue(j:(j+2*(m-1))),m,Crec,dw,
        beta);
        [ql(j,3),qr(j,3)] = WENO(xe(j:(j+2*(m-1))),rve(j:(j+2*(m-1))),m,Crec,dw,
        beta);
        [ql(j,4),qr(j,4)] = WENO(xe(j:(j+2*(m-1))), Ee(j:(j+2*(m-1))),m,Crec,dw,
        beta);
    end;

```

```

% Compute flux
[dq1] = EulerLF2Dx(qr(2:Nx+1,:), q1(3:Nx+2,:),gamma);
[dq2] = EulerLF2Dx(qr(1:Nx ,:), q1(2:Nx+1,:),gamma);

% Update residual
dq(i,:,:) = -(dq1-dq2)/hx;
end

% Apply WENO in the y-direction
for j=1:Nx
    % Extend data and assign boundary conditions in y-direction
    [ye, re] = extend(y(:,j),q(:,j,1),hy,m,'N',0.0,'N',0.0);
    [xe, rue] = extend(y(:,j),q(:,j,2),hy,m,'N',0.0,'N',0.0);
    [ye, rve] = extend(y(:,j),q(:,j,3),hy,m,'N',0.0,'N',0.0);
    [xe, Ee] = extend(y(:,j),q(:,j,4),hy,m,'N',0.0,'N',0.0);

    % define cell left and right interface values
    q1 = zeros(Ny+2,4); qr = zeros(Ny+2,4);
    for i=1:Ny+2
        [q1(i,1),qr(i,1)] = WENO(ye(i:(i+2*(m-1))), re(i:(i+2*(m-1))),m,Crec,dw,
        beta);
        [q1(i,2),qr(i,2)] = WENO(ye(i:(i+2*(m-1))),rue(i:(i+2*(m-1))),m,Crec,dw,
        beta);
        [q1(i,3),qr(i,3)] = WENO(ye(i:(i+2*(m-1))),rve(i:(i+2*(m-1))),m,Crec,dw,
        beta);
        [q1(i,4),qr(i,4)] = WENO(ye(i:(i+2*(m-1))), Ee(i:(i+2*(m-1))),m,Crec,dw,
        beta);
    end;

    % Compute flux
    [dq1] = EulerLF2Dy(qr(2:Ny+1,:), q1(3:Ny+2,:),gamma);
    [dq2] = EulerLF2Dy(qr(1:Ny ,:), q1(2:Ny+1,:),gamma);

    % Update residual
    dq(:,j,:)= dq(:,j,:)-(reshape(dq1,Ny,1,4) ...
        -reshape(dq2,Ny,1,4))/hy;
end
return

```

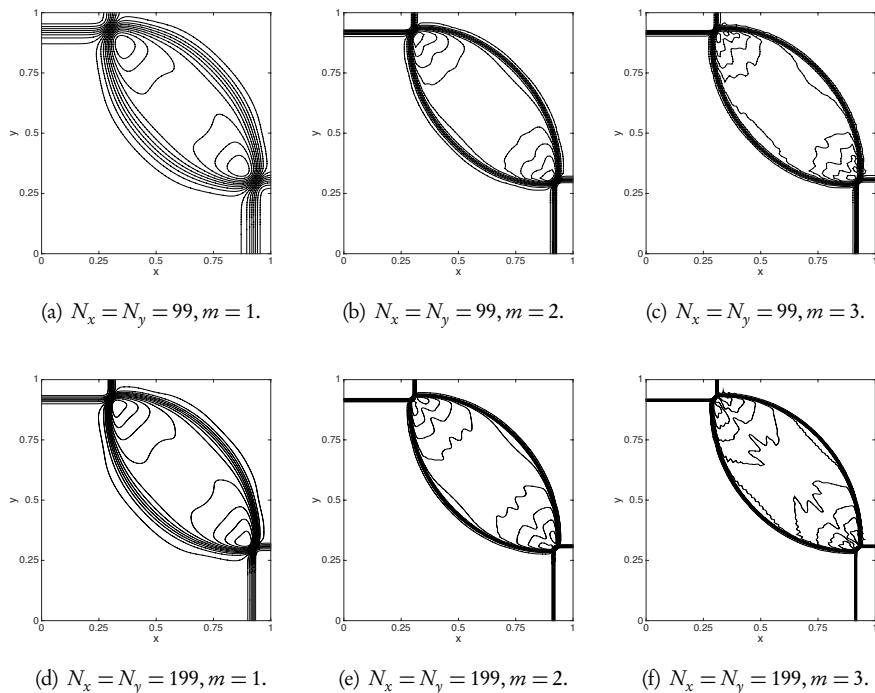
The details of the numerical flux, completing the scheme, are unchanged from the monotone scheme discussed in Chapter 7.

Our first smooth test case is the exact vortex solution. In Table 11.2 we list the  $L^1$ -error for the density at  $T = 0.01$  at different resolutions  $N_x = N_y$ , and orders  $m = 1$  or  $m = 2$ . As expected, we observe an  $\mathcal{O}(h^{2m-1})$ -error. We have used a local Lax-Friedrichs flux but, as we have seen previously, this choice does not impact the accuracy.

To further evaluate the performance of the WENO solver for the Euler equations, let us again consider the two-dimensional Riemann problems discussed in subsection 1.4.2.

**Table 11.2.** Errors for the isentropic vortex test of the two-dimensional Euler equations, solved using a WENO scheme of order  $\mathcal{O}(h^{2m-1})$ . The error, measured in  $\|\cdot\|_{h,1}$ , at  $T = 0.01$  of the density, is shown. A local Lax-Friedrichs flux is used.

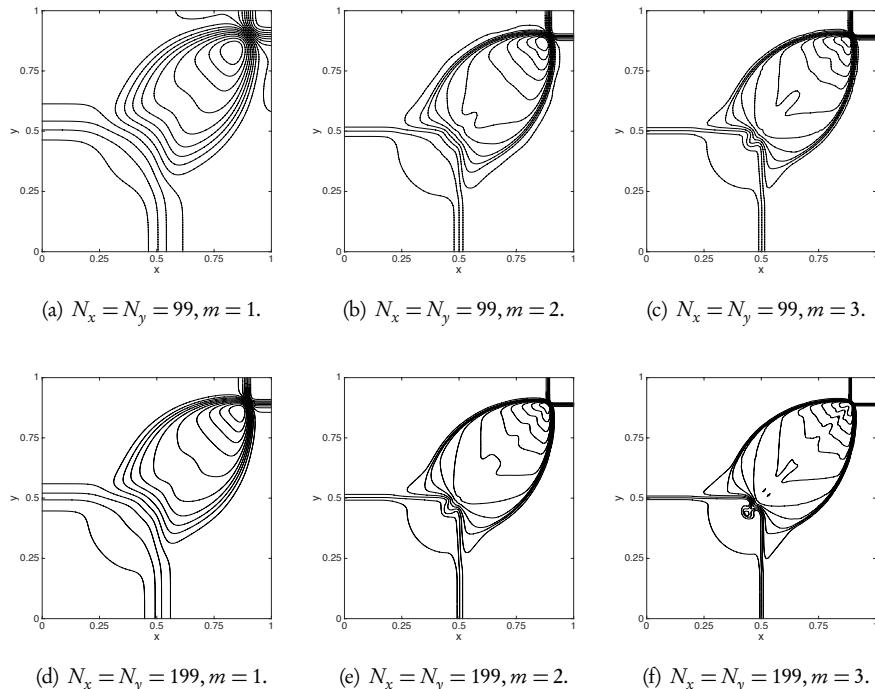
$N_x = N_y$	32	64	128	256	512	1024	Order
m=1	1.92e-1	9.94e-2	5.01e-2	2.51e-2	1.25e-2	6.26e-2	$\mathcal{O}(h)$
m=2	6.23e-3	2.22e-3	6.53e-4	1.67e-4	2.63e-5	2.33e-6	$\mathcal{O}(h^3)$



**Figure 11.20.** *Computed density of Riemann problem number 4, see [18], comprising four shocks, obtained using a WENO scheme of linear order  $2m - 1$  and  $N_x = N_y$  grid cells. A local Lax-Friedrichs flux is used in all cases. Compare to results in Fig. 7.3.*

We first consider the fourth test case, which comprises four interacting shocks. Figure 11.20 shows the computed density,  $\rho$ , at a final time  $T = 0.25$ , obtained with different resolutions and increasing orders. The results indicate convergence and substantial advantages of the higher-order accuracy of the WENO scheme in spite of the shocks. With respect to the results in Fig. 7.3, the first order monotone scheme with  $N_x = N_y = 399$  compares reasonably well to the results obtained with the third order ( $m = 2$ ) WENO scheme with  $N_x = N_y = 199$ . Similarly, the monotone scheme requires  $N_x = N_y = 799$  to obtain a result comparable to that of the fifth order ( $m = 3$ ) WENO scheme with only  $N_x = N_y = 199$ . The savings in the overall computational time are significant.

As a more challenging problem, we consider test case 12, which comprises two shocks and two contact waves. The results in Fig. 11.21 show, as expected, a substantially improved accuracy, in particular for the contact waves. This is consistent with previous results, and highlights the advantage of using higher-order accurate methods for wave-dominated problems. When comparing to the results in Fig. 7.3, it is clear that dissipation of the monotone scheme overwhelms the solution. The results obtained with a fifth order WENO scheme and  $N_x = N_y = 99$  show superior details as compared to the results obtained with a first order monotone scheme and  $N_x = N_y = 799$ .



**Figure 11.21.** Computed density of Riemann problem number 12, see [18], comprising four shocks, obtained using a WENO scheme of linear order  $2m-1$  and  $N_x = N_y$  grid cells. All figures show the density at  $T = 0.25$ . A local Lax-Friedrichs flux is used in all cases. Compare to results in Fig. 7.3.

## References

- [1] Rafael Borges, Monique Carmona, Bruno Costa, and Wai Sun Don. An improved weighted essentially non-oscillatory scheme for hyperbolic conservation laws. *Journal of Computational Physics*, 227(6):3191–3211, 2008.
- [2] Pawal Buchmuller and Christiane Helzel. Improved accuracy of high-order WENO finite volume methods on Cartesian grids. *Journal of Scientific Computing*, 61:343–368, 2014.
- [3] Guy Capdeville. A central WENO scheme for solving hyperbolic conservation laws on non-uniform grids. *Journal of Computational Physics*, 227:2977–3014, 2008.
- [4] I. Carvero, Gabriella Puppo, Matteo Semplice, and G Visconti. CWENO: Uniformly Accurate Reconstructions for Balance Laws. Preprint, arXiv:1607.07319, 2016.
- [5] Marcos Castro, Bruno Costa, and Wai Sun Don. High order weighted essentially non-oscillatory WENO-Z schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 230(5):1766–1792, 2011.

- [6] Philip J. Davis and Philip Rabinowitz. *Numerical Integration*. Blaisdell Publishing Company, 1967.
- [7] Michael Dumbser. *Arbitrary High Order Schemes for the Solution of Hyperbolic Conservation Laws in Complex Domains*. Shaker, 2005.
- [8] Ulrik S. Fjordholm, Siddhartha Mishra, and Eitan Tadmor. Arbitrarily high-order accurate entropy stable essentially nonoscillatory schemes for systems of conservation laws. *SIAM Journal on Numerical Analysis*, 50(2):544–573, 2012.
- [9] Ulrik S. Fjordholm, Siddhartha Mishra, and Eitan Tadmor. ENO reconstruction and ENO interpolation are stable. *Foundations of Computational Mathematics*, 13(2):139–159, 2013.
- [10] Ulrik S. Fjordholm and Deep Ray. A sign preserving WENO reconstruction method. *Journal of Scientific Computing*, 68(1):42–63, 2016.
- [11] Bengt Fornberg. Classroom note: Calculation of weights in finite difference formulas. *SIAM Review*, 40(3):685–691, 1998.
- [12] Gene H. Golub and John H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation*, 23(106):221–230, 1969.
- [13] Sigal Gottlieb, David Gottlieb, and Chi-Wang Shu. Recovering high-order accuracy in WENO computations of steady-state hyperbolic systems. *Journal of Scientific Computing*, 28(2-3):307–318, 2006.
- [14] Ami Harten, Bjorn Engquist, Stanley Osher, and Sukumar R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71(2):231–303, 1987.
- [15] Ami Harten, Stanley Osher, Björn Engquist, and Sukumar R. Chakravarthy. Some results on uniformly high-order accurate essentially nonoscillatory schemes. *Applied Numerical Mathematics*, 2(3):347–377, 1986.
- [16] Andrew K. Henrick, Tariq D. Aslam, and Joseph M. Powers. Mapped weighted essentially non-oscillatory schemes: Achieving optimal order near critical points. *Journal of Computational Physics*, 207(2):542–567, 2005.
- [17] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 1(126):202–228, 1996.
- [18] Alexander Kurganov and Eitan Tadmor. Solution of two-dimensional Riemann problems for gas dynamics without Riemann problem solvers. *Numerical Methods for Partial Differential Equations*, 18(5):584–608, 2002.
- [19] Doron Levy, Gabriella Puppo, and Giovanni Russo. Central WENO schemes for hyperbolic systems of conservation laws. *M2AN Mathematics Modeling and Numerical Analysis*, 33:547–571, 1999.
- [20] Doron Levy, Gabriella Puppo, and Giovanni Russo. Compact central WENO schemes for multidimensional conservation laws. *SIAM Journal of Scientific Computing*, 22(2):656–672, 2000.

- [21] Doron Levy, Gabriella Puppo, and Giovanni Russo. On the behavior of the total variation in CWENO methods for conservation laws. *Applied Numerical Mathematics*, 33:407–414, 2000.
- [22] Tingting Li, Chi-Wang Shu, and Mengping Zhang. Stability analysis of the inverse Lax-Wendroff boundary treatment for high order upwind-biased finite difference schemes. *Journal of Computational and Applied Mathematics*, 299:140–158, 2016.
- [23] Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115(1):200–212, 1994.
- [24] Bradley J. Lucier. Error bounds for the methods of Glimm, Godunov and LeVeque. *SIAM Journal on Numerical Analysis*, 22(6):1074–1081, 1985.
- [25] Barry Merriman. Understanding the Shu–Osher conservative finite difference form. *Journal of Scientific Computing*, 19(1–3):309–322, 2003.
- [26] Sebastian Noelle, Yulong Xing, and Chi-Wang Shu. High-order well-balanced schemes. In: *Numerical Methods for Relaxation Systems and Balance Equations*. Quaderni di Matematica, Dipartimento di Matematica, Seconda Universita di Napoli, 2009.
- [27] Jianxian Qiu and Chi-Wang Shu. On the construction, comparison, and local characteristic decomposition for high-order central WENO schemes. *Journal of Computational Physics*, 183(1):187–209, 2002.
- [28] A. M. Rogerson and Eckert Meiburg. A numerical study of the convergence properties of ENO schemes. *Journal of Scientific Computing*, 5(2):151–167, 1990.
- [29] Florin Sabac. The optimal convergence rate of monotone finite difference methods for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 34(6):2306–2318, 1997.
- [30] Chi-Wang Shu. Numerical experiments on the accuracy of ENO and modified ENO schemes. *Journal of Scientific Computing*, 5(2):127–149, 1990.
- [31] Chi-Wang Shu. *Essentially Non-Oscillatory and Weighted Essentially Non-Oscillatory Schemes for Hyperbolic Conservation Laws*, volume 1697 of Lecture Notes in Mathematics, pages 325–432. Springer, 1998.
- [32] Chi-Wang Shu. High order weighted essentially nonoscillatory schemes for convection dominated problems. *SIAM Review*, 51(1):82–126, 2009.
- [33] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.
- [34] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 83(1):32–78, 1989.
- [35] Gilbert Strang. Accurate partial difference methods I: Linear Cauchy problems. *Archive for Rational Mechanics and Analysis*, 12(1):392–402, 1963.
- [36] Gilbert Strang. Accurate partial difference methods. *Numerische Mathematik*, 6(1):37–46, 1964.

- [37] Sirui Tan and Chi-Wang Shu. Inverse Lax-Wendroff procedure for numerical boundary conditions of conservation laws. *Journal of Computational Physics*, 229(21):8144–8166, 2010.
- [38] Sirui Tan, Cheng Wang, Chi-Wang Shu, and Jianguo Ning. Efficient implementation of high order inverse Lax-Wendroff boundary treatment for conservation laws. *Journal of Computational Physics*, 231(6):2510–2527, 2012.
- [39] Vladimir A. Titarev and Eleuterio F. Toro. ADER schemes for three-dimensional non-linear hyperbolic systems. *Journal of Computational Physics*, 204(2):715–736, 2005.
- [40] Bart S. van Lith, Jan H. M. ten Thije Boonkkamp, and Wibert L. IJzerman. Embedded WENO: A design method to improve existing WENO schemes. *Journal of Computational Physics*, 330:529–549, 2017.
- [41] Yulong Xing and Chi-Wang Shu. High order well-balanced finite volume WENO schemes and discontinuous Galerkin methods for a class of hyperbolic systems with source terms. *Journal of Computational Physics*, 214:567–598, 2006.
- [42] Yulong Xing and Chi-Wang Shu. A survey of high order schemes for the shallow water equations. *Journal of Mathematical Study*, 47:221–249, 2014.
- [43] Yulong Xing, Chi-Wang Shu, and Sebastian Noelle. On the advantage of well-balanced schemes for moving-water equilibria of the shallow water equations. *Journal of Scientific Computing*, 48:339–349, 2011.
- [44] Hamed Zakerzadeh and Ulrik S. Fjordholm. High-order accurate, fully discrete entropy stable schemes for scalar conservation laws. *IMA Journal of Numerical Analysis*, 36:633–654, 2015.

## Chapter 12

# Discontinuous Galerkin methods

Having realized the benefits of high-order accurate schemes for conservation laws and the success of various high-order methods, one can wonder what else is needed. That is, are there shortcomings that suggest we continue to seek other methods?

While this question can be answered in different ways, the strongest argument in favor of the pursuit of new methods is to accommodate a more flexible grid structure. The motivation could be the need to solve problems which require a highly nonuniform, or even time-dependent grid, or a need to resolve general geometric features in two and three spatial dimensions.

Reflecting on the development of the high-order methods so far, one identifies a number of challenges that get in the way of reaching this goal. First of all, we observe that the higher-order accuracy is generally obtained by extending the computational stencil. While this does not introduce issues for simple geometries, it quickly emerges as a challenge when trying to model problems with more elaborate geometric features. Secondly, the stable and accurate treatment of the extended stencil close to boundaries is far from trivial, as discussed in subsection 11.5.1. Finally, the essentially nonoscillatory schemes are computationally expensive due to their inherent nonlinear nature, which requires an analysis of the local solution at every grid point, as discussed in the previous chapter. These challenges become particularly severe for multidimensional problems and for very high order methods where the reconstruction must be done on the characteristic variables.

As we have already discussed, essentially nonoscillatory schemes, expressed in the finite volume form, do allow the use of nonuniform grids and have also been extended to allow fully unstructured grids [2, 59, 97, 33]. Nevertheless, the complexity is considerable, and the computational cost remains high.

In our pursuit of an alternative to these otherwise very successful schemes, we focus on the need for increased geometric flexibility, while seeking to retain the desirable properties we have previously identified, e.g., accuracy and stability, even in the presence of discontinuous solutions.

Let us seek inspiration from the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad x \in \Omega, \quad (12.1)$$

subject to appropriate initial and boundary conditions on the boundary,  $\partial\Omega$ .

When formulating a numerical scheme, we need to consider the following two questions:

- How can we represent the solution  $u(x, t)$  by an approximate solution  $u_b(x, t)$  with a desired accuracy?
- In which way does  $u_b(x, t)$  satisfy the conservation laws?, i.e., if we define the residual

$$R_b(x, t) = \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x}, \quad (12.2)$$

in which sense do we require  $R_b$  to vanish ?

The answers to these two questions separate different methods and define their properties. Looking back at the methods we have discussed so far, we realize that they typically use a piecewise polynomial representation  $u_b$  of the solution  $u$ , based on point values at the cell centers, as in the finite difference methods, or on the cell averages, as in the finite volume methods. The finite difference methods are obtained by requiring that the residual vanishes at the cell center, whereas the finite volume methods arise by requiring that the cell-averaged residual vanishes identically in every cell.

An appealing aspect of this point of view is its simplicity, i.e., the spatial discretization of general problems is often intuitive and, in many cases, leads to efficient and robust schemes. Furthermore, the explicit semidiscrete form ensures flexibility in the choice of time integration methods. Finally, we have an extensive theory to yield confidence in the results, obtained with such methods.

As we have discussed at length, both the finite difference and the finite volume methods do allow for higher-order accuracy by extending the stencil. While this is conceptionally simple, this reliance on the extended one-dimensional polynomial reconstruction is also the Achilles' heel of these methods, as it enforces a dimension-by-dimension structure in higher dimensions. The most prominent consequence is the complexity associated with problems posed on general geometries or with problems where a nonuniform grid is needed.

It appears evident that, in order to ensure geometric flexibility we must abandon this inherent one-dimensional construction of the scheme in favor of something more general. The most natural approach is to introduce an element-based discretization, where we assume that  $\Omega$  is represented by a collection of  $N$  elements  $D_j$  such that

$$\Omega \simeq \Omega_b = \bigcup_{j=1}^N D_j.$$

In the one-dimensional case,  $D_j$  is simply a line segment, but in higher dimensions it can be a simplex, a cube, or a general polygon. The only condition is that the elements, or a combination of these, can tile the computational domain  $\Omega_b$  in a nonoverlapping fashion.

The lack of locality, as in the extended stencil in high-order finite difference and finite volume methods, is a source of concern in our quest for high-order methods with grid flexibility. Thus, we pursue a different approach and increase the degrees of freedom locally on each element. While this eliminates the cell average as the primary variable, this approach offers other advantages, as we shall soon realize.

To pursue this idea, consider the one-dimensional case and define the element  $D_j$  as the interval  $D_j = [x_{j-1/2}, x_{j+1/2}] = [x_j - b_j/2, x_j + b_j/2]$ , where  $x_j$  is the cell center of  $D_j$  and  $b_j$  is the local cell size. We assume that the global approximation is expressed as

$$u_b(x, t) = \sum_{j=1}^N u_j(t) Q_j(x),$$

where  $Q_j$  are known as basis or trial functions. The simplest choice is the piecewise linear shape function,

$$Q_j(x) = \begin{cases} 0, & x \leq x_{j-1}, \\ 1 + \frac{1}{b_j}(x - x_j), & x_{j-1} \leq x \leq x_j, \\ 1 - \frac{1}{b_j}(x - x_j), & x_j \leq x \leq x_{j+1}, \\ 0, & x \geq x_{j+1}, \end{cases} \quad (12.3)$$

with the property  $Q_j(x_i) = \delta_{ij}$ .

To construct a scheme for (12.1), we define a space of test functions  $V_b$ , spanned by the test functions  $\phi_b(x)$ , and require the residual to be  $L^2$ -orthogonal to all test functions:

$$\int_{\Omega} \left( \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x} \right) \phi_b(x) dx = 0 \quad \forall \phi_b \in V_b.$$

The details of the scheme depend on how  $V_b$  is defined. A classic choice, leading to a Galerkin scheme, is to require that the two spaces spanned by the basis functions and test functions, respectively, are the same. If we require that the residual vanishes for all  $\phi_b \in V_b = \text{span} \{Q_j(x)\}_{j=1}^N$ , we recover

$$\int_{\Omega} \left( \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x} \right) Q_j(x) dx = 0 \quad \forall j = 1, \dots, N.$$

If we furthermore approximate the flux as

$$f_b(x, t) = \sum_{j=1}^N f(x_j) Q_j(x),$$

and define the global mass matrix  $M$  and the stiffness matrix  $S$ , as

$$M_{ij} = \int_{\Omega} Q_i(x) Q_j(x) dx, \quad S_{ij} = \int_{\Omega} Q_i(x) \frac{dQ_j}{dx} dx,$$

we recover the scheme

$$M \frac{d\mathbf{u}_b}{dt} + S \mathbf{f}_b = 0. \quad (12.4)$$

Here we have introduced the vector of the unknowns  $\mathbf{u}_b = [u_1, \dots, u_N]^T$ , and the fluxes  $\mathbf{f}_b = [f_1, \dots, f_N]^T$ .

This approach, resulting in the classic finite element method [63, 100, 131, 130, 9, 35], trivially allows for elements of different sizes by defining the trial functions

(12.3) appropriately. The extension to higher-order elements can be achieved by adding additional degrees of freedom to the element, while maintaining shared nodes along the faces of the elements [69]. This can be expressed at the element level as high-order basis functions

$$x \in D_j : u_j(x, t) = \sum_{i=0}^m u_{j,i}(t) \psi_{j,i}(x), \quad (12.5)$$

where  $\psi_{j,i}(x)$  are  $m + 1$  basis functions defined on  $D_j$ .

While the classic finite element method appears to offer both geometric flexibility and the potential for high-order accuracy, the discussion also highlights some disadvantages. First, we see that the globally defined trial/basis functions (12.3), in combination with the Galerkin statement, results in the semidiscrete scheme (12.4) which requires that  $M$  be inverted, i.e., the scheme is implicit even in semidiscrete form. For time-dependent problems, this is a clear disadvantage when compared to the finite difference and finite volume methods we have discussed so far.

An additional issue is that the trial/basis functions are symmetric in space and depend solely on the grid. Hence, there is no straightforward way to exploit upwinding which may cause stability problems (see, e.g., [63, 131, 105]) when solving conservation laws.

We conclude that to ensure geometric flexibility and support for a locally adapted resolution, we need an element-based method where high-order accuracy is enabled through the local approximation (12.5), as in the finite element method. However, globally defined basis and test functions, combined with the global Galerkin statement on the residual, destroy the locality of the scheme and eliminate the opportunity to take advantage of upwinding.

This suggests that we seek a clever combination of the finite element and the finite volume method, utilizing a space of locally defined basis and test functions to mimic the finite element method, yet satisfying the equation in a sense closer to the spirit of the finite volume method. This combination is precisely what leads to the discontinuous Galerkin method.

To recover such a scheme, we maintain the notion of elements as in the finite element scheme

$$\Omega = \bigcup_{j=1}^N D_j, \quad D_j = [x_{j-1/2}, x_{j+1/2}],$$

but introduce variables located at all the cell interfaces  $x_{j\pm 1/2}$  to enable locality. Hence, the global vector of unknowns becomes

$$\mathbf{u}_b = [\underbrace{u_{1/2}, u_{3/2}, \dots, u_{N-1/2}}_{D_1}, \underbrace{u_{3/2}, u_{5/2}, \dots, u_{N-1/2}}_{D_2}, \dots, \underbrace{u_{N-1/2}, u_{N+1/2}}_{D_N}]^T,$$

i.e., we have  $2N$  entries as opposed to  $N + 1$  in the finite element method. Within each element we now assume that the local solution and the flux can be expressed as

$$x \in D_j : u_j(x, t) = \sum_{i=0}^m \hat{u}_{j,i}(t) \psi_{j,i}(x), \quad f_j(x, t) = \sum_{i=0}^m \hat{f}_{j,i}(t) \psi_{j,i}(x),$$

where  $\psi_{j,i}$  span the piecewise polynomial solution space  $V_b = \text{span} \{ \psi_{j,i} \}_{(j,i)=(1,0)}^{(N,m)}$ . This representation is local and imposes no restrictions on the smoothness of the basis

functions between the elements. As a consequence, the solution is only piecewise smooth.

Following the formulation of the finite element scheme, we assume that the local solution can be well represented by  $u_h \in V_h$  and form the residual (12.2). We require this to be  $L^2$ -orthogonal to the piecewise polynomial test space to recover the  $N$  local statements

$$\int_{D_j} R_h(x, t) \psi_{j,i} dx = 0 \quad \forall i : 0 \dots m, \quad \forall j : 1 \dots N, \quad (12.6)$$

for all the test functions  $\psi_{j,i}$ . The local nature is a direct consequence of  $V_h$  being a broken space. Note that we need not use the same space of test and trial functions, which would result in a Petrov–Galerkin scheme. We shall return to this in section 12.3, and restrict ourselves to the pure Galerkin formulation until then.

At first glance, the locality appears problematic, as it is not obvious how to recover a global solution from the  $N$  local solutions. Furthermore, since the cell interfaces are shared by two elements, the question arises of how to ensure uniqueness of the solution at these points.

A simple solution is suggested by observing that if we take the local basis to be  $\psi_{j,0} = 1/h_j$ , the scheme reduces to a finite volume method for which we have developed the theory of Riemann problems and numerical fluxes to address this exact situation at the cell interfaces. Following this line of thinking, integration by parts recovers

$$\begin{aligned} \int_{D_j} \left( \frac{\partial u_j}{\partial t} \psi_{j,i} - f_j \frac{d\psi_{j,i}}{dx} \right) dx &= - \left[ f_j \psi_{j,i} \right]_{x_{j-1/2}}^{x_{j+1/2}} \\ &= - \left( f_j(x_{j+1/2}) \psi_{j,i}(x_{j+1/2}) - f_j(x_{j-1/2}) \psi_{j,i}(x_{j-1/2}) \right), \end{aligned}$$

for all  $(j, i)$ . As for the finite volume method at each interface,  $x_{j\pm 1/2}$ , we are left with a (generalized) Riemann problem to solve. We introduce the numerical flux  $F_{j\pm 1/2} = f(u_{j\pm 1/2}^*)$  to recover

$$\int_{D_j} \left( \frac{\partial u_j}{\partial t} \psi_{j,i} - f_j \frac{d\psi_{j,i}}{dx} \right) dx = - \left( F_{j+1/2} \psi_{j,i}(x_{j+1/2}) - F_{j-1/2} \psi_{j,i}(x_{j-1/2}) \right).$$

This is the discontinuous Galerkin method stated in weak form. Integration by parts once again yields the strong form

$$\begin{aligned} \int_{D_j} R_h(x, t) \psi_{j,i}(x) dx &= \left( f(u(x_{j+1/2})) - F_{j+1/2} \right) \psi_{j,i}(x_{j+1/2}) \\ &\quad - \left( f(u(x_{j-1/2})) - F_{j-1/2} \right) \psi_{j,i}(x_{j-1/2}), \end{aligned} \quad (12.7)$$

for all  $(j, i)$ . It is clear that, as for finite volume methods, care in the choice of the numerical flux  $F_{j\pm 1/2}$  is critical.

To mimic the terminology of the finite element scheme, we express the scheme at each cell as

$$M_j \frac{d\mathbf{u}_j}{dt} - (S_j)^T \mathbf{f}_j = -F_{j+1/2} \psi_j(x_{j+1/2}) + F_{j-1/2} \psi_j(x_{j-1/2})$$

and

$$\mathbf{M}_j \frac{d\mathbf{u}_j}{dt} - \mathbf{S}_j \mathbf{f}_j = (f(u(x_{j+1/2})) - F_{j+1/2}) \psi_j(x_{j+1/2}) - (f(u(x_{j-1/2})) - F_{j-1/2}) \psi_j(x_{j-1/2}),$$

where we expose the local dependency of the mass and stiffness matrix on the element  $D_j$ . Here  $\mathbf{u}_j$  is the vector of local unknowns and  $\mathbf{f}_j$  is the corresponding vector of fluxes. Furthermore, we set  $\psi_j(x) = [\psi_{j,0}(x), \dots, \psi_{j,m}(x)]^T$  and define the local matrices

$$(\mathbf{M}_j)_{kl} = \int_{D_j} \psi_{j,k}(x) \psi_{j,l}(x) dx, \quad (\mathbf{S}_j)_{kl} = \int_{D_j} \psi_{j,k}(x) \frac{d\psi_{j,l}}{dx} dx. \quad (12.8)$$

While the structure of the discontinuous Galerkin method is very similar to that of the finite element method, there are essential differences. In particular, the mass matrix is local rather than global and can be inverted at low cost, resulting in an explicit semidiscrete scheme. Furthermore, by having to choose the numerical flux, we recover the ability to exploit upwinding. Compared to the finite volume method, the discontinuous Galerkin method maintains locality at higher order by enriching the local basis.

However, all of this comes at a price—most notably an increase in the total number of degrees of freedom, resulting from the decoupling of the elements. For linear elements ( $m = 1$ ), this yields a doubling in the total number of degrees of freedom when compared to the continuous finite element method. If we consider a  $d$ -dimensional simplex as the element and an order of approximation  $m$ , the storage penalty becomes

$$S_{m,d} = \frac{m+d}{m - \frac{1}{2}d(d-1)},$$

assuming  $m > d$ . Thus, when increasing the order of approximation at fixed  $d$ , the penalty quickly approaches one, reducing the impact of the overhead. On the other hand, for  $m \leq d$  all degrees of freedom are associated with edges, which results in a doubling of the degrees of freedom.

The discontinuous Galerkin method appears to have been proposed first in [92] as a way to solve the steady-state neutron transport equation

$$\sigma u + \nabla \cdot (a u) = f,$$

where  $\sigma$  is a real constant,  $a(x)$  is piecewise constant, and  $u$  is the unknown. The first analysis of the method was presented in [77, 68, 87]. A general extension to the hyperbolic systems and their discretization is given in [10, 34, 35].

The extension to the nonlinear scalar conservation law was first attempted in [20] by means of linear elements and a forward Euler method in time, resulting in an unstable method unless a very restrictive time step is used. In [19], this was resolved by adding a slope limiter, yielding a convergent scheme, albeit of low order in both space and time. This was improved in [28] by combining the strongly stable Runge–Kutta methods, discussed in Chapter 9, with an improved slope limiter [98] to ensure formal high-order accuracy in smooth regions and sharp shock resolution. By further extending this approach through the introduction of a generalized slope limiter and high-order strongly stable Runge–Kutta methods, these techniques were extended to high-order

accuracy in [27], to one-dimensional systems [26], to multidimensional scalar problems [25], and finally to systems [30]. Numerous applications and further details concerning these methods can be found by consulting recent textbooks [56, 94, 78, 35, 69], which offer different introductions to this active research area.

Before we embark on a more rigorous discussion of the discontinuous Galerkin methods, we need to introduce some additional notation. In particular, both global norms, defined on  $\Omega$ , and broken norms, defined over  $\Omega_h$  as sums of  $N$  elements  $D_j$ , will be needed. To keep things simple, we focus the discussion on the one-dimensional case. However, unless otherwise stated, all results can be extended to multidimensional problems. For a detailed discussion of this, we refer to [56].

For the solution  $u(x, t)$  we define the global  $L^2$ -norm over the domain  $\Omega$  as

$$(u, v)_\Omega = \int_\Omega u v dx \quad (u, v)_\Omega = \|u\|_\Omega^2,$$

and the broken norms

$$\|u\|_h^2 = \sum_{j=1}^N \|u_j\|_{D_j}^2, \quad \|u_j\|_{D_j}^2 = \int_{D_j} u_j^2 dx.$$

In a similar way, we define the associated Sobolev norms

$$\|u\|_q^2 = \sum_{l=0}^q \|u^{(l)}\|_\Omega^2, \quad \|u\|_{h,q}^2 = \sum_{j=1}^N \|u_j\|_{D_j,q}^2, \quad \|u_j\|_{D_j,q}^2 = \sum_{l=0}^q \|u_j^{(l)}\|_{D_j}^2.$$

As usual,  $u^{(l)}$  refers to the  $l$ th derivative of  $u$ .

We will also define  $H^q$  as the space of functions for which  $\|u\|_{\Omega,q}$  or  $\|u\|_{\Omega,q,h}$  is bounded. Finally, we will need the seminorms

$$|u|_{h,q}^2 = \sum_{j=1}^N |u_j|_{D_j,q}^2, \quad |u|_{D_j,q}^2 = \|u_j^{(q)}\|_{D_j}^2.$$

## 12.1 ■ The basics

Let us now take a more detailed look at the discontinuous Galerkin method. Consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to appropriate initial and boundary conditions. To formulate the scheme, we assume that the global solution  $u_h(x, t) \in V_h$  is expressed as

$$u(x, t) \simeq u_h(x, t) = \bigoplus_{j=1}^N u_j(x, t),$$

where  $u_j$  represents the local solution on element  $D_j$  with  $x \in [x_{j-1/2}, x_{j+1/2}]$ . We further assume that

$$u_j(x, t) = \sum_{i=0}^m u_{j,i}(t) \psi_{j,i}(x),$$

where  $\psi_{j,i}$  is a family of suitably chosen basis functions defined locally on  $D_j$ . We define the space of test functions  $V_h = \text{span} \{ \psi_{j,i}(x) \}_{(j,i)=(1,0)}^{(N,m)}$ , and require that the residual is  $L^2$ -orthogonal to all test functions  $v_h \in V_h$  as

$$\int_{D_j} \left( \frac{\partial u_j}{\partial t} \psi_{j,i} - f_j \frac{d\psi_{j,i}}{dx} \right) dx = - \left( F_{j+1/2} \psi_{j,i}(x_{j+1/2}) - F_{j-1/2} \psi_{j,i}(x_{j-1/2}) \right),$$

for  $j = 1 \dots N, i = 0 \dots m$ , where we, after integration by parts, introduce the numerical flux to recover a global scheme.

While the formulation of the scheme is relatively straightforward, it leaves open a number of questions related to the choice of the basis functions  $\psi_{j,i}$ , the choice of the numerical flux  $F_{j\pm 1/2}$ , and the characteristics of the scheme in terms of accuracy and stability. To gain some intuition for these aspects, let us consider an example.

**Example 12.1.** Consider the linear wave equation

$$\frac{\partial u}{\partial t} - 2\pi \frac{\partial u}{\partial x} = 0, \quad x \in [0, 2\pi],$$

with periodic boundary conditions and an initial condition defined as

$$u(x, 0) = \sin(lx), \quad l = \frac{2\pi}{\lambda}.$$

Here  $\lambda$  is the wavelength.

We use polynomials up to order  $m$ , defined locally on the element, as basis functions, and choose the numerical flux for  $f(u) = au$  as

$$F(u, v) = a \frac{u + v}{2} - |a| \frac{1 - \alpha}{2} (v - u).$$

Here  $0 \leq \alpha \leq 1$  is a free parameter. The choice of  $\alpha = 0$  corresponds to upwinding, while  $\alpha = 1$  yields a central flux.

The solutions obtained with these two different fluxes are shown in Fig. 12.1, highlighting the discontinuous nature of the solution. The dissipative nature of the upwind flux is also apparent.

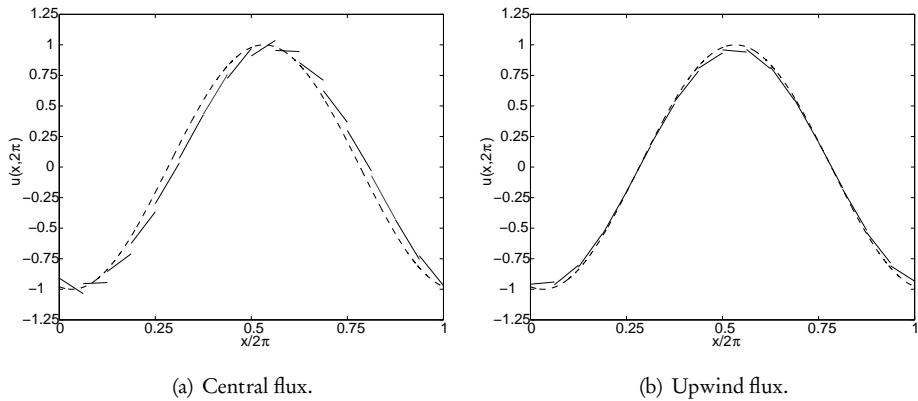
To further evaluate the performance of the scheme, we test its accuracy for different values of  $m$  and  $N$ , using a high-order Runge–Kutta method to integrate in time. We choose a time step small enough to ensure that temporal errors are negligible.

Table 12.1 lists the global  $L^2$ -error at  $T = \pi$  as a function of the number of elements  $N$ , and the order of the local approximation  $m$ . First of all, the scheme is clearly convergent and the results show that there are two paths to convergence—one can increase the local order of approximation  $m$  or one can increase the number of elements  $N$ .

However, the rate of convergence is not the same when changing  $m$  and  $N$ . If we define  $h = 2\pi/N$  as the cell size we observe that

$$\|u - u_h\|_h \leq Ch^{m+1}.$$

Thus, it appears that the order of the local approximation determines the fast convergence rate. The constant,  $C$ , is independent of  $h$ , but it may depend on the final time  $T$  of the solution. ■



**Figure 12.1.** Solution of the wave equation, computed with  $m = 1$  and  $N = 12$  elements, using a discontinuous Galerkin method with a central flux (a) and an upwind flux (b). In both cases, the dashed line represents the exact solution.

**Table 12.1.** Global  $L^2$ -errors for the solution of the wave equation with a discontinuous Galerkin method with  $N$  elements, and a local order of approximation,  $m$ . For  $m = 8$ , finite precision effects dominate and destroy the expected convergence rate.

$m \setminus N$	2	4	8	16	32	64	Convergence rate
1	–	4.0E-01	9.1E-02	2.3E-02	5.7E-03	1.4E-03	2.0
2	2.0E-01	4.3E-02	6.3E-03	8.0E-04	1.0E-04	1.3E-05	3.0
4	3.3E-03	3.1E-04	9.9E-06	3.2E-07	1.0E-08	3.3E-10	5.0
8	2.1E-07	2.5E-09	4.8E-12	2.2E-13	5.0E-13	6.6E-13	$\simeq 9.0$

In what follows we dive a bit deeper into these observations and explore their generality.

### 12.1.1 • The local approximation

We assume that the local solution is expressed as

$$x \in D_j = [x_{j-1/2}, x_{j+1/2}] : u_j(x, t) = \sum_{i=0}^m u_{j,i}(t) \psi_{j,i}(x).$$

While one, in principle, could choose any  $L^2$ -complete family of basis functions  $\psi_{j,i}$ , a polynomial family remains the standard choice and we shall restrict ourselves to this choice. The results of Ex. 12.1 illustrate that the accuracy of the method is closely related to the order of the local polynomial representation.

Let us begin by introducing the affine mapping

$$x \in \mathcal{D}_j : x(r) = x_{j-1/2} + \frac{1+r}{2} b_j, \quad b_j = x_{j+1/2} - x_{j-1/2}, \quad (12.9)$$

where  $r \in \mathcal{I} = [-1, 1]$  is the reference variable and  $b_j$  is the local cell size.

### The modal representation

We first consider a local polynomial representation of the form

$$x \in \mathcal{D}_j : u_j(x(r), t) = \sum_{i=0}^m \hat{u}_{j,i}(t) \psi_{j,i}(r),$$

known as the modal representation. To simplify the notation we drop the explicit reference to the element and to time and consider

$$u_b(r) = \sum_{i=0}^m \hat{u}_i \psi_i(r).$$

As a first choice, we could consider a monomial basis,  $\psi_i(r) = r^i$ . This leaves only the question of how to recover the coefficients  $\hat{u}_i$ . A natural way is by means of an  $L^2$ -projection,

$$(\mathbf{u}(r), \psi_j(r))_{\mathcal{I}} = \sum_{i=0}^m \hat{u}_i (\psi_i(r), \psi_j(r))_{\mathcal{I}}, \quad j = 0 \dots m, \quad (12.10)$$

which yields

$$\hat{\mathbf{M}} \hat{\mathbf{u}} = \mathbf{u}, \quad (12.11)$$

where

$$\hat{\mathbf{M}}_{ij} = (\psi_i, \psi_j)_{\mathcal{I}}, \quad \hat{\mathbf{u}} = [\hat{u}_0, \dots, \hat{u}_m]^T, \quad \mathbf{u} = [u_0, \dots, u_m]^T, \quad u_j = (\mathbf{u}, \psi_j)_{\mathcal{I}}.$$

Thus, we recover  $m+1$  equations for the  $m+1$  unknown expansion coefficients  $\hat{u}_i$ .

However, with the monomial basis we have

$$\hat{\mathbf{M}}_{ij} = \frac{1}{i+j+1} [1 + (-1)^{i+j}], \quad (12.12)$$

which resembles a Hilbert matrix. For increasing values of  $m$ , this matrix becomes very ill-conditioned. As a result, even for moderate orders, e.g.,  $m > 10$ , one cannot accurately compute  $\hat{\mathbf{u}}$  to obtain a polynomial representation  $u_b$ .

A natural solution to this problem is to seek an orthonormal basis. This can easily be achieved through an  $L^2$ -based Gram-Schmidt orthogonalization of the monomial basis, resulting in the orthonormal basis [103]

$$\psi_i(r) = P_i(r) = \frac{\tilde{P}_i(r)}{\sqrt{\gamma_i}},$$

where  $\tilde{P}_i(r)$  is the classic Legendre polynomial of order  $i$  and

$$\gamma_i = \frac{2}{2i+1}$$

is a normalization. This new basis can be evaluated through the recurrence relation

$$rP_i(r) = \alpha_i P_{i-1}(r) + \alpha_{i+1} P_{i+1}(r), \quad \alpha_i = \sqrt{\frac{i^2}{(2i+1)(2i-1)}},$$

with the initial values

$$P_0(r) = \frac{1}{\sqrt{2}}, \quad P_1(r) = \sqrt{\frac{3}{2}}r.$$

The evaluation is illustrated in `LegendreP.m`.

**Script 12.1. *LegendreP.m*: Evaluation of orthonormal Legendre polynomials of order  $m$  at  $x$ .**

---

```
function [P] = LegendreP(x,m);
% function [P] = LegendreP(x,m)
% Purpose: Evaluate orthonormal m'th order Legendre Polynomial at point x
xp = x; dims = size(xp); if (dims(2)==1) xp = xp'; end;

% Initial values P_0(x) and P_1(x)
PL = zeros(m+1,length(xp));
PL(1,:) = sqrt(1.0/2.0); if (m==0) P=PL'; return; end;
PL(2,:) = sqrt(3.0/2.0)*xp; if (m==1) P=PL(m+1,:)' ; return; end;

% Forward recurrence using the symmetry of the recurrence.
aold = sqrt(1.0/3.0);
for i=1:m-1
    anew = 2/(2*i+2)*sqrt((i+1)*(i+1)*(i+1)*(i+1)/(2*i+1)/(2*i+3));
    PL(i+2,:) = 1/anew*(-aold*PL(i,:) + xp.*PL(i+1,:));
    aold = anew;
end;
P = PL(m+1,:)';
return
```

---

The first few Legendre polynomials are illustrated in Fig. 12.2.

The  $L^2$ -orthogonality of the basis implies that the mass matrix  $\hat{M}$  is an identity, thus eliminating the issue of conditioning in (12.11). The question of how to compute  $u_i$ , given as

$$u_i = (u, \psi_i)_I = (u, P_i)_I = \hat{u}_i$$

remains open. The last equality follows from  $\hat{M}$  being the identity. For a general function  $u$ , the evaluation of this inner product is nontrivial, and the only practical way is to evaluate the integral numerically. As discussed in subsection 11.2.3, it is natural to use a Gaussian quadrature of the form

$$\hat{u}_i \simeq \sum_{j=0}^m u(r_j) P_i(r_j) w_j,$$

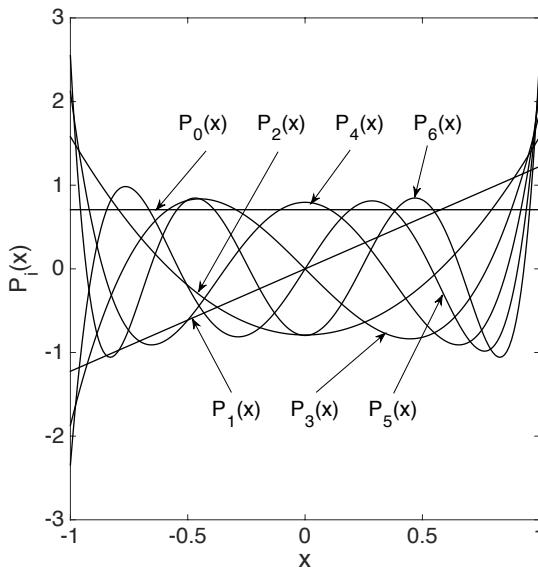


Figure 12.2. Illustration of the first six orthonormal Legendre polynomials.

where  $r_j \in [-1, 1]$  are the quadrature points and  $w_j$  are the quadrature weights. An important property of such a quadrature is that it is exact for polynomials of order  $2m + 1$  [32]. The weights and nodes for the Gaussian quadrature can be computed using LegendreGQ.m, discussed in subsection 11.2.3.

Equipped with the orthonormal basis we can express the stiffness matrix  $\hat{S}$ , with entries defined in (12.8). To accomplish this, the property [43]

$$\frac{d}{dr} P_j(r) = \sum_{\substack{i=0 \\ i+j \text{ odd}}}^{j-1} \sqrt{(2i+1)(2j+1)} P_i(r),$$

allows us to recover

$$\hat{S}_{ij} = \int_{-1}^1 P_i(r) \frac{d}{dr} P_j(r) dr = \begin{cases} \sqrt{(2i+1)(2j+1)}, & (i, j) \geq 0; i + j \text{ odd}, \\ 0, & \text{otherwise.} \end{cases}$$

Before continuing, it is worth understanding the error associated with the polynomial approximation. As we experienced in Ex. 12.1, we expect this to be closely related to the overall accuracy of the discontinuous Galerkin method.

We first estimate what can be expected under order enrichment. Let us, for simplicity, assume that all elements have length  $h$  and introduce the new variable

$$v(r) = u\left(\frac{h}{2}r\right) = u(x).$$

We now consider the question of how well

$$v_h(r) = \sum_{i=0}^m \hat{v}_i P_i(r)$$

approximates  $v \in L^2(\mathbb{I})$ . We begin by exploiting the orthonormality of  $P_i$  to recover  $\hat{v}_i$  as

$$\hat{v}_i = \int_{\mathbb{I}} v(r) P_i(r) dr. \quad (12.13)$$

An immediate consequence of the orthonormality of the basis is

$$\|v - v_b\|_{\mathbb{I}}^2 = \sum_{i=m+1}^{\infty} |\hat{v}_i|^2, \quad (12.14)$$

from which a basic approximation result follows [13].

**Theorem 12.2.** *Let  $v \in H^p(\mathbb{I})$  and let  $v_b$  represent a polynomial projection of  $v$  of order  $m$ . Then, for each integer  $q$  such that  $0 \leq q \leq p$ , it holds*

$$\|v - v_b\|_{\mathbb{I},q} \leq m^{\rho-p} \|v\|_{\mathbb{I},p},$$

where

$$\rho = \begin{cases} \frac{3}{2}q, & 0 \leq q \leq 1, \\ 2q - \frac{1}{2}, & q \geq 1. \end{cases}$$

**Proof.** We shall only sketch the proof and refer to [13] for full details. The Legendre polynomials satisfy the singular Sturm–Liouville problem

$$\frac{d}{dr}(1-r^2) \frac{d}{dr} P_i(r) + i(i+1)P_i(r) = \mathcal{L}P_i(r) + \lambda_i P_i(r) = 0. \quad (12.15)$$

For ease of notation, we define

$$\mathcal{L} = \frac{d}{dr}(1-r^2) \frac{d}{dr}, \quad \lambda_i = i(i+1).$$

Recalling the definition (12.13) of  $\hat{v}_i$ , we use (12.15) and integration by parts  $2p$  times to recover

$$|\hat{v}_i| \leq \lambda_i^{-p} \int_{\mathbb{I}} (\mathcal{L}^p v(r)) P_i(r) dr.$$

Combining this with Parseval's identity yields the  $L^2$ -estimate

$$\|v - v_b\|_{\mathbb{I},0} \leq m^{-p} \|v\|_{\mathbb{I},p},$$

for  $v \in H_p(\mathbb{I})$ ,  $p \geq 0$ , hence establishing the result for  $q = 0$ .

An intuitive understanding of the bound in the higher norms can be obtained by the classic inverse inequality [13, 103]

$$\left\| \frac{dP_i(r)}{dr} \right\|_{\mathbb{I},0} \leq i^2 \|P_i(r)\|_{\mathbb{I},0}.$$

Hence, for  $q \geq 1$  one expects to lose two orders in the order of convergence for each derivative, as reflected in the theorem. The more delicate estimate for  $0 \leq q \leq 1$  in the theorem relies on a careful analysis of the behavior of the polynomials.  $\square$

If the local solution is smooth, i.e.,  $v \in H^p(\mathbb{I})$  for  $p$  large, the result suggests that convergence is very fast, and exponential for an analytic function [104]. Thus, we recover the trademark of a classic spectral method [53], where the error decays exponentially fast with increasing  $m$ . We return to this case in more detail in Chapter 13.

We need a slightly refined estimate to gain a deeper insight into the convergence behavior. First recall the following lemma [95].

**Lemma 12.3.** *Let  $v \in H^p(\mathbb{I})$ ,  $p \geq 0$ , and assume that*

$$v(r) = \sum_{i=0}^{\infty} \hat{v}_i P_i(r).$$

*Then*

$$\int_{-1}^1 |v^{(q)}|^2 (1-r^2)^q dr = \sum_{i \geq q} |\hat{v}_i|^2 \frac{(i+q)!}{(i-q)!} \leq \|v\|_{\mathbb{I},q}^2,$$

*for  $0 \leq q \leq p$ .*

The proof relies on properties of the orthogonal polynomials; see [95] for the details. A consequence is the following lemma.

**Lemma 12.4.** *Let  $v \in H^p(\mathbb{I})$ ,  $p \geq 1$ . Then*

$$\|v - v_b\|_{\mathbb{I},0} \leq \left( \frac{(m+1-\sigma)!}{(m+1+\sigma)!} \right)^{1/2} \|v\|_{\mathbb{I},\sigma},$$

*where  $\sigma = \min(m+1, p)$ .*

**Proof.** By (12.14), we have

$$\|v - v_b\|_{\mathbb{I},0}^2 = \sum_{i=m+1}^{\infty} |\hat{v}_i|^2 = \sum_{i=m+1}^{\infty} |\hat{v}_i|^2 \frac{(i-\sigma)!}{(i+\sigma)!} \frac{(i+\sigma)!}{(i-\sigma)!}.$$

Provided  $\sigma \leq m+1$ , this implies

$$\|v - v_b\|_{\mathbb{I},0}^2 \leq \frac{(m+1-\sigma)!}{(m+1+\sigma)!} \sum_{i=m+1}^{\infty} |\hat{v}_i|^2 \frac{(i+\sigma)!}{(i-\sigma)!}.$$

Combined with Lemma 12.3 and  $\sigma = \min(m+1, p)$ , this yields the result.  $\square$

Let us now return to the behavior for the general element of length  $h$  to recover the estimates for  $u(x)$  rather than  $v(r)$ .

**Theorem 12.5.** *Let  $u \in H^p(D_j)$  and let  $u_b$  represent a piecewise polynomial approximation of  $v$  of order  $m$ . Then, for each integer  $q$  such that  $0 \leq q \leq \sigma = \min(m+1, p)$ , it holds that*

$$\|u - u_b\|_{b,q} \leq C h^{\sigma-q} |u|_{b,\sigma}.$$

*Proof.* We have

$$|v|_{l,q}^2 = \int_{\Gamma} (v^{(q)})^2 dr = C \int_{D_j} b^{2q-1} (u^{(q)})^2 dx = C b^{2q-1} |u|_{D_j,q}^2.$$

Similarly, we have

$$\|u\|_{D_j,q}^2 = \sum_{p=0}^q |u|_{D_j,p}^2 = C \sum_{p=0}^q b^{1-2p} |v|_{l,p}^2 \leq C b^{1-2q} |v|_{l,q}^2.$$

Combining these estimates we obtain

$$\|u - u_b\|_{D_j,q}^2 \leq C b^{1-2q} |v - v_b|_{l,q}^2 \leq C b^{1-2q} |v|_{l,\sigma}^2 = C b^{2\sigma-2q} |u|_{D_j,\sigma}^2,$$

where  $\sigma = \min(m+1, p)$ . Summation over all elements yields the final result.  $\square$

For the case of a more general grid with variable element length,  $h = \max_j h_j$  is used in the estimate. This result summarizes the basic approximation properties and highlights the conditions on  $u$  under which we can expect a high-order rate of convergence.

### The nodal representation

Let us consider a slightly different approach and define the expansion coefficients  $\tilde{v}_n$  such that the approximation is interpolatory:

$$v(r_j) = v_b(r_j) = \sum_{i=0}^m \tilde{v}_i P_i(r_j), \quad (12.16)$$

where  $r_j$  represents a set of  $m+1$  distinct grid points such that  $r_j \in [-1, 1]$ . We can express this in matrix form as

$$\hat{V} \tilde{v} = v,$$

where

$$\hat{V}_{ij} = P_j(r_i), \quad \tilde{v}_i = \tilde{v}_i, \quad v_i = v(r_i).$$

The matrix  $\hat{V}$ , recognized as a generalized Vandermonde matrix, shall play a central role in the developments that follow as it establishes a connection between the modes  $\tilde{v}$  and the nodal values  $v$ . As for the mass matrix, it is important to ensure that it is well conditioned.

Since the normalized Legendre polynomials comprise the optimal basis, we are left with the freedom to choose the grid points  $r_i$  to define the Vandermonde matrix.

We begin by first recognizing that if (12.16) is an interpolant, it can be expressed as

$$v_b(r) = \sum_{i=0}^m v(r_i) \ell_i(r), \quad \ell_i(r) = \prod_{\substack{j=0 \\ j \neq i}}^m \frac{r - r_j}{r_i - r_j},$$

where  $\ell_i$  is the interpolating Lagrange polynomial. To appreciate how the choice of these points relates to the conditioning of  $\hat{V}$ , observe that by uniqueness of the interpolation we have

$$\hat{V}^T \ell(r) = \mathbf{P}(r),$$

where  $\ell = [\ell_0(r), \dots, \ell_m(r)]^T$  and  $\mathbf{P}(r) = [P_0(r), \dots, P_m(r)]^T$ . Recalling Cramer's rule for solving linear systems of equations it is reasonable to seek  $r_i$  such that the determinant of  $\hat{V}$  is maximized. For this one-dimensional case, the answer to this is the  $m + 1$  zeros of [48, 53]

$$f(r) = (1 - r^2) P'_m(r).$$

These are closely related to the normalized Legendre polynomials and are known as the Legendre–Gauss–Lobatto (LGL) quadrature points, computable using `LegendreGL.m`, which also computes the associated quadrature weights [41]. This quadrature is exact for polynomials of degree  $2m - 1$ , but has the property that both end points of the integration domain are included as quadrature points [32].

---

**Script 12.2. *LegendreGL.m: Computation of the Legendre–Gauss–Lobatto quadrature nodes and weights of order  $m$ .***

---

```
function [x,w] = LegendreGL(m);
% function [x,w] = LegendreGL(m)
% Purpose: Compute the m'th order LGL quadrature points , x, and weights , w
x = zeros(m+1,1); w = zeros(m+1,1);
if (m==1) x(1)=-1.0; x(2)=1.0; w(1)=1.0; w(2)=1.0; return; end;
if (m==2)
    x(1)=-1.0; x(2)=0.0; x(3)=1.0; w(1)=1/3; w(2)=4/3; w(3)=1/3; return;
end;

% Form symmetric matrix from recurrence.
J = zeros(m-1); h1 = 2*(0:m-2)+2;
J = diag(2./((h1(1:m-2)+2).*sqrt((1:m-2).*((1:m-2)+2).*...
((1:m-2)+1).*((1:m-2)+1)./(h1(1:m-2)+1)./(h1(1:m-2)+3))),1);
J(1,1)=0.0; J = J + J';

%Compute quadrature by eigenvalue solve
[V,D] = eig(J); x = diag(D); x = [-1, x', 1]';
[P] = LegendreP(x,m); w = (2*m+1)/m/(m+1)./P.^2;
return;
```

---

While these points will serve us well going forward, there is nothing special about them and other choices are possible. For a discussion of this aspect we refer to [56].

By choosing the orthonormal Legendre basis and the LGL nodal points  $r_i$ , we ensure that  $\hat{V}$  is well-conditioned, and that the resulting interpolation is well-behaved. The initialization of  $\hat{V}$  is shown in `VandermondeDG.m`.

**Script 12.3.** *VandermondeDG.m: Routine for initializing the Vandermonde matrix,  $\hat{V}$ , of order  $m$  based on the point(s)  $r$ .*

---

```
function [V] = VandermondeDG(m, r)
% function [V] = VandermondeDG(m, r)
% Purpose : Initialize the 1D Vandermonde Matrix , V_{ij} = phi_j(r_i);

V = zeros(length(r), m+1);
for j=1:m+1
    V(:, j) = LegendreP(r (:), j-1);
end;
return
```

---

Using  $\hat{V}$  allows us to transform between modal representations, using  $\hat{\phi}_n$  as the unknowns, and nodal representations, employing  $\phi(r_i)$  as the unknowns. The two representations are mathematically equivalent but computationally different. Each has certain advantages and disadvantages, depending on the application at hand.

With a suitable local representation of the solution in place, we return to the elementwise operators  $M_j$  and  $S_j$ . For the former, the entries are given as

$$(M_j)_{kl} = \int_{x_{j-1/2}}^{x_{j+1/2}} \ell_k(x) \ell_l(x) dx = \frac{b_j}{2} \int_{-1}^1 \ell_k(r) \ell_l(r) dr = \frac{b_j}{2} (\ell_k, \ell_l)_l = \frac{b_j}{2} \hat{M}_{kl},$$

where the coefficient  $b_j/2$  is the Jacobian of (12.9) and  $\hat{M}$  is the mass matrix defined on the reference element  $l$ .

If we recall that

$$\ell_i(r) = \sum_{j=0}^m (\hat{V}^{-T})_{ij} P_j(r),$$

we recover

$$\begin{aligned} \hat{M}_{ij} &= \int_{-1}^1 \sum_{k=0}^m (\hat{V}^{-T})_{ik} P_k(r) \sum_{l=0}^m (\hat{V}^{-T})_{jl} P_l(r) dr \\ &= \sum_{k=0}^m \sum_{l=0}^m (\hat{V}^{-T})_{ik} (\hat{V}^{-T})_{jl} (P_k, P_l)_l = \sum_{k=0}^m (\hat{V}^{-T})_{ik} (\hat{V}^{-T})_{jk}, \end{aligned}$$

where the latter follows from orthonormality of  $P_i$ . Thus, we have

$$M_j = \frac{b_j}{2} \hat{M} = \frac{b_j}{2} (\hat{V} \hat{V}^T)^{-1}.$$

As far as the stiffness matrix  $S_j$  is considered, we obtain

$$(S_j)_{kl} = \int_{x_{j-1/2}}^{x_{j+1/2}} \ell_k(x) \frac{d\ell_l(x)}{dx} dx = \int_{-1}^1 \ell_k(r) \frac{d\ell_l(r)}{dr} dr = \hat{S}_{kl}.$$

Unlike the mass matrix, no metric constant is introduced by the transformation. To obtain a simple expression for  $\hat{S}$  we define the differentiation matrix  $\hat{D}$  with the entries

$$\hat{D}_{ij} = \left. \frac{d\ell_j}{dr} \right|_{r_i}.$$

The entries of  $\hat{M}\hat{D}$  are recovered as

$$\begin{aligned} (\hat{M}\hat{D})_{ij} &= \sum_{n=0}^m \hat{M}_{in} \hat{D}_{nj} = \sum_{n=0}^m \int_{-1}^1 \ell_i(r) \ell_n(r) \frac{d\ell_j}{dr} \Big|_{r_n} dr \\ &= \int_{-1}^1 \ell_i(r) \sum_{n=0}^m \frac{d\ell_j}{dr} \Big|_{r_n} \ell_n(r) dr = \int_{-1}^1 \ell_i(r) \frac{d\ell_j(r)}{dr} dr = \hat{S}_{ij}, \end{aligned}$$

which yields the identity

$$\hat{M}\hat{D} = \hat{S}. \quad (12.17)$$

The entries of  $\hat{D}$  can be found directly by considering

$$\hat{V}^T \ell(r) = P(r) \Rightarrow \hat{V}^T \frac{d}{dr} \ell(r) = \frac{d}{dr} P(r).$$

Collocating at the nodes  $r_i$  yields

$$\hat{V}^T \hat{D}^T = \left( \frac{d\hat{V}}{dr} \right)^T, \quad \frac{d\hat{V}}{dr} \Big|_{ij} = \frac{dP_j}{dr} \Big|_{r_i},$$

where the derivative of  $\hat{V}$  can be computed directly by the identity [103]

$$(1 - r^2) \frac{dP_i(r)}{dr} = -i r P_i(r) + i \sqrt{\frac{2i+1}{2i-1}} P_{i-1}(r)$$

in combination with the end-point values

$$\frac{dP_i}{dr}(\pm 1) = (\pm 1)^{i+1} \frac{i(i+1)}{2} \sqrt{\frac{2i+1}{2}}.$$

These identities are implemented in `GradLegendreP.m` to evaluate the gradient of  $\hat{V}$ , at the grid points  $r$ , as in `GradVandermondeDG.m`.

---

**Script 12.4. `GradLegendreP.m`: Evaluation of the derivative of the Legendre polynomial of order  $m$  at  $r$ .**

---

```
function [dP] = GradLegendreP(r, m);
% function [dP] = GradLegendreP(r, m);
% Purpose: Evaluate the derivative of the m'th order Legendre polynomial
% at points r
dP = zeros(length(r), 1);
if (m>0)
    Ph = -m*r.*LegendreP(r, m) + m*sqrt((2*m+1)/(2*m-1)).*LegendreP(r, m-1);
    dPe = r.^m*(m+1)*m*(m+1)/2*sqrt((2*m+1)/2);
    endp = (abs(abs(r)-1)>10*eps); rh = r.*endp;
    dP = ~endp.*dPe + endp.*Ph./(1-rh.^2);
end;
return
```

---

---

**Script 12.5. *GradVandermondeDG.m*: Evaluation of the derivative of the one-dimensional Vandermonde matrix  $\hat{V}$  of order  $m$  based on the grid points  $r$ .**

---

```
function [Vr] = GradVandermondeDG(m, r)
% function [Vr] = GradVandermondeDG(m, r)
% Purpose : Initialize the gradient of the Vandermonde matrix
% of order m at (r)
Vr = zeros(length(r), (m+1));
for i=0:m
    [Vr(:, i+1)] = GradLegendreP(r(:, i));
end
return
```

---

Finally, the entries of  $\hat{D}$  can be recovered directly as

$$\hat{D} = \left( \frac{d\hat{V}}{dr} \right) \hat{V}^{-1},$$

and shown in *DmatrixDG.m*. Once the differentiation matrix is in place, we can recover  $\hat{S}$  directly by (12.17).

---

**Script 12.6. *DmatrixDG.m*: Initialization of the one-dimensional differentiation matrix of order  $m$  based on the grid points  $r$ .**

---

```
function [D] = DmatrixDG(m, r, V)
% function [D] = DmatrixDG(m, r, V)
% Purpose : Initialize the (r) differentiation matrices ,
% evaluated at (r) at order m
Vr = GradVandermondeDG(m, r); D = Vr/V;
return
```

---

Let us now return to the error analysis for the nodal representation. While the difference between the projection and the interpolation may seem minor, it is essential to appreciate it. Consider the expression

$$\hat{v}_b(r) = \sum_{i=0}^m \hat{v}_i P_i(r), \quad v_b(r) = \sum_{i=0}^m \tilde{v}_i P_i(r),$$

where  $\hat{v}_b$  is the approximation, based on projection, and  $v_b$  refers to the interpolation.

By the interpolation property we have

$$v(r_j) = v_b(r_j) = (\hat{V}\tilde{v})_j = \sum_{i=0}^{\infty} \hat{v}_i P_i(r_j) = \sum_{i=0}^m \hat{v}_i P_i(r_j) + \sum_{i=m+1}^{\infty} \hat{v}_i P_i(r_j).$$

From this we recover

$$\hat{V}\tilde{v} = \hat{V}\hat{v} + \sum_{i=m+1}^{\infty} \hat{v}_i P_i(r),$$

where  $P_i(r) = [P_i(r_0), \dots, P_i(r_m)]^T$ . This implies

$$v_b(r) = \hat{v}_b(r) + P^T(r) \hat{V}^{-1} \sum_{i=m+1}^{\infty} \hat{v}_i P_i(r),$$

where  $\mathbf{P}(r) = [P_0(r), \dots, P_m(r)]^T$ . The difference between the modal and nodal expansion, expressed by the last term on the right-hand side, is given as

$$\mathbf{P}^T(r)\hat{\nabla}^{-1}\sum_{i=m+1}^{\infty}\hat{v}_iP_i(r)=\sum_{i=m+1}^{\infty}\hat{v}_i\left(\mathbf{P}^T(r)\hat{\nabla}^{-1}P_i(r)\right),$$

provided  $v \in H^p(\mathbb{I})$ ,  $p > 1/2$  [95]. Since

$$\mathbf{P}^T(r)\hat{\nabla}^{-1}P_i(r)=\sum_{l=0}^m p_lP_l(r), \quad \hat{\nabla}p=P_i(r),$$

we can interpret this term as the contribution from those high-order modes (recall  $i > m$ ) which look like lower-order modes on the grid. This phenomenon is known as aliasing and is the fundamental difference between an interpolation and a projection.

The impact of this on the overall accuracy, the proof of which is technical and given in [6], yields the following theorem.

**Theorem 12.6.** *Let  $v \in H^p(\mathbb{I})$ ,  $p > \frac{1}{2}$ , and let  $v_h$  be a polynomial interpolation of  $v$  of order  $m$ . Then, for all integer  $q$  such that  $0 \leq q \leq p$ , it holds that*

$$\|v - v_h\|_{\mathbb{I},q} \leq m^{2q-p+1/2}|v|_{\mathbb{I},p}.$$

Hence, aliasing can lead to a reduction of one in the rate of convergence, but not more than that. Combined with the result in Theorem 12.5, we now have a thorough understanding of the local approximation error and its relation to the smoothness of the function being approximated.

### 12.1.2 • Key properties

Let us now return to the scalar conservation law

$$\begin{aligned} \frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} &= 0, \quad x \in \Omega, \\ u(x, 0) &= u_0(x), \end{aligned} \tag{12.18}$$

where  $f(u)$  is assumed to be a convex flux. We also assume that appropriate boundary conditions are given.

Before continuing, let us briefly analyze the operators  $\hat{\mathbf{M}}$  and  $\hat{\mathbf{S}}$ . It is immediate to realize that

$$\mathbf{v}^T \hat{\mathbf{M}} \mathbf{v} = \int_{\mathbb{I}} \sum_{i=0}^m v(r_i) \ell_i(r) \sum_{j=0}^m v(r_j) \ell_j(r) dr = \|v_h\|_{\mathbb{I}}^2,$$

which we recognize as the local  $L^2$ -norm. Similarly,

$$\mathbf{v}^T \hat{\mathbf{S}} \mathbf{v} = \int_{\mathbb{D}_j} \sum_{i=0}^m v(r_i) \ell_i(r) \sum_{j=0}^m v(r_j) \frac{d\ell_j}{dr} dr = \int_{\mathbb{I}} v_h(r) v'_h(r) dr = \frac{1}{2} [v_h(r)^2]_{-1}^1$$

mimics integration by parts.

If we assume that the computational domain is represented by  $N$  elements, taken to be of equal size for simplicity, we recover the elementwise scheme in weak form

$$\frac{b}{2} \hat{M} \frac{d \mathbf{u}_j}{dt} - (\hat{S})^T \mathbf{f}_j = -F_{j+1/2} \psi(x_{j+1/2}) + F_{j-1/2} \psi(x_{j-1/2}), \quad (12.19)$$

and the corresponding strong form

$$\begin{aligned} \frac{b}{2} \hat{M} \frac{d \mathbf{u}_j}{dt} - \hat{S} \mathbf{f}_j &= (f(u(x_{j+1/2})) - F_{j+1/2}) \psi(x_{j+1/2}) \\ &\quad - (f(u(x_{j-1/2})) - F_{j-1/2}) \psi(x_{j-1/2}). \end{aligned} \quad (12.20)$$

Since  $f(u)$  is nonlinear we assume that its polynomial representation is obtained through a projection, i.e.,

$$f_j(x, t) = \sum_{i=0}^m \hat{f}_{j,i} \psi_{j,i}(x), \quad \hat{f}_{j,i} = \int_{D_j} f_j(x, t) \psi_{j,i}(x) dx,$$

where the integration is assumed to be exact. This eliminates aliasing errors.

**Theorem 12.7.** *The discontinuous Galerkin method with a single-valued numerical flux is locally and globally conservative. Furthermore, if the scheme converges, it converges to a weak solution of the conservation law.*

**Proof.** Without loss of generality, we consider the semidiscrete scheme in weak form. For a constant test function  $\phi = 1$ , we obtain

$$\phi^T \frac{b}{2} \hat{M} \frac{d \mathbf{u}_j}{dt} - \phi^T \hat{S}^T \mathbf{f}_j = \frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} u_j(x) dx = -F_{j+1/2} + F_{j-1/2}, \quad (12.21)$$

which ensures local conservation. Summing over all the elements, we recover

$$\sum_{j=1}^N \frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} u_j(x) dx = F_{-1/2}^+ + \sum_{j=0}^{N-1} (F_{j+1/2}^+ - F_{j+1/2}^-) - F_{N+1/2}^-.$$

If the numerical flux is single valued, i.e.,  $F_{j+1/2}^- = F_{j+1/2}^+$ , global conservation follows.

Now consider a general smooth test function [16],

$$\phi(x, t) \simeq \phi_b(x, t) = \bigoplus_{j=1}^N \phi_j(x, t), \quad \phi_j(x, t) = \sum_{i=0}^m \tilde{\phi}_{j,i}(t) \psi_{j,i}(x),$$

with  $\tilde{\phi}_{j,i} = \phi(x_i, t)$  in the nodal basis and  $\tilde{\phi}_{j,i} = (\phi_j(x), \psi_{j,i})_1$  for the modal representation. In both cases, the smooth test function is approximated by a piecewise polynomial of order  $m$ . As usual, we assume that the test function vanishes for large values of  $t$  and is compactly supported on  $\Omega$ . We recover

$$\left( \phi_j, \frac{\partial \mathbf{u}_j}{\partial t} \right)_{D_j} - \left( \frac{\partial \phi_j}{\partial x}, f_j \right)_{D_j} = -F_{j+1/2} \phi_j(x_{j+1/2}) + F_{j-1/2} \phi_j(x_{j-1/2}) = -[\phi_j F]_{x_{j-1/2}}^{x_{j+1/2}}.$$

Integration by parts in time yields

$$\int_0^T \left( \left( \frac{\partial \phi_j}{\partial t}, u_j \right)_{D_j} + \left( \frac{\partial \phi_j}{\partial x}, f_j \right)_{D_j} - [\phi_j F]_{x_{j-1/2}}^{x_{j+1/2}} \right) dt + (\phi_j(0), u_j(0))_{D_j} = 0.$$

By summation over all elements we recover

$$\begin{aligned} & \int_0^T \left( \left( \frac{\partial \phi_b}{\partial t}, u_b \right)_b + \left( \frac{\partial \phi_b}{\partial x}, f_b \right)_b \right) dt + (\phi_b(0), u_b(0))_{\Omega, b} \\ &= \int_0^T \left[ \phi(x_{-1/2}) F_{-1/2}^+ + \sum_{j=0}^{N-1} \phi(x_{j+1/2}) (F_{j+1/2}^+ - F_{j+1/2}^-) - F_{N+1/2}^- \phi(x_{N+1/2}) \right] dt. \end{aligned}$$

Since  $\phi_b$  is a polynomial representation of a smooth and compactly supported test function,  $\phi$ , it converges as  $m$  increases and/or  $h$  decreases. This ensures that the right-hand side vanishes if a single-valued numerical flux is used. Thus, if  $u_b$  converges almost everywhere to a solution  $u$ , then this is a weak solution to the conservation law.  $\square$

An often used single-valued flux is

$$F(u, v) = \frac{f(u) + f(v)}{2} - \frac{\alpha}{2} \hat{n}(v - u),$$

whose similarity to the Lax-Friedrichs flux is clear. Here  $\hat{n}$  is the outward pointing normal vector at the boundary.

Additionally, we have the following important result [67].

**Theorem 12.8.** *Consider the scalar conservation law with a nonlinear flux that admits an exact polynomial representation. Then the discontinuous Galerkin method with a monotone flux is stable,*

$$\frac{d}{dt} \|u_h\|_h \leq 0,$$

and satisfies a cell entropy condition.

**Proof.** Without loss of generality, we consider the semidiscrete nodal scheme on strong form

$$\frac{b}{2} \hat{M} \frac{d}{dt} u_j + \hat{S} f_j = [\ell_j(x)(f_j - F)]_{x_{j-1/2}}^{x_{j+1/2}}.$$

If we multiply by  $u_j^T$  from the left, we obtain

$$\frac{1}{2} \frac{d}{dt} \|u_j\|_{D_j}^2 + \int_{D_j} u_j \frac{\partial}{\partial x} f_j dx = [u_j(f_j - F)]_{x_{j-1/2}}^{x_{j+1/2}}.$$

For the scalar conservation law, we consider the convex entropy function and the associated entropy flux

$$\eta(u) = \frac{u^2}{2}, \quad \psi'(u) = \eta' f'.$$

This implies

$$\psi(u) = \int f' u \, du = f(u)u - \int f \, du = f(u)u - g(u), \quad g(u) = \int f(u) \, du. \quad (12.22)$$

If we now consider

$$\begin{aligned} \int_{D_j} u_j \frac{\partial}{\partial x} f_j \, dx &= \int_{D_j} \eta'(u_j) f'(u_j) \frac{\partial}{\partial x} u_j \, dx \\ &= \int_{D_j} \psi'(u_j) \frac{\partial}{\partial x} u_j \, dx = \int_{D_j} \frac{\partial}{\partial x} \psi(u_j) \, dx, \end{aligned} \quad (12.23)$$

we recover

$$\frac{1}{2} \frac{d}{dt} \|u_j\|_{D_j}^2 = [u_j(f_j - F) - \psi(u_j)]_{x_{j-1/2}}^{x_{j+1/2}}.$$

To ensure nonlinear stability at each interface, we have the requirement

$$\psi(u^+) - \psi(u^-) - u^+(f^+ - F) + u^-(f^- - F) \leq 0.$$

As usual,  $u^-$  is the solution to the left of the interface, and  $u^+$  is the solution to the right of the interface. Using (12.22), we have

$$-g(u^+) + g(u^-) - F(u^- - u^+) \leq 0. \quad (12.24)$$

Since  $g(u)$  by definition is at least continuous, the mean value theorem implies

$$g(u^-) - g(u^+) = g'(\xi)(u^- - u^+) = f(\xi)(u^- - u^+),$$

for some  $\xi \in [u^-, u^+]$ , and we recover the condition

$$(f(\xi) - F)(u^- - u^+) \leq 0.$$

We recognize this as the definition of an E-flux [85] and recall that this condition holds for all monotone fluxes, as discussed in Ex. 8.5. Summation over all elements yields nonlinear stability

$$\frac{d}{dt} \|u_b\|_b \leq 0. \quad (12.25)$$

This result was recovered without any stabilization or limiting of the solution. However, its validity hinges on the exact projection of  $f(u)$  onto the polynomial space in (12.23)—a requirement that generally can only be fulfilled for nonlinear fluxes of polynomial type.

We now define the consistent entropy flux  $\Psi$

$$\Psi(x) = F(x)u(x) - g(x),$$

and consider the cell entropy in weak form

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \|u_j\|_{D_j}^2 - \int_{D_j} \frac{\partial u_j}{\partial x} f_j \, dx + [u_j F]_{x_{j-1/2}}^{x_{j+1/2}} \\ = \frac{d}{dt} \int_{D_j} \eta(u_j) \, dx + \Psi(x_{j+1/2}) - \Psi(x_{j-1/2}) = 0. \end{aligned}$$

Since the polynomial solution  $u_j$  is smooth, it is a strict equality. If we include an interface, we obtain

$$\begin{aligned} & \frac{d}{dt} \int_{D_j} \eta(u_j) dx + \Psi(x_{j+1/2}^-) - \Psi(x_{j-1/2}^+) \\ &= \frac{d}{dt} \int_{D_j} \eta(u_j) dx + \Psi(x_{j+1/2}^-) - \Psi(x_{j-1/2}^-) + \Psi(x_{j-1/2}^-) - \Psi(x_{j-1/2}^+) \\ &= \frac{d}{dt} \int_{D_j} \eta(u_j) dx + \Psi(x_{j+1/2}^-) - \Psi(x_{j-1/2}^-) + \Phi_{j-1/2} = 0. \end{aligned}$$

Hence, if

$$\Phi_{j-1/2} = \Psi(x_{j-1/2}^-) - \Psi(x_{j-1/2}^+) = F_{j-1/2}(u_{j-1/2}^- - u_{j-1/2}^+) + g(u_{j-1/2}^+) - g(u_{j-1/2}^-) \geq 0,$$

we have established a cell entropy condition. However, this follows directly from (12.24), provided  $F$  is an E-flux. Consequently, we have

$$\frac{d}{dt} \int_{D_j} \eta(u_j) dx + \Psi(x_{j+1/2}^-) - \Psi(x_{j-1/2}^-) \leq 0, \quad (12.26)$$

which completes the proof.  $\square$

This fundamental result generalizes to multidimensional scalar problems and, as presented in [58], to symmetric hyperbolic systems with a quadratic entropy. As discussed in section 2.3, this suffices, under light additional conditions, to guarantee convergence to the unique entropy solution for scalar problems.

The extension to more general entropy functions is, however, less immediate and such developments are only very recent, initiated in [40, 14] and based on the development of summation-by-parts finite difference methods [37]. To sketch the approach, let us follow [21] and consider the scalar conservation laws

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0,$$

subject to an initial condition and periodic boundary conditions to keep it simple. The nodal discontinuous Galerkin scheme is

$$\frac{h}{2} \hat{M} \frac{d\mathbf{u}_j}{dt} + \hat{S} \mathbf{f}_j = \mathbf{B} [\mathbf{f}_j - \mathbf{F}],$$

where we have introduced the matrix  $\mathbf{B} = -\mathbf{e}_0 \mathbf{e}_0^T + \mathbf{e}_m \mathbf{e}_m^T$ . One easily proves that  $\hat{S} + \hat{S}^T = \mathbf{B}$ . Furthermore, we have

$$\sum_{l=0}^m \hat{S}_{il} = \sum_{l=0}^m \hat{D}_{il} = 0,$$

since these operators differentiate polynomials and we recall that  $\hat{S} = \hat{M} \hat{D}$ . Finally, we recover that

$$\sum_{l=0}^m \hat{S}_{lj} = \tau_j = \mathbf{B}_{jj}.$$

We recall from subsection 8.2.2 that if

$$(v^+ - v^-)F_{j+1/2} = \zeta^+ - \zeta^-,$$

with  $v = \eta'(u)$  being the entropy variable, then we call  $F_{j+1/2}$  an entropy conservative flux. Let us now consider the local entropy scheme

$$\frac{h}{2} \frac{d\mathbf{u}}{dt} + 2\hat{\mathbf{D}}\mathbf{f}_E = \hat{\mathbf{M}}^{-1} \mathbf{B}[\mathbf{f} - \mathbf{F}],$$

where we have removed the explicit reference to the element  $j$ . Here  $f_E(\mathbf{u}_i, \mathbf{u}_j)$  represents a symmetric entropy conservative flux. Local conservation is easily established since

$$\begin{aligned} \frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x) dx &= \sum_{l=0}^m \tau_l (f_l - F_l) - 2 \sum_{l=0}^m \sum_{k=0}^m \hat{S}_{kl} f_E(\mathbf{u}_k, \mathbf{u}_l) \\ &= \sum_{l=0}^m \tau_l (f_l - F_l) - \sum_{l=0}^m \sum_{k=0}^m (\hat{S}_{kl} + \hat{S}_{lk}) f_E(\mathbf{u}_k, \mathbf{u}_l) \\ &= \sum_{l=0}^m \tau_l (f_l - F_l) - \sum_{l=0}^m \sum_{k=0}^m \mathbf{B}_{kl} f_E(\mathbf{u}_k, \mathbf{u}_l) \\ &= \sum_{l=0}^m \tau_l (f_l - F_l) - \sum_{l=0}^m \tau_l f(\mathbf{u}_l) = -(F_0 - F_m), \end{aligned}$$

where we used consistency of the entropy flux. Since the numerical flux is single valued, global conservation follows.

To establish entropy conservation, we pursue a similar approach. Using  $v = \eta'(u)$  we recover

$$\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u) dx = \sum_{l=0}^m \tau_l v_l (f_l - F_l) - 2 \sum_{l=0}^m \sum_{k=0}^m \hat{S}_{kl} v_l f_E(\mathbf{u}_k, \mathbf{u}_l). \quad (12.27)$$

Since

$$\begin{aligned} 2 \sum_{l=0}^m \sum_{k=0}^m \hat{S}_{kl} v_l f_E(\mathbf{u}_k, \mathbf{u}_l) &= \sum_{l=0}^m \sum_{k=0}^m (\hat{S}_{kl} + \mathbf{B}_{lk} - \hat{S}_{lk}) v_l f_E(\mathbf{u}_k, \mathbf{u}_l) \\ &= \sum_{l=0}^m \tau_l v_l f_l + \sum_{l=0}^m \sum_{k=0}^m \hat{S}_{kl} (v_l - v_k) f_E(\mathbf{u}_l, \mathbf{u}_k) \\ &= \sum_{l=0}^m \tau_l v_l f_l + \sum_{l=0}^m \sum_{k=0}^m \hat{S}_{kl} (\zeta_l - \zeta_k), \end{aligned}$$

which, when combined with (12.27), yields

$$\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} \eta(u) dx = \sum_{l=0}^m \tau_l (\zeta_l - v_l f_E),$$

we recover entropy conservation. In [21] this is extended to symmetric hyperbolic systems and advection-diffusion problems. However, it is worth emphasizing that

entropy conservation is a local result while the establishment of a global result remains open. Also, the results are restricted to the semidiscrete problem and no results for the fully discrete problem are currently known.

### 12.1.3 • Error estimates

While the stability of the scheme is ensured by the appropriate choice of the numerical flux, we still need to investigate the accuracy of the method.

Consider the linear scalar problem

$$\frac{\partial u}{\partial t} = \mathcal{L}u,$$

subject to appropriate initial and boundary conditions to ensure wellposedness. We consider an approximation by an  $m$ th order piecewise polynomial  $u_b$ , which satisfies the semidiscrete scheme

$$\frac{du_b}{dt} + L_b u_b = 0.$$

Here  $L_b$  is the discrete approximation of  $\mathcal{L}$ . If we define the error as

$$\varepsilon(x, t) = u(x, t) - u_b(x, t),$$

then, as discussed in subsection 5.1.1, convergence is ensured by consistency

$$\begin{cases} \lim_{\text{dof} \rightarrow \infty} \|\varepsilon(\cdot, 0)\|_b = 0, \\ \lim_{\text{dof} \rightarrow \infty} \|\mathcal{T}(u(t))\|_b = 0, \end{cases}$$

where the growth in degrees of freedom (dof) reflects  $h$ -refinement, i.e., increasing  $N$ ; order enrichment, i.e., increasing  $m$ ; or some combination thereof. We also have  $\mathcal{T}(u(t))$  which represents the truncation error. We must also require stability:

$$\lim_{\text{dof} \rightarrow \infty} \|\exp(-L_b t)\|_b \leq C_b \exp(\alpha_b t), \quad t \geq 0. \quad (12.28)$$

Hence, the error comprises two components. The first one is associated with the accuracy of the representation of the initial condition. This is clearly an approximation problem and the accuracy of this is expressed in Theorems 12.5 and 12.6, suggesting an optimal error of  $\mathcal{O}(h^{m+1})$  in agreement with the results in Ex. 12.1.

However, this simple analysis does not account for the impact of the accumulated truncation error. To further simplify matters, consider

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0. \quad (12.29)$$

While Theorem 12.6 establishes consistency for the approximation of the spatial operator, stability is ensured by the choice of a monotone flux. The direct use of Theorems 12.5 and 12.6 yields

$$\|u - u_b\|_b \leq \frac{b^m}{m^{p-5/2}} |u|_{h,p},$$

for a smooth  $u$ . When compared to the results of Ex. 12.1, this appears to be a suboptimal result. To improve on this estimate, we need to consider the problem in more detail and establish convergence directly rather than through the equivalence theorem.

Following [23], let us define the bilinear form  $\mathcal{B}(u, \phi)$  as

$$\mathcal{B}(u, \phi) = (u_t, \phi)_\Omega + a(u_x, \phi)_\Omega.$$

Clearly  $\mathcal{B}(u, \phi) = 0$  must hold for all smooth test functions  $\phi$  if  $u$  is a solution to (12.29). For simplicity we assume that the problem is periodic, although this is not essential.

An immediate consequence is

$$\mathcal{B}(u, u) = \frac{1}{2} \frac{d}{dt} \|u\|_\Omega^2 = 0,$$

which expresses energy conservation. This implies that if we solve (12.29) with two different initial conditions,  $u_1(0)$  and  $u_2(0)$ , then

$$\frac{1}{2} \frac{d}{dt} \|\varepsilon\|_\Omega^2 = 0 \Rightarrow \|\varepsilon(T)\|_\Omega = \|u_1(0) - u_2(0)\|_\Omega,$$

where  $\varepsilon(t) = u_1(t) - u_2(t)$ .

If we mimic this approach for the discrete solution  $u_h \in V_h$ , we obtain

$$\mathcal{B}_h(u_h, \phi_h) = ((u_h)_t, \phi_h)_h + a((u_h)_x, \phi_h)_h - (\hat{n} \cdot (a u_h - F), \phi_h)_{\partial\Omega, h} = 0,$$

which must hold for all test functions,  $\phi_h \in V_h$ .

As the choice of the numerical flux, we first consider a central flux

$$F(u, v) = a \frac{u + v}{2}$$

to obtain

$$\mathcal{B}_h(u_h, \phi_h) = ((u_h)_t, \phi_h)_h + a((u_h)_x, \phi_h)_h - \frac{1}{2} ([\![a u_h]\!], \phi_h)_{\partial\Omega, h} = 0. \quad (12.30)$$

Here we have resorted to the standard notation

$$\{\{u\}\} = \frac{u^+ + u^-}{2}, \quad [\![u]\!] = \hat{n}^- u^- + \hat{n}^+ u^+,$$

which represent, respectively, the local average and the jump at the cell interface.

Since the jump term vanishes for the exact (smooth) solution, we recover

$$\mathcal{B}_h(u, \phi_h) = 0$$

and the error equation

$$\mathcal{B}_h(\varepsilon, \phi_h) = 0, \quad \varepsilon = u - u_h.$$

Observe that  $\varepsilon$  is not an element in  $V_h$ . Let us therefore write this as

$$\mathcal{B}_h(\varepsilon, \phi_h) = \mathcal{B}_h(\varepsilon_h, \phi_h) + \mathcal{B}_h(\varepsilon - \varepsilon_h, \phi_h) = 0, \quad (12.31)$$

where  $\varepsilon_h = \mathcal{P}_h \varepsilon \in V_h$  and  $\mathcal{P}_h$  represents the projection operator onto  $V_h$ . Taking  $\phi_h = \varepsilon_h$  to mimic the continuous case, we recover

$$\mathcal{B}_h(\varepsilon_h, \varepsilon_h) = ((\varepsilon_h)_t, \varepsilon_h)_h + a((\varepsilon_h)_x, \varepsilon_h)_h - \frac{1}{2} ([\![a \varepsilon_h]\!], \varepsilon_h)_{\partial\Omega, h}.$$

After integration by parts in space and use of periodicity, the boundary term vanishes and we recover

$$\mathcal{B}_b(\varepsilon_b, \varepsilon_b) = \frac{1}{2} \frac{d}{dt} \|\varepsilon_b\|_b^2.$$

Since

$$\mathcal{P}_b \varepsilon - \varepsilon = \mathcal{P}_b(u - u_b) - (u - u_b) = \mathcal{P}_b u - u,$$

we recall (12.31) to recover

$$\frac{1}{2} \frac{d}{dt} \|\varepsilon_b\|_b^2 = \mathcal{B}_b(\mathcal{P}_b u - u, \varepsilon_b) = (\hat{n}\{\{a(u - \mathcal{P}_b u)\}\}, \varepsilon_b)_{\partial\Omega, b}$$

by (12.30) and integration by parts. To bound this, we rely on the following result [22].

**Lemma 12.9.** *Let  $u \in H^{p+1}(\mathcal{D}_j \cup \mathcal{D}_{j+1})$ . Then it holds that*

$$\{\{a(u - \mathcal{P}_b u)\}\}|_{x_{j+1/2}} \leq C_j b^{\sigma-1/2} \frac{|a|}{2} |u|_{\mathcal{D}_{j,\sigma}},$$

where the constant  $C_j$  depends only on  $j$  and  $\sigma = \min(m+1, p)$ .

The proof follows from the application of the Bramble–Hilbert lemma, Theorem 12.6, and the standard trace inequality

$$\|u_b\|_{\partial\mathcal{D}} \leq \frac{C(m)}{\sqrt{b}} \|u_b\|_{\mathcal{D}}.$$

We now have

$$\begin{aligned} |\mathcal{B}_b(u - \mathcal{P}_b u, \varepsilon_b)| &\leq \frac{1}{2} ((\{\{a(u - \mathcal{P}_b u)\}\}, \{\{a(u - \mathcal{P}_b u)\}\})_{\partial\Omega, b} + (\varepsilon_b, \varepsilon_b)_{\partial\Omega, b}) \\ &\leq C |a| b^{2\sigma-1} \|u\|_{b,\sigma+1}^2, \end{aligned}$$

where we apply Lemma 12.9 to bound the first term and Theorem 12.6 in combination with the trace inequality to bound the second term.

The final result is

$$\frac{d}{dt} \|\varepsilon_b\|_b^2 \leq C |a| b^{2\sigma-1} \|u\|_{b,\sigma+1}^2,$$

from which we recover the error estimate

$$\|\varepsilon_b(T)\| \leq (C_1 + C_2 T) b^{m+1/2},$$

provided that  $u$  is sufficiently smooth. This result was first established in [77]. If we changed the flux to a more general one, the result still holds [23].

Recovery of the optimal  $\mathcal{O}(b^{m+1})$  convergence result relies on the use of a superconvergence property, and only applies to one-dimensional linear problems with strict upwinding. The main idea is to replace the orthogonal projection with a downwinded projection. Although the Galerkin orthogonality no longer holds in this case, the

boundary terms in the fluxes vanish identically. Consequently, the main error contribution comes from the interior parts of the scheme. This can be estimated by the results in Theorem 12.6, resulting in the optimal estimate

$$\|\varepsilon_h(T)\|_{\Omega,h} \leq C(m)h^{m+1}(1 + C_1(m)T).$$

This estimate also shows that we cannot generally hope for a slower-than-linear growth in time of the error for the linear advection problem, defined on  $\mathbf{R}$ . The optimal convergence rate for the linear case has been proven in [77] and a further discussion can be found in [68, 23].

In the semidiscrete case, the extension to the smooth nonlinear scalar case yields similar results [31]. The analysis of the fully discrete case with an  $n$ th order SSPERK method to advance in time, yields error estimates as  $\mathcal{O}(h^{m+\sigma} + k^n)$ . The value of  $\sigma$  depends on the choice of the flux. For a central flux  $\sigma = 1/2$ , while  $\sigma = 1$  for upwinding. We refer to [123, 124] for the details. The extension to linear systems yields similar results, provided the numerical flux employs exact upwinding.

For the multidimensional case, which we revisit in section 12.4, a counter example [87] shows that one cannot do better than  $\mathcal{O}(h^{m+1/2})$  for smooth problems on general grids.

### 12.1.4 ■ Phase error analysis

In spite of the suboptimal, yet sharp, result just discussed, one generally observes optimal convergence for multidimensional cases and general grids, provided a monotone flux is used. This can be attributed to the excellent dispersive properties of the discontinuous Galerkin method.

Consider again the wave equation

$$\begin{aligned} \frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} &= 0, \\ u(x, 0) &= \exp(i l x), \end{aligned} \tag{12.32}$$

where  $x \in \mathbf{R}$  and  $l$  is the wave number for the initial condition. If we seek spatially periodic solutions of the form

$$u(x, t) = \exp(i(lx - \omega t)),$$

we recover  $\omega = al$ , recognized as the dispersion relation, and  $a$  is the exact phase velocity. Taking inspiration from the phase error analysis in section 8.1, we seek to quantify how well the discontinuous Galerkin scheme reproduces this behavior.

Assume that the computational domain is split into elements  $D_j$  of equal length  $h$ . We recover the basic semidiscrete local scheme

$$\frac{h}{2} \hat{M} \frac{d\mathbf{u}_j}{dt} + a \hat{S} \mathbf{u}_j = \mathbf{e}_m \left[ (a \mathbf{u}_j) - F \right]_{x_{j+1/2}} - \mathbf{e}_0 \left[ (a \mathbf{u}_j) - F \right]_{x_{j-1/2}}, \tag{12.33}$$

where  $x \in D_j$  and  $\mathbf{e}_i$  is the  $i$ th standard basis vector of length  $m + 1$ . We consider the flux

$$F(u^-, u^+) = \{au\} + |a| \frac{1-\alpha}{2} [[u]].$$

Without loss of generality, we assume that  $\alpha \geq 0$  and recover

$$\begin{aligned} \frac{b}{2} \hat{\mathbf{M}} \frac{d\mathbf{u}_j}{dt} + a \hat{\mathbf{S}} \mathbf{u}_j &= \frac{\alpha}{2} \mathbf{e}_m (u_{j+1/2}^- - u_{j+1/2}^+) \\ &\quad - \frac{\alpha(2-\alpha)}{2} \mathbf{e}_0 (u_{j-1/2}^+ - u_{j-1/2}^-). \end{aligned}$$

We now look for local solutions of the form

$$\mathbf{u}_j(x, t) = \mathbf{U}_j \exp(i(lx - \omega t))$$

and periodicity of the solution

$$u_{j+1/2}^\pm = \exp(ilh) u_{j-1/2}^\pm.$$

This yields

$$\begin{bmatrix} \frac{i\omega h}{2} \hat{\mathbf{M}} + a \hat{\mathbf{S}} - \frac{\alpha}{2} \mathbf{e}_m (\mathbf{e}_m^T - \exp(ilh) \mathbf{e}_0^T) \\ + \frac{\alpha(2-\alpha)}{2} \mathbf{e}_0 (\mathbf{e}_0^T - \exp(-ilh) \mathbf{e}_m^T) \end{bmatrix} \mathbf{U}_j = 0.$$

Slightly rewriting this yields a generalized eigenvalue problem

$$\begin{bmatrix} 2 \hat{\mathbf{S}} - \alpha \mathbf{e}_m (\mathbf{e}_m^T - \exp(iL(m+1)) \mathbf{e}_0^T) \\ + (2-\alpha) \mathbf{e}_0 (\mathbf{e}_0^T - \exp(-iL(m+1)) \mathbf{e}_m^T) \end{bmatrix} \mathbf{U}_j = i\Omega \hat{\mathbf{M}} \mathbf{U}_j,$$

where we have introduced the dimensionless parameters

$$L = \frac{lh}{m+1} = \frac{2\pi}{\lambda} \frac{b}{m+1} = 2\pi p^{-1}, \quad \Omega = \frac{\omega b}{a}.$$

The parameter  $p$ , defined as

$$p = \frac{\lambda}{b/(m+1)},$$

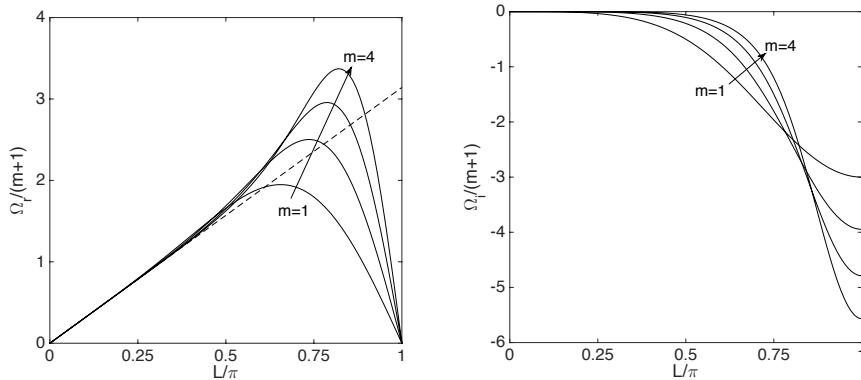
measures the number of degrees of freedom per wavelength, as introduced in section 8.1.

We recognize  $L = \Omega/(m+1)$  as the numerical dispersion relation. Hence, by solving the eigenvalue problem for  $\Omega = \Omega_r + i\Omega_i$ , we recover the dispersion relation of the numerical scheme. Here  $\Omega_r$  is an approximation to the frequency  $\omega$  while  $\Omega_i \simeq 0$  is the dissipation associated with the scheme. In particular, we note that

$$\frac{\Omega_r}{(m+1)L}$$

provides a measure of the phase velocity of the numerical scheme, also considered in section 8.1.

Figure 12.3 shows the dispersion relation for the upwind scheme for different orders of approximation. As expected, for  $L \ll 1$ , the numerical phase speed approximates the physical wave speed very well and, as the order of the approximation increases, this



**Figure 12.3.** Numerical dispersion relations of the discontinuous Galerkin approximation for the linear advection operator with an upwind flux. We show the phase speed (left) and the dissipation (right) for different orders of approximation  $m$ . The dashed line represents the exact dispersion relation.

agreement improves for an increasing range of  $L$ . This supports the benefits of using high-order schemes for wave propagation. Furthermore, we see that there is a range of marginally resolved wave numbers for which the numerical wave speed is faster than the physical speed. Finally, Fig. 12.3 shows  $\Omega_i$  for different orders of approximation. Consistent with our understanding of the upwind flux, a significant dissipation of the high-frequency components occurs.

Similar investigations can be used to understand the properties of the scheme for different choices of the flux. Computational studies of the dispersive and dissipative properties of discontinuous Galerkin methods were initiated in [62, 96], including the extension to two spatial dimensions. Steps toward a more rigorous analysis were taken in [60, 61], including extensions to two-dimensional problems and the impact of boundary conditions. A number of conjectures made in earlier work have been established rigorously in [4] and we outline the key results in the following.

In the highly resolved region  $lh \ll 1$ , provided  $\alpha \neq 1$ , the dispersive error is [4]

$$|\mathcal{R}(\tilde{lh}) - \mathcal{R}(lh)| \simeq \frac{1}{2} \left[ \frac{m!}{(2m+1)!} \right]^2 (lh)^{2m+3}$$

and the dissipative error

$$|\mathcal{I}(\tilde{lh})| \simeq \frac{1}{2} \left[ \frac{m!}{(2m+1)!} \right]^2 (1-\alpha)^{(-1)^m} (lh)^{2m+2}.$$

Here  $l$  is the exact wave number while  $\tilde{l}$  is the numerical wave number,  $\tilde{l} = \Omega/h$ .

For the nondissipative central flux, i.e.,  $\alpha = 1$ , the situation is a little more complicated since

$$|\mathcal{R}(\tilde{lh}) - \mathcal{R}(lh)| \simeq \frac{1}{2} \left[ \frac{m!}{(2m+1)!} \right]^2 \begin{cases} -\frac{1}{2} (lh)^{2m+3}, & m \text{ even}, \\ 2(\tilde{lh})^{2m+1}, & m \text{ odd} \end{cases}$$

illustrates an order reduction for odd values of  $m$ . In all cases, the coefficient in front of  $lh$  decreases rapidly with  $m$ , highlighting the benefits of using large values of  $m$ .

The excellent dispersive properties help to understand why one very often observes  $\mathcal{O}(h^{m+1})$  error in computational experiments, even when the error analysis does not support this. Simply put, the approximation error associated with the representation of the initial conditions shows up as a dominating component. The additional dispersive and dissipative errors accumulate at a very slow rate, and do not dominate unless very long time integration is considered.

This addresses directly how the dispersive and dissipative errors will behave as  $h$  approaches zero, while keeping  $m$  and  $l$  fixed, i.e., under  $h$ -refinement. However, it is also interesting to consider the case in which  $lh$  is kept fixed but we increase  $m$ , i.e., order enrichment.

Following section 8.1, we define the phase error

$$\rho_m = \left| \frac{\exp(ilh) - \exp(i\tilde{lh})}{\exp(ilh)} \right|.$$

The convergence of  $\rho_m$  falls into three separate regions with distinct behavior [4]:

$$\rho_m \simeq \begin{cases} 2m+1 < lh - C(lh)^{1/3}, & \text{no convergence,} \\ lh - o(lh)^{1/3} < 2m+1 < lh + o(lh)^{1/3}, & \mathcal{O}(m^{-1/3}) \text{ convergence,} \\ 2m+1 \gg lh, & \mathcal{O}(hl/(2m+1))^{2m+2} \text{ convergence.} \end{cases}$$

It is interesting to note that the threshold for convergence is

$$2 \simeq \frac{lh}{m+1} = 2\pi p^{-1}.$$

This reflects that  $p \geq \pi$  is required to achieve rapid convergence, in agreement with classic results from spectral methods [43].

Finally, the following result provides a useful guideline to determine the resolution requirements for a general wave problem [4].

**Theorem 12.10.** *Let  $\chi > 1$  be fixed. If  $m, lh \rightarrow \infty$ , and  $(2m+1)/lh \rightarrow \chi$ , then*

$$\rho_m \simeq C \exp(-\beta(m+1/2)),$$

where  $\beta > 0$  and  $C$  are constants that depends on  $\chi$  but not on  $N$ .

Thus, as long as  $\chi > 1$  (i.e.,  $p > \pi$ ), there is exponential decay of the phase error.

## 12.2 • Nonsmooth problems

Let us now turn the attention to how discontinuous Galerkin methods deal with conservation laws which are expected to develop discontinuous solutions. As discussed at length in section 8.3, the use of a linear high-order approximation, e.g., a polynomial representation, will lead to artificial oscillations. While these oscillations may not ultimately destroy the accuracy of the solution, there are numerous situations where such oscillations cannot be tolerated.

To illustrate the behavior of the discontinuous Galerkin method for discontinuous solutions, let us consider an example.

**Example 12.11.** Consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\pi}{6} \frac{\partial u}{\partial x} = 0, \quad x \in [0, 1],$$

with the initial condition

$$u(x, 0) = 1 - H(x - 0.25),$$

where  $H$  is the Heaviside step-function. The exact solution is given as  $u(x, t) = 1 - H(x - 0.25 - \pi/6t)$ . We solve the problem with a discontinuous Galerkin method of order  $m$  for different numbers  $N$  of elements and employ an upwind flux and a third order SSPERK scheme to advance in time.

The results are illustrated in Fig. 12.4 and they allow us to make a number of observations. First of all, we recognize the discontinuous nature of the solution and the emergence of oscillations, which persist even as the resolution is increased. At the highest resolution, we observe the similarity with the numerical oscillations discussed in section 8.3, recognized as a manifestation of the Gibbs phenomenon.

A more subtle observation is that the spreading of the oscillations appears to be localized to a few cells in the neighborhood of the discontinuity. In other words, the oscillations only have a local effect on the overall accuracy of the error.

A local analysis [24] confirms this observation for the case of  $m = 1$  and a second order SSPERK scheme. The influence of the discontinuity is limited to a range from  $\mathcal{O}(T^{1/3}h^{2/3} \log 1/h)$  to the left of the discontinuity to  $\mathcal{O}(T^{1/2}h^{1/2} \log 1/h)$  to the right of the discontinuity. This is in qualitative agreement with the results in Fig. 12.4, where the impact of the discontinuity decreases with  $h$  and is stronger downstream of the discontinuity than upstream of it. Outside of this range, full order of accuracy can be expected. No rigorous results are known for higher values of  $m$ . However, computational results suggest that this carries over to general values of  $m$ . ■

### 12.2.1 • A detour on hidden accuracy

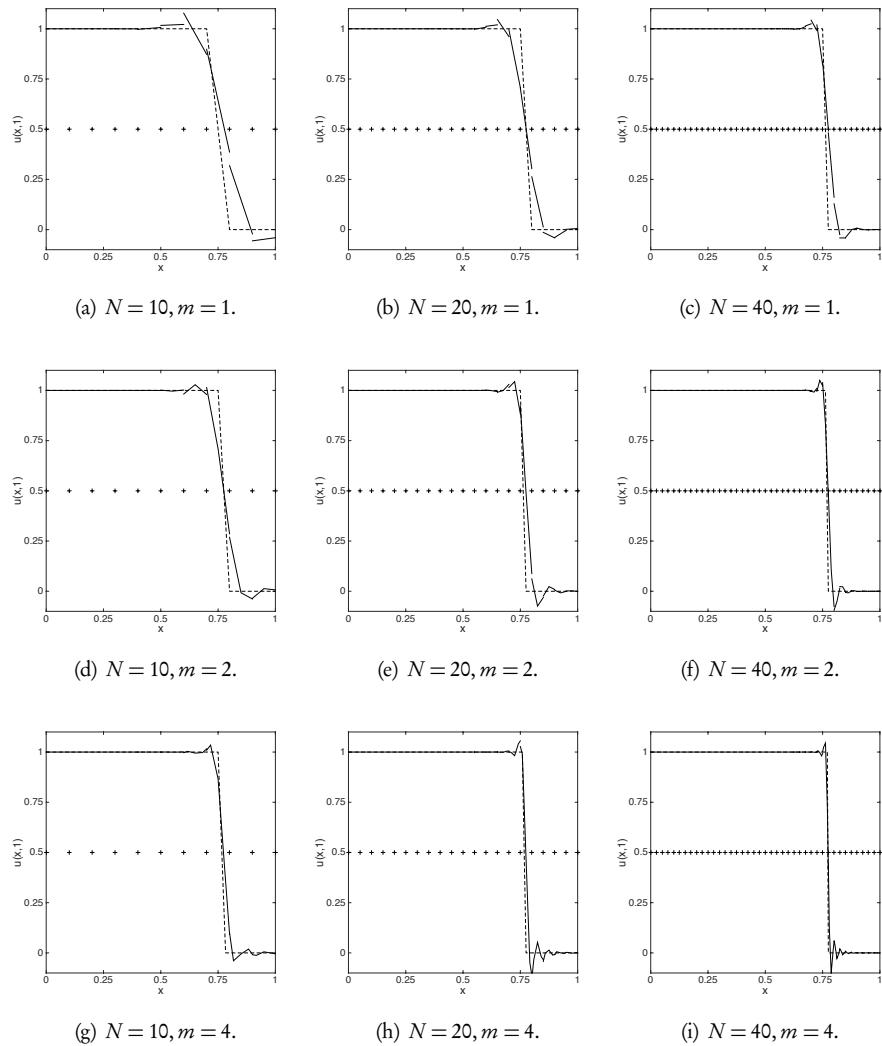
Before continuing with the discussion on how to diminish, or potentially eliminate, the artificial oscillations, it is worth understanding the extend to which these oscillations destroy the accuracy of the solution.

By a simple inspection of the solutions in Fig. 12.4, it is clear that the oscillations have a substantial impact on the pointwise accuracy of the solution, even though this impact remains localized. On the other hand, as discussed in section 8.3, the oscillations may not be as destructive as one may imagine, and a high-order accurate solution may still be available, i.e., while the accuracy is hidden [84, 76] in the sense that the pointwise accuracy is destroyed, the accuracy of the moments remains intact. In such a case, it may be possible to recover a high-order accurate solution.

The result in section 8.3 pertains to linear problems with nonsmooth initial conditions. While certainly relevant, it is hardly the only way that reduced smoothness can enter into a problem.

Let us consider the problem

$$\frac{\partial u}{\partial t} + \frac{\partial a(x)u}{\partial x} - \sigma(x)u = 0, \quad x \in \Omega, \quad (12.34)$$



**Figure 12.4.** Solution of the linear wave problem at  $T = 1.0$  with a discontinuous initial condition. Results are shown for different orders of approximation,  $m$ , and different numbers of elements,  $N$ . The dashed line indicates the exact solution while “+” marks the position of cell interfaces.

subject to periodic boundary conditions and an appropriate initial condition  $u_0$ . Here  $a$  and  $\sigma$  are possibly nonsmooth functions. We define the auxiliary problem

$$\frac{\partial v}{\partial t} + a(x) \frac{\partial v}{\partial x} + \sigma(x)v = 0, \quad (12.35)$$

subject to periodic boundary conditions and with  $v_0$  as the initial condition. We have the following result.

**Lemma 12.12 (continuous Green's identity).** *The periodic solutions  $u, v$  to (12.34) and (12.35) satisfy*

$$(u(x, t), v(x, t))_{\Omega} = (u_0(x), v_0(x))_{\Omega} \quad \forall t \in (0; T]. \quad (12.36)$$

**Proof.** Integration by parts yields

$$\begin{aligned} \partial_t(u(x, t), v(x, t))_{\Omega} &= -(\partial_x(a(x)u(x, t)), v(x, t))_{\Omega} - (u(x, t), a(x)\partial_x v(x, t))_{\Omega} \\ &= -(\partial_x(a(x)u(x, t)), v(x, t))_{\Omega} + (\partial_x(a(x)u(x, t)), v(x, t))_{\Omega} = 0, \end{aligned}$$

from which the result follows.  $\square$

Now consider the semidiscrete discontinuous Galerkin scheme for (12.34). We seek  $u_b \in V_b$  such that

$$\begin{aligned} \sum_{j=1}^N \partial_t \int_{D_j} u_j \psi_j dx - \int_{D_j} a(x) u_j \partial_x \psi_j dx + F_{j+1/2} \psi_j(x_{j+1/2}) - F_{j-1/2} \psi_j(x_{j-1/2}) \\ - \int_{D_j} \sigma u_j \psi_j dx = 0 \end{aligned} \quad (12.37)$$

holds for all  $\psi_b \in V_b$ . We express this as

$$\partial_t \mathcal{M}(u_b, \psi_b) + \mathcal{B}(u_b, \psi_b) = 0 \quad \forall \psi_b \in V_b, \quad (12.38)$$

where

$$\mathcal{M}(u, v) = \sum_{j=1}^N \int_{D_j} u v dx, \quad (12.39)$$

$$\mathcal{B}(u, v) = \sum_{j=1}^N - \int_{D_j} a(x) u \partial_x v dx + F_{j+1/2}(u) v(x_{j+1/2}) \quad (12.40)$$

$$- F_{j-1/2}(u) v(x_{j-1/2}) - \int_{D_j} \sigma u_j \psi_j dx.$$

We define adjoint-consistency for smooth hyperbolic problems as follows.

**Definition 12.13 (adjoint-consistency).** *The discontinuous Galerkin approximation (12.38) of problem (12.34) is adjoint-consistent if*

$$\partial_t \mathcal{M}(\phi_b, v) + \mathcal{B}(\phi_b, v) = 0 \quad \forall \phi_b \in V_b \quad (12.41)$$

*is satisfied for all  $t \in (0, T]$  by the exact solution of (12.35).*

This provides a variational formulation of the auxiliary problem, namely, find  $v_b \in V_b$  such that

$$\partial_t \mathcal{M}(\phi_b, v_b) - \mathcal{B}(\phi_b, v_b) = 0 \quad \forall \phi_b \in V_b. \quad (12.42)$$

It is straightforward to show that standard discontinuous Galerkin methods are adjoint-consistent; see [132]. This leads to the following result.

**Lemma 12.14 (semidiscrete Green's identity).** *Consider the periodic solutions  $u_b, v_b$  of the semidiscrete approximations of (12.34) and (12.35), respectively, and assume that the scheme is stable, consistent, and adjoint-consistent. Then*

$$(u_b(x, t), v_b(x, t))_h = (u_b(x, 0), v_b(x, 0))_h \quad \forall t \in (0, T]. \quad (12.43)$$

**Proof.** The result follows directly from the two variational statements, (12.38) and (12.42), when,  $v_b$  and  $u_b$ , respectively, are chosen as test functions.  $\square$

The approximation error of the moment of the initial condition can be bounded as follows.

**Lemma 12.15.** *Let  $u_0 \in L^2(\Omega)$  and  $v_0 \in C^\infty(\Omega)$  and let  $u_b(x, 0)$  and  $v_b(x, 0)$  be their  $m$ th order piecewise polynomial approximations. Then*

$$|(u_b(x, 0), v_b(x, 0))_h - (u_0(x), v_0(x))_h| \leq C \frac{\|v_0^{(s)}\|_h}{m^s} \sim C \|v_0\|_h \frac{s!}{m^s} \quad (12.44)$$

holds for  $s \geq 0$ . The constant  $C$  depends on  $u_0$ , but is independent of  $m$ .

**Proof.** We express the error as

$$\begin{aligned} & (u_b(x, 0), v_b(x, 0))_h - (u_0(x), v_0(x))_h \\ &= (u_b(x, 0) - u_0(x), v_b(x, 0))_h + (u_0(x), v_b(x, 0) - v_0(x))_h. \end{aligned}$$

Since  $v_b(\cdot, 0)$  is an  $m$ th order polynomial the first term vanishes by Galerkin orthogonality. By the Cauchy–Schwarz inequality and Theorem 12.2, we have that

$$|(u_0(x), v_b(x, 0) - v_0(x))_h| \leq C \|u_0\|_h \|v_b - v_0\|_h \leq C \|u_0\|_h \|v_0^{(s)}\|_h / m^s, \quad (12.45)$$

which yields the result.  $\square$

Hence, the approximation of the initial moments is exponentially accurate for  $m \rightarrow \infty$ . This now paves the way for the classic result [1].

**Theorem 12.16.** *Let  $u, v$  be the exact solutions of (12.34) and (12.35), respectively, and assume that  $a$  and  $\sigma$  are smooth functions. Furthermore, let  $u_b, v_b$  be the semidiscrete solutions obtained by the discontinuous Galerkin method. Also, let  $v_0 \in C^\infty(\Omega)$  be an arbitrary function. Then, for  $t \in (0, T]$  and  $s \geq 0$ , it holds that*

$$|(u_b(x, t) - u(x, t), v(x, t))_h| \sim C \|v_0\|_h \frac{s!}{m^s}. \quad (12.46)$$

**Proof.** Since  $v_0 \in C^\infty(\Omega)$ , then  $v(x, t) \in C^\infty(\Omega)$  for  $t \in (0, T]$ . Hence, we can replace  $v$  by  $v_b$  in the inner products as  $m \rightarrow \infty$ . Therefore, we obtain that

$$\begin{aligned} & |(u_b(x, t), v_b(x, t))_h - (u(x, t), v(x, t))_h| \\ &= |(u_b(x, 0), v_b(x, 0))_h - (u_0(x), v_0(x))_h| \sim C \|v_0\|_h s! / m^s, \end{aligned}$$

which completes the proof.  $\square$

Even though the polynomial approximation of the discontinuous initial condition destroys the pointwise accuracy, the accuracy of the moments is maintained, i.e., with careful post processing, a high-order accurate solution remains attainable. The extension to the case of a discontinuous forcing function  $f$  requires the definition of a discrete Duhamel principle, but otherwise yields the same result [132].

Let us now consider the general case in which  $a$  and/or  $\sigma$  are discontinuous. We recover the following result.

**Theorem 12.17.** *Let  $u_0 \in C^\infty(\Omega)$  and  $u_b$  be the  $m$ th order polynomial solution of (12.34), obtained using a discontinuous Galerkin method. Furthermore, let  $f \in C^\infty(\Omega)$  be an arbitrary smooth function. As  $m \rightarrow \infty$ , there exists a sequence of functions  $\{g_b\} \in V_b$  such that*

$$\lim_{m \rightarrow \infty} \|g_m - f\|_\Omega \rightarrow 0 \quad (12.47)$$

and

$$|(u_b(x, t), g_b(x))_b - (u(x, t), f(x))_b| \leq C \frac{\|u_0^{(s)}\|_b}{m^s} \sim C \|u_0\|_b \frac{s!}{m^s}, \quad (12.48)$$

for any  $s > 0$ .

The theorem states the surprising result that exponentially accurate information about the moments is contained in the numerical data, even for a problem with non-smooth coefficients. Furthermore, the functions  $g_b$  are uniquely determined, i.e.,  $g_b = \Gamma_b(f)$ , with some mapping  $\Gamma_b : C^\infty \rightarrow V_b$ .

**Proof.** Define  $v_0$  in (12.35) such that  $f(x) = v(x, t)$  and require that  $f \in C^\infty(\Omega)$ . This is possible since we restrict ourselves to a hyperbolic problem. However, due to the low regularity of  $a$ , the corresponding initial condition  $v_0$ , obtained as the solution to the backward problem, will exhibit low regularity.

Now, we use  $v_0$  as the initial condition for the semidiscrete solution of (12.35), and solve the problem forward in time to  $t$  using a discontinuous Galerkin method. We set  $g_b(x) = v_b(x, t)$ . As  $a, \sigma$  are discontinuous functions, we have

$$\|a - a_b\|_b \sim m^{-1}, \quad \|\sigma - \sigma_b\|_b \sim m^{-1},$$

which implies that  $\|g_b - f\|_b \sim m^{-1}$ .

By Galerkin orthogonality we have

$$(u_b(x, 0), v_b(x, 0) - v_0(x))_b = 0. \quad (12.49)$$

Since  $u_0 \in C^\infty(\Omega)$ , its  $L^2$ -projection is high-order accurate. Combined with (12.49) we have that for  $m \rightarrow \infty$  and  $s > 0$

$$|(u_b(x, 0), g_b(x, 0))_b - (u_0(x), v_0(x))_b| \leq C \frac{\|u^{(s)}\|_b}{m^s}, \quad (12.50)$$

where  $C$  is independent of  $m$ . Hence the approximation of the moments of the initial conditions is high-order accurate, even though the initial condition  $v_0$  of the adjoint problem is a discontinuous function. The result follows by applying Lemma 12.12 and Lemma 12.14 to (12.50).  $\square$

This result suggests that one can recover high-order accurate information. To illustrate this, let us consider an example [132].

**Example 12.18.** Consider the linear problem

$$\frac{\partial u}{\partial t} + \frac{\partial a u}{\partial x} = 0,$$

solved using an upwind discontinuous Galerkin method. The adjoint problem, needed to obtain  $g_h$ , is solved by the naturally arising scheme (12.42). We consider the space-time domain  $[0, 2\pi] \times (0, \pi/8]$ , which is discretized by  $N = 6$  elements and various orders of approximation. Quadrature rules of sufficient order are used to integrate all terms exactly. We employ

$$u_0(x) = \sin(x) + \sin(2x + 0.32) + \sin(9x + 1.31) + 0.1 \sin(12x + 2.11) \quad (12.51)$$

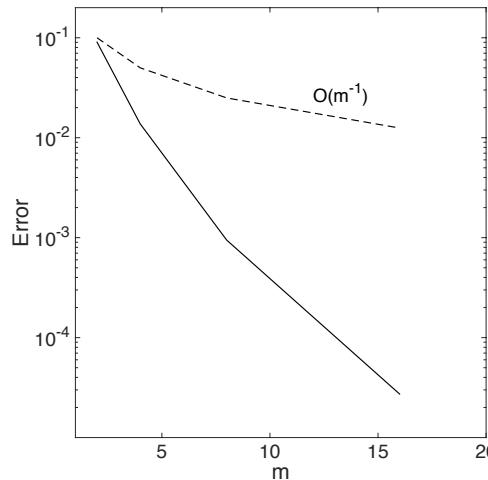
as the initial condition.

The coefficient  $a$  is defined as

$$a(x) = \begin{cases} 1.0, & 1.0 \leq x \leq 2\pi/3 + 1.0, \\ 0.5, & \text{otherwise.} \end{cases} \quad (12.52)$$

Note that the discontinuity does not coincide with an element interface.

The error for the oscillatory moment with  $f(x) = \sin(x)$  is shown in Fig 12.5. Rapid convergence for  $m \rightarrow \infty$  can be observed, in agreement with Theorem 12.17. ■



**Figure 12.5.** Convergence of first oscillatory moment  $(u_h, g_h)$  to  $(u, f)$  with increasing resolution with  $f(x) = \sin(x)$ . The errors obtained without reconstruction (dashed) and accuracy of the results with reconstruction (solid line) are shown.

Let us now turn to the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to a smooth initial condition and periodic boundary conditions.

As established in Theorem 12.8, the discontinuous Galerkin method is unique in that  $L^2$ -stability and an entropy inequality can be established by choosing a monotone flux.

We focus on Burgers' equation and assume for simplicity that  $u$  does not change sign. The semidiscrete scheme is

$$\sum_{j=1}^N \partial_t \int_{D_j} u_j \psi_j dx - \int_{D_j} f(u_j) \partial_x \psi_j dx + F_{j+1/2} \psi_j(x_{j+1/2}) - F_{j-1/2} \psi_j(x_{j-1/2}) = 0. \quad (12.53)$$

For convenience, we consider the Lax–Friedrichs flux

$$F(u, v) = \frac{1}{2} (f(u) + f(v) - \alpha \hat{n}(v - u)), \quad (12.54)$$

where  $\alpha$  is an upper bound of the absolute wave speed. In operator notation we have

$$\partial_t \mathcal{M}(u_b, \psi_b) + \mathcal{F}(u_b, u_b, \psi_b) = 0 \quad \forall \psi_b \in V_b, \quad (12.55)$$

where  $\mathcal{M}$  is defined in (12.39) and

$$\begin{aligned} \mathcal{F}(u, v, w) = & \sum_{j=1}^N - \int_{D_j} f(u, v) \partial_x w dx \\ & + F_{j+1/2}(u, v) w(x_{j+1/2}) - F_{j-1/2}(u, v) w(x_{j-1/2}). \end{aligned} \quad (12.56)$$

Adjoint-consistency for smooth solutions of Burgers' equation is established as follows.

**Lemma 12.19.** *Let  $u$  be a smooth solution to Burgers' equation. The semidiscrete, Lax–Friedrichs-based discontinuous Galerkin scheme for Burgers' equation is adjoint-consistent since*

$$\partial_t \mathcal{M}(\phi_b, v) + \mathcal{F}(\phi_b, u, v) = 0 \quad \forall \phi_b \in V_b, \quad (12.57)$$

for the exact (and smooth) solution  $v$  of

$$\frac{\partial v(x, t)}{\partial t} - u(x, t) \frac{\partial v(x, t)}{\partial x} = 0. \quad (12.58)$$

**Proof.** We choose an arbitrary  $\phi_b \in V_b$  and set  $a = \phi_b$ ,  $b = u$ , and  $c = v$  in (12.39) and (12.56). The volume terms in (12.53) vanish. To show that the numerical flux integral vanishes as well, we consider the flux terms at a unique cell interface to recover

$$\alpha(\phi_b^- - \phi_b^+)(v^- - v^+) = \alpha[\phi_b][v]. \quad (12.59)$$

According to the Rankine–Hugoniot condition,  $v$  is continuous along its characteristics, i.e.  $\alpha(v^- - v^+) = 0$ , which completes the proof.  $\square$

This gives rise to a numerical scheme for the auxiliary problem

$$\frac{\partial v(x, t)}{\partial t} - u(x, t) \frac{\partial v(x, t)}{\partial x} = 0, \quad (12.60)$$

to recover  $v_h \in V_h$  such that

$$\frac{\partial}{\partial t} \mathcal{M}(\phi_h, v_h) - \mathcal{F}(\phi_h, u_h, v_h) = 0 \quad \forall \phi_h \in V_h. \quad (12.61)$$

**Theorem 12.20.** *Let  $u_0 \in C^\infty(\Omega)$ , let  $u_h$  be the  $m$ th order discontinuous Galerkin solution computed with  $N$  elements, and let  $u$  be the entropy solution to a scalar conservation law. If  $u_h$  converges as  $h \rightarrow 0$ , it converges to the entropy solution  $u$ . Furthermore, for every  $f \in C^\infty(\Omega)$  there exists a sequence of functions  $\{g_h\} \in V_h$  such that*

$$\lim_{m \rightarrow \infty} \|g_h - f\|_\Omega \rightarrow 0 \quad (12.62)$$

and

$$|(u_h(x, t), g_h(x))_\Omega - (u(x, t), f(x))_\Omega| \leq C h^{m+1} + C'. \quad (12.63)$$

The functions  $g_h$  are uniquely determined, i.e., there exists a mapping  $\Gamma_h : C^\infty(\Omega) \times C^\infty(\Omega) \rightarrow V_h$  such that  $g_h = \Gamma_h(f, u_0)$ . The constant  $C'$  is given by

$$C' = \sum_e \int_0^t \alpha[[u_h]] [[v_h]] dt. \quad (12.64)$$

**Proof.** Because of the hyperbolic nature of the conservation law, the auxiliary problem (12.60) is hyperbolic as well. Hence, it can be solved exactly backward in time. We apply a Lax-Friedrichs-based approximation to the auxiliary scheme (12.60), and solve it forward in time. The theorem follows by applying the local entropy results and continues as in the proof of Theorem 12.17. As the auxiliary problem depends on the entropy solution  $u$ ,  $g_h = \Gamma_h(f, u_0)$  is now a function of  $f$  and  $u_0$ .  $\square$

This result, unique to the discontinuous Galerkin method, suggests that, in spite of the oscillations, the high-order accurate information remains intact even for nonlinear conservation laws. In other words, removing the oscillations may lead to a poorer solution than retaining them and, subsequently, reconstruct the solution. While the result highlights that the moments maintain accuracy, it does not offer a constructive approach to the recovery of the high-order accurate solution. For this task, other techniques are needed, as we shall discuss below and further in Chapter 13.

Finally, let us consider an example.

**Example 12.21.** We consider the inviscid Burgers' equation

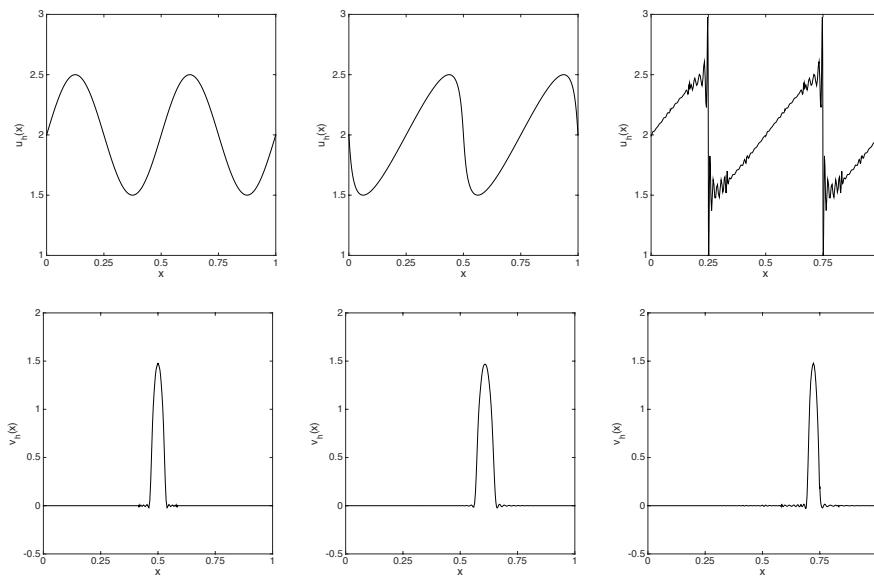
$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

with the initial condition

$$u_0(x) = 2 + 0.5 \sin 2x \quad (12.65)$$

on the periodic domain  $[0, 2\pi]$ . We apply a discontinuous Galerkin scheme with  $N = 12$  elements, and various orders of approximation. For the numerical test, we set

$$v_0(x) = \begin{cases} 0, & |x - c| > \epsilon, \\ \frac{1}{\epsilon} \exp \frac{-1}{1 - ((x - c)/\epsilon)^2}, & \text{otherwise,} \end{cases} \quad (12.66)$$



**Figure 12.6.** Solution of Burgers' equation using a discontinuous Galerkin method with  $N = 12$  elements, each of order  $m = 16$ . The bottom row shows the corresponding solution to the auxiliary solution. The results are shown at  $t = 0$  (left column), at  $t = \pi/8$  (center column), and at  $t = \pi/4$  (right column).

where  $c = \pi$  and  $\epsilon = 0.25$ . The fields,  $u$  and  $v$ , are integrated forward in time until  $u$  develops a shock. Once the solution forms a shock, it is not possible to solve the auxiliary problem exactly, neither forward nor backward in time. Instead, we apply Lemma 12.12 and compute the initial exact moments.

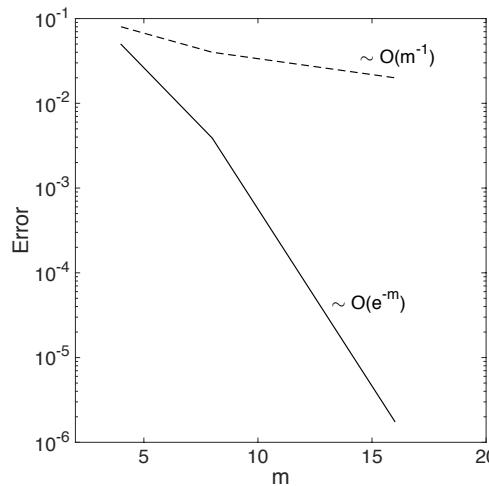
Note that the initial condition (12.66) ensures that the lemma is applicable. In Fig. 12.6 the solutions  $u_h$  and  $v_h$  for  $N = 12$  are shown at  $t = 0, \pi/8, \pi/4$ . Figure 12.7 shows the resulting convergence plot for Burgers' equation, confirming that high-order information is available in this nonlinear and nonsmooth problem. ■

## 12.2.2 • Filtering

As already discussed at the end of section 8.3, filtering can have a positive effect on the stability of numerical schemes. We now extend this discussion to the discontinuous Galerkin method. As we shall see shortly, filtering can, likewise, help to localize the impact of numerical oscillations generated by the use of a high-order polynomial to represent a discontinuous solution.

### Enhancing stability

While the nonlinear stability result for discontinuous Galerkin methods in Theorem 12.8 is very strong, its validity is restricted to polynomial nonlinearities and scalar conservation laws. Among many important applications, the Euler equations are not included in this analysis. Furthermore, the cost of an exact projection of the nonlinear flux onto the polynomial basis can be high. An attractive way to overcome this latter



**Figure 12.7.** Convergence of first oscillatory moment  $(u_h, g_h)_\Omega$  to  $(u, f)_\Omega = (u_0, v_0)_\Omega$  with increasing resolution. Recovery at  $t = \pi/4$ . The solid line represents the accuracy of the recovered moment while the dashed line represents the pointwise accuracy of the solution.

issue is to use a collocation approach, i.e., by evaluating the nonlinear flux at the nodal points, leading to the introduction of aliasing errors. As reasonable as this sounds, it is not without challenges and may ultimately result in an unstable scheme. To gain insight into this, consider [42]

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(a(x)u) = 0, \quad x \in [-1, 1],$$

and assume  $a(x) > 0$  and smooth. For simplicity, but without loss of generality, we direct our attention to one element with periodic boundary conditions and consider the scheme

$$\frac{\partial u_h}{\partial t} + \frac{\partial}{\partial x}\mathcal{I}_m(a u_h) = 0,$$

where  $\mathcal{I}_m$  is the interpolation operator. We rewrite this as

$$\begin{aligned} \frac{\partial u_h}{\partial t} + \frac{1}{2} \frac{\partial}{\partial x} \mathcal{I}_m(a u_h) + \frac{1}{2} \mathcal{I}_m \left( a \frac{\partial u_h}{\partial x} \right) \\ + \frac{1}{2} \mathcal{I}_m \frac{\partial}{\partial x} a u_h - \frac{1}{2} \mathcal{I}_m \left( a \frac{\partial u_h}{\partial x} \right) \\ + \frac{1}{2} \frac{\partial}{\partial x} \mathcal{I}_m(a u_h) - \frac{1}{2} \mathcal{I}_m \frac{\partial}{\partial x} a u_h = 0, \end{aligned}$$

and express it as

$$\frac{\partial u_h}{\partial t} + \mathcal{N}_1 + \mathcal{N}_2 + \mathcal{N}_3 = 0. \quad (12.67)$$

For simplicity we use a central flux, in which case the nodal discontinuous Galerkin scheme becomes

$$\int_{-1}^1 \left( \frac{\partial u_b}{\partial t} + \mathcal{N}_1 + \mathcal{N}_2 + \mathcal{N}_3 \right) \ell_i(x) dx = [\mathcal{I}_m(a u_b)] \{\ell_i\}. \quad (12.68)$$

Multiplying by  $u(x_i)$  from the left and summing over all points yields

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_{\Omega}^2 = -(\mathcal{N}_1, u_b)_{\Omega} - (\mathcal{N}_2, u_b)_{\Omega} - (\mathcal{N}_3, u_b)_{\Omega} + [\mathcal{I}_m(a u_b)] \{u_b\}. \quad (12.69)$$

We consider the three terms individually, beginning with

$$\begin{aligned} & -(\mathcal{N}_1, u_b)_{\Omega} + [\mathcal{I}_m(a u_b)] \{u_b\} \\ &= -\frac{1}{2} \left( \frac{\partial}{\partial x} \mathcal{I}_m(a u_b), u_b \right)_{\Omega} - \frac{1}{2} \left( \mathcal{I}_m \left( a \frac{\partial u_b}{\partial x} \right), u_b \right)_{\Omega} + [\mathcal{I}_m(a u_b)] \{u_b\} \\ &= \frac{1}{2} \left( \mathcal{I}_m(a u_b), \frac{\partial}{\partial x} u_b \right)_{\Omega} - \frac{1}{2} \left( \mathcal{I}_m \left( a \frac{\partial u_b}{\partial x} \right), u_b \right)_{\Omega} = 0. \end{aligned}$$

The boundary terms vanish after integration by parts since  $a(x)$  is smooth and the volume terms cancel by the interpolation property.

For the second term, we have

$$\begin{aligned} -2(\mathcal{N}_2, u_b)_{\Omega} &= - \left( \mathcal{I}_m \left( \frac{\partial}{\partial x} a u_b \right), u_b \right)_{\Omega} + \left( \mathcal{I}_m \left( a \frac{\partial}{\partial x} u_b \right), u_b \right)_{\Omega} \\ &= - \left( \mathcal{I}_m \left( a \frac{\partial}{\partial x} u_b \right), u_b \right)_{\Omega} - (\mathcal{I}_m(a_x u_b), u_b)_{\Omega} + \left( \mathcal{I}_m \left( a \frac{\partial}{\partial x} u_b \right), u_b \right)_{\Omega} \\ &= -(\mathcal{I}_m(a_x u_b), u_b)_{\Omega} \leq \max_x |a_x| \|u_b\|_{\Omega}^2. \end{aligned}$$

This term cannot drive an instability, as it is bounded independently of  $m$ . Finally, let us consider the third term

$$\begin{aligned} -2(\mathcal{N}_3, u_b)_{\Omega} &= - \left( \frac{\partial}{\partial x} \mathcal{I}_m(a u_b), u_b \right)_{\Omega} + \left( \mathcal{I}_m \frac{\partial}{\partial x} a u_b, u_b \right)_{\Omega} \\ &= \left( \mathcal{I}_m \frac{\partial}{\partial x} a u_b - \frac{\partial}{\partial x} \mathcal{I}_m(a u_b), u_b \right)_{\Omega} \\ &\leq \left\| \mathcal{I}_m \frac{\partial}{\partial x} a u_b - \frac{\partial}{\partial x} \mathcal{I}_m(a u_b) \right\|_{\Omega}^2 + \|u_b\|_{\Omega}^2. \end{aligned}$$

The last contribution causes no problems. The first one can be bounded as

$$\left\| \mathcal{I}_m \frac{d}{dx} v - \frac{d}{dx} \mathcal{I}_m v \right\|_{\Omega}^2 \leq \left\| \frac{d}{dx} v - \mathcal{I}_m \frac{d}{dx} v \right\|_{\Omega}^2 + \left\| \frac{d}{dx} v - \frac{d}{dx} \mathcal{I}_m v \right\|_{\Omega}^2.$$

Using a slightly refined version of the estimates in subsection 12.1.1 [12], assuming that  $v \in H^p(\Omega)$ , we have

$$\left\| \frac{d}{dx} v - \mathcal{I}_m \frac{d}{dx} v \right\|_{\Omega} \leq C \frac{h^{\sigma-1}}{m^{p-1}} |v|_p,$$

when  $\sigma = \min(m+1, p)$ . In a similar way, we have

$$\left\| \frac{d}{dx} v - \frac{d}{dx} \mathcal{I}_m v \right\|_{\Omega} \leq \|v - \mathcal{I}_m v\|_1 \leq C \frac{h^{\sigma-1}}{m^{p-1}} |v|_p,$$

which yields [42]

$$\frac{1}{2} \frac{d}{dt} \|u_h\|_{\Omega} \leq C_1 \|u_h\|_{\Omega} + C_2(h, a) m^{1-p} |u|_p.$$

This implies that, if  $u$  is not sufficiently smooth, e.g.,  $p \leq 1$ , we cannot control the last term and the scheme may be unstable. This is clearly an issue for conservation laws and problems with discontinuous solutions.

The term driving the instability appears since interpolation and differentiation do not commute. This is a result of aliasing and this type of instability is referred to as an aliasing-driven instability. If  $u$  is smooth but under-resolved, adding resolution eliminates the instability by reducing the commutation error. A further discussion of these aspects can be found in [42, 53].

With the understanding of this aliasing-driven instability, we seek a more practical solution than simply adding more resolution. As we have already discussed at some length, a classic approach to overcome a relatively weak instability is to add dissipation to the problem. To understand whether this remedy suffices, consider the modified problem

$$\frac{\partial u_j}{\partial t} + \frac{\partial}{\partial x} \mathcal{I}_m(a u_j) = \varepsilon (-1)^{s+1} \left[ \frac{\partial}{\partial x} (1-x^2) \frac{\partial}{\partial x} \right]^s u_j, \quad x \in [-1, 1]. \quad (12.70)$$

Note that the right-hand side is a local operation, restricted to each element  $D_j$ . The motivation behind its particular form will become clear shortly. Let us now consider the term

$$\varepsilon (-1)^{s+1} \left( u_j, \left[ \frac{\partial}{\partial x} (1-x^2) \frac{\partial}{\partial x} \right]^s u_j \right)_{D_j},$$

which enters into the energy statement (12.69) through the extra term in (12.70). Integration by parts  $s$  times yields

$$\varepsilon (-1)^{s+1} \left( u_j, \left[ \frac{\partial}{\partial x} (1-x^2) \frac{\partial}{\partial x} \right]^s u_j \right)_{D_j} = -\varepsilon \|u_j^{(s)}\|_{D_j}^2 = -\varepsilon |u_j|_{D_j, s}^2.$$

Now the reason for the special form of the operator becomes clear as the singular nature of the operator ensures that no boundary terms are introduced. We recover the modified energy statement

$$\frac{1}{2} \frac{d}{dt} \|u_j\|_{D_j}^2 \leq C_1 \|u_j\|_{D_j}^2 + C_2 m^{2-2p} |u|_{D_j, s}^2 - C_3 \varepsilon |u_j|_{D_j, s}^2.$$

Taking  $\varepsilon \propto m^2$  is sufficient, but may not be necessary, to ensure that the dissipative term dominates the unstable term, thus restoring stability.

A remaining question is how to implement the extra term in (12.70) in the most efficient way. To reduce the computational overhead, we include it in a time-splitting fashion, i.e., we first advance the conservation law

$$\frac{\partial u_j}{\partial t} + \frac{\partial}{\partial x} \mathcal{I}_m f(u_j) = 0$$

one time step, followed by

$$\frac{\partial u_j}{\partial t} = \varepsilon(-1)^{s+1} \left[ \frac{\partial}{\partial x} (1-x^2) \frac{\partial}{\partial x} \right]^s u_j. \quad (12.71)$$

We restrict our attention to this latter problem and advance it by time step  $k$  with a forward Euler method:

$$u_j^* = u_j(x, t+k) = u_j(x, t) + \varepsilon k (-1)^{s+1} \left[ \frac{\partial}{\partial x} (1-x^2) \frac{\partial}{\partial x} \right]^s u_j(x, t). \quad (12.72)$$

Recall now that

$$u_j(x, t) = \sum_{i=0}^m \tilde{u}_{j,i}(t) P_i(x).$$

Since  $P_i$  satisfies (12.15) we obtain

$$\begin{aligned} u_j^*(x, t) &\simeq u_j(x, t) - \varepsilon k \sum_{i=0}^m \tilde{u}_{j,i}(t) (i(i+1))^s P_i(x) \\ &\simeq \sum_{i=0}^m \sigma\left(\frac{i}{m}\right) \tilde{u}_i(t) P_i(x), \quad \varepsilon \propto \frac{1}{k m^{2s-2}}. \end{aligned}$$

Here, we have introduced the filter function  $\sigma(\eta)$ , discussed at length in section 8.3. Following Definition 8.20, suitable choices for the filter function include

$$\sigma(\eta) = 1 - \alpha \eta^{2s}$$

and

$$\sigma(\eta) = \exp(-\alpha \eta^{2s}),$$

known as an exponential filter. Both filters are of order  $2s$ .

The actual choice of  $\alpha$  is somewhat arbitrary. If we choose  $\alpha = 0$ , no dissipation is introduced. On the other hand, taking  $\alpha$  large results in substantial dissipation. An often used choice is  $\alpha \simeq -\log(\varepsilon_M) \simeq 36$ , where  $\varepsilon_M$  is the machine double precision. This choice implies that  $\sigma(1) \simeq \varepsilon_M$  in the exponential filter. Naturally, this can also be approached in an adaptive manner [36] by defining an indicator that controls the local strength of the filter.

There is one very important difference between solving (12.71) and applying the filter. In the latter case, no temporal integration is required and, thus, no additional stability constraint is needed in contrast to what is required to directly solve (12.71). Thus, the use of the filter is a practical way to introduce dissipation.

In the general formulation developed in subsection 12.1.1, filtering is equivalent to multiplication with the filter matrix  $\hat{F}$ , defined as

$$\hat{F} = \hat{V} \Lambda \hat{V}^{-1},$$

where the diagonal matrix,  $\Lambda$ , has entries

$$\Lambda_{ii} = \sigma\left(\frac{i}{m}\right), \quad i = 0, \dots, m.$$

A script for defining this, assuming a filter of the form

$$\sigma(\eta) = \begin{cases} 1, & 0 \leq \eta \leq \eta_c = \frac{m_c}{m}, \\ \exp(-\alpha((\eta - \eta_c)/(1 - \eta_c))^p), & \eta_c \leq \eta \leq 1, \end{cases} \quad (12.73)$$

is shown in `FilterDG.m`. To simplify matters,  $p = 2s$  must be even and we choose  $\alpha = -\log(\varepsilon_M)$ . The parameter  $m_c$  represents a cutoff below which the modes are left untouched.

**Script 12.7.** *FilterDG.m: Definition of a filter matrix based on an exponential filter of order  $p$  with a cutoff at  $m_c$ .*

---

```
function [F] = FilterDG(m,mc,p,V)
% function [F] = FilterDG(m,mc,p,V)
% Purpose : Initialize DG filter matrix of size m.
% Order of exponential filter is (even) p with cutoff at mc;
filterdiag = ones(m+1,1); alpha = -log(eps);

% Initialize filter function
for i=mc:m
    filterdiag(i+1) = exp(-alpha*((i-mc)/(m-mc))^p);
end;
F = V*diag(filterdiag)*inv(V);
return;
```

---

To gain some additional insight into the impact of the filter, let us consider an example.

**Example 12.22.** Consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [-1, 1],$$

with the initial condition

$$u(x, 0) = 2 - H(x + 0.75),$$

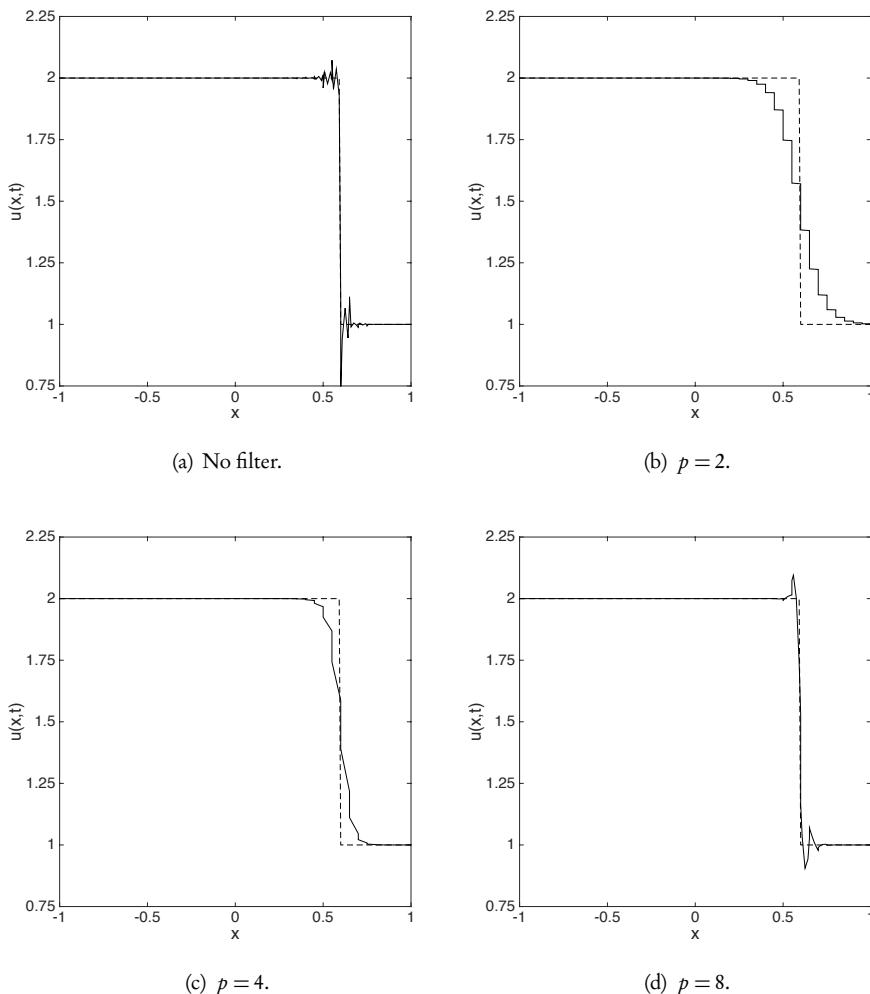
i.e., a shock at  $x = -0.75$ . Its exact solution is

$$u(x, 0) = 2 - H(x + 0.75 - 3t).$$

We solve the problem using a nodal discontinuous Galerkin method of order  $m = 4$  with  $N = 20$  cells. A Lax-Friedrichs flux and a third order SSPERK are used for the time-stepping scheme.

Figure 12.8 shows the results obtained for different orders of the filter. We observe that by increasing the order of the filter, the smearing of the shock is reduced very substantially. Since a  $p = 2$  filter corresponds to a second order operator, the maximum principle ensures elimination of the oscillations. Although the unfiltered scheme in this case is stable, the oscillations in the solution are very substantial. Higher-order filtering helps to localize and limit such oscillations.

In this example, the filter matrix is applied to the solution vector after each Runge-Kutta step. However, depending on the problem, this can be modified e.g., filtering can be applied after each stage or after a certain number of time steps as needed. ■



**Figure 12.8.** Impact of filtering on the solution of Burgers' equation with a propagating shock. In the upper left corner is the unfiltered solution, obtained with  $m = 4$  and  $N = 20$ . The other panels show results obtained with increasing order  $p$  of the filter. The dashed line indicates the exact solution.

The amount of filtering required for stabilization is clearly problem-dependent. For example, problems with underresolved features, shocks, or highly nonlinear terms are likely to suffer more from instabilities than simpler types of problem. Nevertheless, for specific problems, it is often relatively easy to choose suitable parameters, e.g., one starts with a small amount of filtering and then increases it over a few short runs until the scheme is stabilized. For a reasonably resolved computation, a good set of starting parameters is  $\alpha = 18 - 36$ ,  $N_c = 0$ , and  $p = 10 - 16$ .

If we take a second look at the operator associated with filtering

$$\left[ \frac{d}{dx} (1-x^2) \frac{d}{dx} \right]^s = \left[ (1-x^2) \frac{d^2}{dx^2} - 2x \frac{d}{dx} \right]^s, \quad x \in [-1, 1],$$

it reveals that the dissipation is added in a nonuniform way; for example, it is most effective near the center of the element and weaker as the edges of the element are approached. This introduces the possibility of over-dissipating in the center to ensure stability throughout the element [54].

Since we are now solving a modified problem (12.70), we need to revisit the question of whether  $u_b$  converges to a weak solution of the conservation law [16]. Let us assume that the local modification is

$$\left( \phi_j, (-1)^{s+1} \frac{1}{m^{2s-1}} \left[ \frac{d}{dx} (1-x^2) \frac{d}{dx} \right]^s u_j \right)_{D_j},$$

where  $\phi_j$  represents a compactly supported smooth test function. To maintain convergence to the weak solution, we must ensure that this term vanishes as  $m \rightarrow \infty$ . Let us first recall the definition of  $u_j$  and  $\phi_j$  as

$$\phi_j(x, t) = \sum_{i=0}^m \tilde{\phi}_{j,i}(t) P_i(x), \quad u_j(x, t) = \sum_{i=0}^m \tilde{u}_{j,i}(t) P_i(x).$$

The local modification then becomes

$$-\frac{1}{m^{2s-1}} \sum_{i=0}^m \tilde{\phi}_{j,i}(t) \tilde{u}_{j,i}(t) (i+1)^s i^s,$$

since  $P_i$  satisfies (12.15) and is orthonormal.

Furthermore, if  $u_j$  is bounded, i.e., the scheme is stable, Theorem 12.2 ensures

$$|\tilde{u}_{j,i} \tilde{\phi}_{j,i}| \leq C i^{-3},$$

provided that  $\phi_j$  is at least continuous. Then

$$\frac{1}{m^{2s-1}} \sum_{i=0}^m \frac{(i+1)^s i^s}{i^3} \leq \frac{C}{m}.$$

Thus, the stabilizing term vanishes and, by the Lax–Wendroff theorem, we recover convergence to the weak solution. The extension to  $N$  elements is straightforward.

### Enhancing accuracy

While the use of a filter has a stabilizing effect on the discontinuous Galerkin method, the results in Fig. 12.8 suggest that the overall accuracy of the approximation improves with use of the filter and helps to contain the impact to the neighborhood of the discontinuity. To further investigate this, consider the following example.

**Example 12.23.** Since the accuracy improvement is a question of approximation, let us consider the function

$$u(x) = 1 - 2H(x), \quad x \in [-1, 1],$$

where  $H(x)$  is the Heaviside function. We approximate this by a nodal representation

$$u(x) \simeq u_b(x) = \sum_{i=0}^m \tilde{u}_i P_i(x).$$

To understand the impact of the filter, we show in Fig. 12.9 the approximation and the pointwise error for the function and its filtered approximation for increasing resolution. We use filters of orders of 2, 4, and 8. Although all examples use an exponential filter, results with other types of filters are qualitatively similar.

The results in Fig. 12.9 allow us to make a few general observations. First of all, we observe that without the filter, the order of accuracy of the local approximation reduces to first order and, as expected, convergence is lost at the point of the discontinuity.

Applying the filter has a profound impact on the accuracy of the global approximation. While the pointwise accuracy of the approximation does not improve at the point of discontinuity, increasing the order of the filter dramatically improves the accuracy away from the discontinuity. Furthermore, there appears to be a close relation between the order of the filter and the observed order of convergence away from the discontinuity. However, it is also clear that, when  $m$  is small, the use of even a weak filter can adversely impact the overall accuracy. In this case, the benefits of applying the filter are less clear. ■

In section 8.3 we discussed the impact of filtering on periodic high-order representations and made observations very similar to those emerging from Ex. 12.23. In that case, the periodic nature of the approximation allows for the development of a relatively complete theory [109] which supports the numerical experiments. We shall revisit this in more detail in Chapter 13.

While the observations made for Ex. 12.23 are qualitatively similar to those in section 8.3, the theoretical understanding of the impact of filtering on polynomial approximations is largely open, with the only known results in [54]. These results confirm that the filter has no impact at the point of discontinuity and does not destroy the accuracy when applied to a smooth function. The pointwise error away from the point of discontinuity, located at  $x = \xi$ , is conjectured to behave as

$$|u(x) - \mathcal{F}u_h(x)| \leq C(x, \xi) m^{1-r} \|\|u\|\|_r,$$

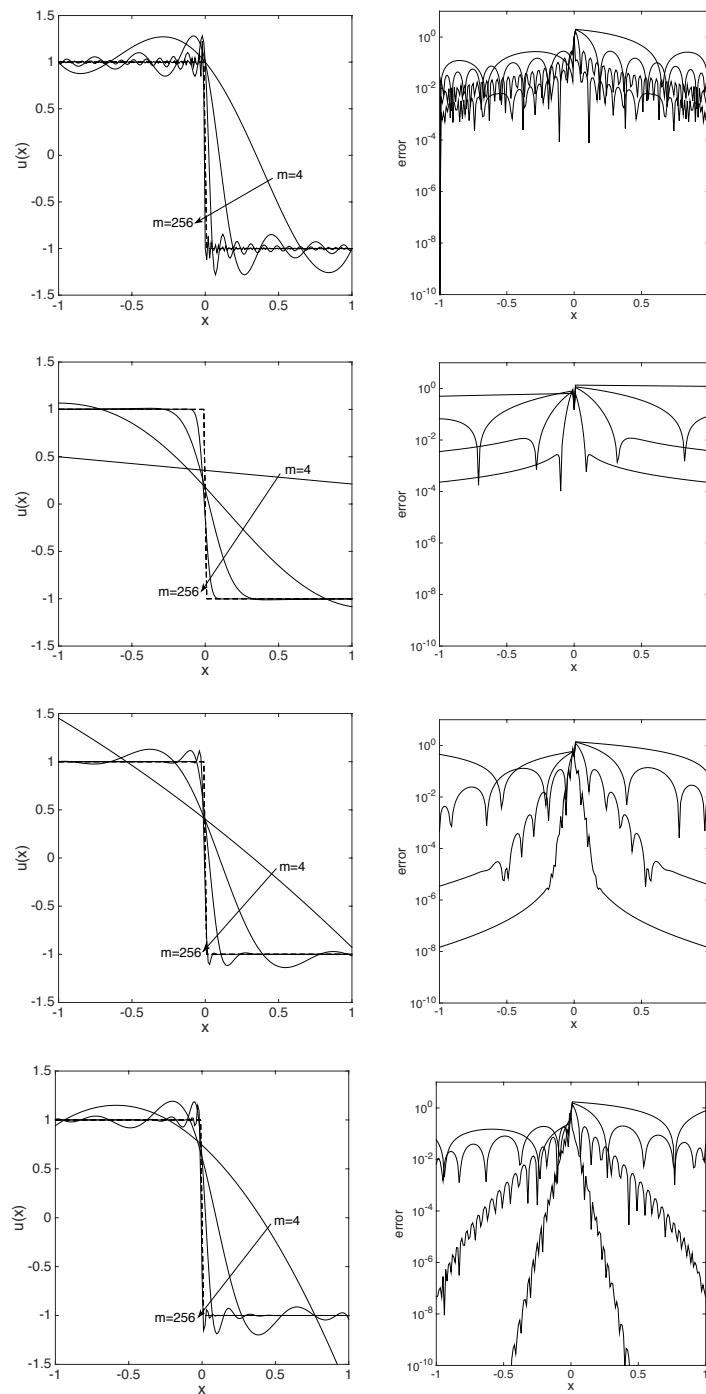
where  $C(x, \xi)$  depends on the distance to the point of discontinuity. Here  $\|\cdot\|_r$ , for  $r > 1$  is a piecewise Sobolev norm and  $r = \min(p, q)$ , where  $p$  is the filter order and  $u \in H^q$ . Thus, if the filter order is sufficiently high, the approximation of a smooth function will not be impacted.

### 12.2.3 • Nonlinear dissipation

While the use of filtering has a number of attractive features, such as simplicity, enhanced stability, and accuracy, the linearity of the approach adversely impacts its efficiency for complex problems. For instance, in the case of a problem comprising shocks and highly oscillatory yet smooth features, filtering typically results in an overly dissipative solution due to the need to reduce oscillations in the neighborhood of shocks. In many ways, this is similar to what we observe for monotone schemes and other low-order schemes.

A possible way to address this concern is through a more localized nonlinear dissipation. To describe the basic concept, consider

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0,$$



**Figure 12.9.** Impact of filtering on the representation of a nonsmooth function. The rows are obtained with no filter and filters of order 2, 4, and 8, respectively. In the left column we show the polynomial approximation for orders of approximation  $m$  being 4, 16, 64, and 256, respectively. In the right column is shown the corresponding pointwise error. In the latter figures, the error decreases with increasing order of approximation  $m$  in each figure.

subject to appropriate initial and boundary conditions. We introduce local dissipation as

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = \frac{\partial}{\partial x} \nu(u) \frac{\partial u}{\partial x}, \quad (12.74)$$

where  $\nu(u) \geq 0$  is a locally determined function which is selected to control oscillations in nonsmooth regions, and vanishes in smooth regions. Although the second order operator ensures a maximum principle, it must be defined carefully to avoid an overly aggressive dissipation.

One potential challenge with such an approach is that the type of the equation is changed. Consequently, the number of boundary conditions needed to ensure well-posedness must be modified. As an example, consider  $f(u) = au$ . Whereas the original problem needs only one boundary condition, the stabilized version (12.74) requires two boundary conditions. Since this is a consequence of the choice of the numerical scheme, it may not be entirely clear how to define these conditions without impacting the solution. One solution is to impose homogeneous Neumann conditions where new conditions are needed, i.e., at outflow boundaries where no physical boundary condition is available.

As simple as the idea of adding artificial dissipation appears, it raises two fundamental questions. On one hand, we need to understand how a second order operator can be implemented in a discontinuous Galerkin method which, so far, has only been formulated for first order problems. Furthermore, we need to identify suitable ways to define  $\nu(u)$  in a simple, yet efficient, manner.

### Interlude on dissipative operators

To keep the discussion simple, let us consider the heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x} a(x) \frac{\partial u}{\partial x}$$

and impose  $u = 0$  at the outer boundaries. The coefficient  $a$  is often referred to as the coefficient of diffusivity or viscosity. It is easily shown that  $a(x) \geq 0$  suffices to guarantee wellposedness [46].

By rewriting the second order problem as a system of first order problems:

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x} \sqrt{a} q, \quad q = \sqrt{a} \frac{\partial u}{\partial x},$$

we can employ techniques for the conservation laws and assume that  $(u, q)$  can be approximated as

$$\begin{bmatrix} u(x, t) \\ q(x, t) \end{bmatrix} \approx \begin{bmatrix} u_b(x, t) \\ q_b(x, t) \end{bmatrix} = \bigoplus_{j=1}^N \begin{bmatrix} u_j(x, t) \\ q_j(x, t) \end{bmatrix},$$

where we represent  $(u, q)$  by  $m$ th order piecewise polynomials over  $N$  elements. We recover the strong form

$$\frac{h_j}{2} \hat{M} \frac{d u_j}{d t} = \hat{S}_j^{\sqrt{a}} q_j - \left[ (\sqrt{a} q_j - Q) \psi_j \right]_{x_{j-1/2}}^{x_{j+1/2}}, \quad (12.75)$$

$$\frac{h_j}{2} \hat{M} q_j = S_j^{\sqrt{a}} u_j - \left[ (\sqrt{a} u_j - F) \psi_j \right]_{x_{j-1/2}}^{x_{j+1/2}},$$

where  $F(u, q)$  is the numerical flux for  $u$  and  $Q(u, q)$  is the numerical flux for  $q$ . The corresponding weak form is obtained as

$$\begin{aligned} \frac{h_j}{2} \hat{\mathbf{M}} \frac{d\mathbf{u}_j}{dt} &= -(\mathbf{S}_j^{\sqrt{a}})^T \mathbf{q}_j + [Q\psi_j]_{x_{j-1/2}}^{x_{j+1/2}}, \\ \frac{h_j}{2} \hat{\mathbf{M}} \mathbf{q}_j &= -(\tilde{\mathbf{S}}_j^{\sqrt{a}})^T \mathbf{u}_j + [F\psi_j]_{x_{j-1/2}}^{x_{j+1/2}}. \end{aligned} \quad (12.76)$$

The two operators  $\mathbf{S}_j^{\sqrt{a}}$  and  $\tilde{\mathbf{S}}_j^{\sqrt{a}}$  are defined as

$$(\tilde{\mathbf{S}}_j^{\sqrt{a}})_{ik} = \int_{D_j} \psi_{j,i}(x) \frac{d\sqrt{a(x)}\psi_{j,k}(x)}{dx} dx, \quad (\mathbf{S}_j^{\sqrt{a}})_{ik} = \int_{D_j} \sqrt{a(x)}\psi_{j,i}(x) \frac{d\psi_{j,k}(x)}{dx} dx.$$

They are closely connected since

$$(\tilde{\mathbf{S}}_j^{\sqrt{a}})_{ik} + (\mathbf{S}_j^{\sqrt{a}})_{ki} = [\sqrt{a(x)}\psi_{j,i}(x)\psi_{j,k}(x)]_{x_{j-1/2}}^{x_{j+1/2}}.$$

To define the numerical flux we recall that, in general, it can have dependencies as

$$\begin{aligned} Q(u, q) &= Q((\sqrt{a}q_j)^-, (\sqrt{a}q_j)^+, (\sqrt{a}u_j)^-, (\sqrt{a}u_j)^+), \\ F(u, q) &= F((\sqrt{a}q_j)^-, (\sqrt{a}q_j)^+, (\sqrt{a}u_j)^-, (\sqrt{a}u_j)^+). \end{aligned}$$

This general form implies that the two first order equations in (12.75) or (12.76) are coupled through the numerical flux and, hence, must be solved as a global system. In this case, the local nature of the discontinuous Galerkin method is destroyed.

However, if we restrict the form of the numerical flux as

$$\begin{aligned} Q(u, q) &= Q((\sqrt{a}q_j)^-, (\sqrt{a}q_j)^+, (\sqrt{a}u_j)^-, (\sqrt{a}u_j)^+), \\ F(u) &= F((\sqrt{a}u_j)^-, (\sqrt{a}u_j)^+), \end{aligned}$$

then  $q_j$  can be recovered in a local element-based operation. In other words, the auxiliary function  $q$  is a truly local variable, which is used to express derivatives and impose boundary conditions in an elementwise fashion.

Since the heat equation has no preferred direction of propagation, it is natural to consider the central flux

$$Q(u, q) = \{\{\sqrt{a}q\}\}, \quad F(u) = \{\{\sqrt{a}u\}\}.$$

Semidiscrete stability of this scheme is stated as follows

**Theorem 12.24.** *The discontinuous Galerkin scheme for the heat equation with central fluxes is stable.*

**Proof.** Let us first consider the situation over element  $D_j = [x_{j-1/2}, x_{j+1/2}]$ . We introduce the two functions  $(\phi_j, \pi_j) \in V_b$  and define the local, elementwise operator

as

$$\begin{aligned}\mathcal{B}_b(u_j, q_j; \phi_j, \pi_j) &= \phi_j^T \frac{b_j}{2} \hat{M} \frac{d}{dt} u_j - \phi_j^T \hat{S}_j^{\sqrt{a}} q_j + [\phi_j(x)(\sqrt{a}q_j - Q)]_{x_{j-1/2}}^{x_{j+1/2}} \\ &\quad + \pi_j^T \frac{b_j}{2} \hat{M} q_j - \pi_j^T \hat{S}_j^{\sqrt{a}} u_j + [\pi_j(x)(\sqrt{a}u_j - F)]_{x_{j-1/2}}^{x_{j+1/2}}.\end{aligned}$$

We note that if  $(u_j, q_j)$  satisfy the numerical scheme, then

$$\mathcal{B}_b(u_j, q_j; \phi_j, \pi_j) = 0, \quad \forall (\phi_j, \pi_j) \in V_b.$$

If we now choose the test functions as  $\phi_j = u_j$ ,  $\pi_j = q_j$  and exploit that

$$u_j^T \hat{S}_j^{\sqrt{a}} q_j + q_j^T \hat{S}_j^{\sqrt{a}} u_j = u_j^T \hat{S}_j^{\sqrt{a}} q_j + u_j^T (\hat{S}_j^{\sqrt{a}})^T q_j = [\sqrt{a} u_j q_j]_{x_{j-1/2}}^{x_{j+1/2}},$$

we recover

$$\frac{1}{2} \frac{d}{dt} \|u_j\|_{D_j}^2 + \|q_j\|_{D_j}^2 + \Theta_{j+1/2} - \Theta_{j-1/2} = 0,$$

where

$$\Theta = \sqrt{a} u_j q_j - Q(u_j, q_j) u_j - F(u_j) q_j.$$

Let us first assume that  $a$  is continuous. Then the central fluxes can be written as

$$Q(u_j, q_j) = \sqrt{a} \{ \{ q_j \} \}, \quad F(u_j) = \sqrt{a} \{ \{ u_j \} \}.$$

If we consider the interface  $x_{j+1/2}$ , we recover the term

$$\Theta_{j+1/2} = -\frac{\sqrt{a}}{2} (u_j^- q_j^+ + u_j^+ q_j^-).$$

Summation over all elements, yields

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_b^2 + \|q_b\|_b^2 = 0$$

and, thus, stability.

For the general case of  $a$  being only piecewise smooth, we consider the numerical flux

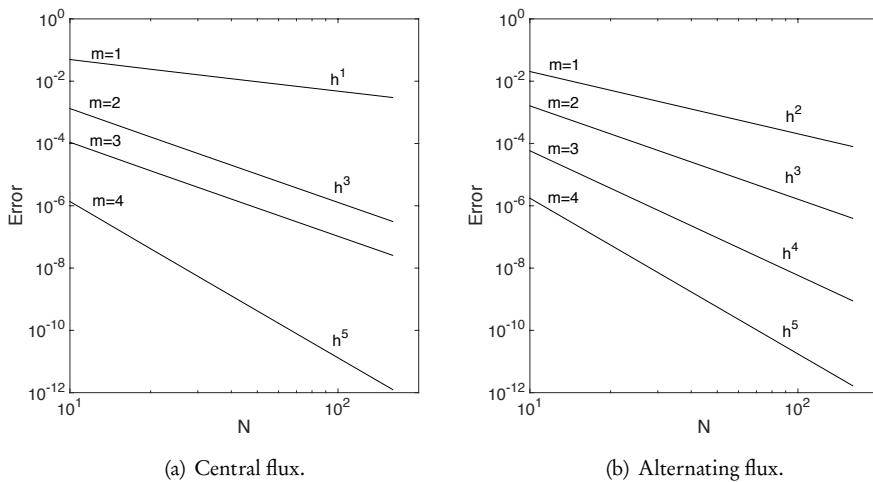
$$Q(u_j, q_j) = \{ \{ \sqrt{a} q_j \} \} + \frac{1}{2} [\![ \sqrt{a} ]\!] q_j^+, \quad F(u_j) = \{ \{ \sqrt{a} u_j \} \} + \frac{1}{2} [\![ \sqrt{a} ]\!] u_j^+,$$

which reduces to the central flux for the continuous case. This yields

$$\Theta_{j+1/2} = -\frac{1}{2} \{ \{ \sqrt{a} \} \} (u_j^- q_j^+ + u_j^+ q_j^-)$$

and, hence, after summation over all elements, stability.  $\square$

We highlight the behavior of this scheme, through an example.



**Figure 12.10.** Convergence of the discontinuous Galerkin method when solving the heat equation with different choices of the numerical flux.

**Example 12.25.** Consider the heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad x \in [0, 2\pi],$$

with homogeneous Dirichlet boundary conditions and the initial condition  $u(x, 0) = \sin(x)$ . The exact solution is easily found to be  $u(x, t) = e^{-t} \sin(x)$ .

We use a discontinuous Galerkin method with  $N$  elements of order  $m$ , along with a third order SSPERK scheme with a small time step to avoid temporal errors. Figure 12.10 displays the global error as a function of both  $m$  and  $N$ .

Although the scheme converges, the rate of convergence varies in unexpected ways. Indeed, for  $m$  odd, the rate is limited to  $\mathcal{O}(h^m)$ , while the optimal rate  $\mathcal{O}(h^{m+1})$  is recovered when  $m$  is even. ■

The even-odd pattern is confirmed by the following theorem [29].

**Theorem 12.26.** Let  $\varepsilon_u = u - u_h$  and  $\varepsilon_q = q - q_h$  represent the pointwise errors for the heat equation with a periodic boundary and a constant diffusivity  $a$ , computed with (12.76) and central fluxes. Then it holds

$$\|\varepsilon_u(T)\|_h^2 + \int_0^T \|\varepsilon_q(s)\|_h^2 ds \leq C h^{2m},$$

where  $C$  depends on  $T$ ,  $m$ , and the regularity of  $u$ . For  $m$  even,  $C$  behaves as  $\mathcal{O}(h^2)$ .

The proof, which is technical, can be found in [29]. The results of Ex. 12.25 confirm that the theorem is sharp. We also note that the approximation errors for  $u$  and  $q$  are of the same order.

The loss of optimal convergence suggests that one seeks an alternative formulation. When pursuing this, we should keep in mind that we have considerable freedom in

the choice of the numerical flux. In particular, we can use stability considerations to guide this choice. Additionally, we recall that the use of upwind fluxes often results in a scheme with an optimal convergence.

In the proof of Theorem 12.24 we recover a term like

$$\Theta = \sqrt{a} u_j q_j - Q(u_j, q_j) u_j - F(u_j) q_j$$

at each interface, and seek to ensure

$$\Theta_{j+1/2}^+ - \Theta_{j+1/2}^- \geq 0$$

to guarantee stability. One easily shows that this is guaranteed by the flux choice

$$F(u_j) = \sqrt{a^-} u_j^-, \quad Q(u_j, q_j) = \{\{\sqrt{a}\}\} q_j^+$$

or, alternatively, by

$$F(u_j) = \{\{\sqrt{a}\}\} u_j^+, \quad Q(u_j, q_j) = \sqrt{a^-} q_j^-.$$

A slightly more symmetric expression is [29]

$$F(u_j) = \{\{\sqrt{a} u_j\}\} + \hat{\beta} \cdot [\llbracket \sqrt{a} u_j \rrbracket], \quad Q(u_j, q_j) = \{\{\sqrt{a} q_j\}\} - \hat{\beta} \cdot [\llbracket \sqrt{a} q_j \rrbracket].$$

We can take  $\hat{\beta}$  to be  $\hat{n}$  or  $-\hat{n}$ . The essential property is the difference in sign between the two fluxes.

This definition of the fluxes, leading to methods known as local discontinuous Galerkin (LDG) methods, is surprising as it relies on upwinding for the heat equation. However, upwinding is done in a careful manner, always doing upwinding for  $u_h$  and downwinding for  $q_h$  or vice versa. This is the essential element that ensures stability.

As elegant as this construction appears, its motivation is to improve the convergence rate when  $m$  is odd. This is confirmed by the following theorem, the proof of which can be found in [29].

**Theorem 12.27.** *Let  $\varepsilon_u = u - u_h$  and  $\varepsilon_q = q - q_h$  represent the pointwise errors for the heat equation with a periodic boundary and a constant diffusivity  $a$ , computed with (12.76) and alternating fluxes. Then, it holds that*

$$\|\varepsilon_u(T)\|_h^2 + \int_0^T \|\varepsilon_q(s)\|_h^2 ds \leq C h^{2m+2},$$

where  $C$  depends  $T$ ,  $m$ , and the regularity of  $u$ .

Figure 12.10 also shows the results for the heat equation obtained with the alternating flux, and confirms that the optimal order of accuracy is restored for all values of  $m$ .

In `HeatDGrhs.m` we illustrate the implementation of the discontinuous Galerkin method for the heat equation with a general diffusivity/viscosity coefficient, defining the numerical flux, and imposing the boundary conditions.

---

**Script 12.8. HeatDGrhs1D.m: Evaluation of the heat operator using a discontinuous Galerkin method.**

---

```

function [ rhsu ] = HeatDGrhs1D(x,u,h,m,N,Ma,S,VtoE,nu)
% function [rhsu] = HeatDGrhs1D(u,x,u,h,m,N,Ma,S,VtoE,nu)
% Purpose : Evaluate RHS flux in 1D heat equation using using a DG method
Imat = eye(m+1); ue = zeros(2,N+2); qe = zeros(2,N+2);

% Extend data and assign boundary conditions
[ue] = extendDG(u(VtoE), 'D', 0, 'D', 0);

% Compute numerical fluxes at interfaces
fluxr = HeatFlux(ue(2,2:N+1),ue(1,3:N+2), 'C');
fluxl = HeatFlux(ue(2,1:N),ue(1,2:N+1), 'C');

% Compute aux variable q
qh = S*u - (Imat(:,m+1)*fluxr(1,:)) - Imat(:,1)*fluxl(1,:));
q = nu.*((h/2*Ma)\qh);

% Extend data and assign boundary conditions
[qe] = extendDG(q(VtoE), 'N', 0, 'N', 0);

% Compute numerical fluxes at interfaces
fluxr = HeatFlux(qe(2,2:N+1),qe(1,3:N+2), 'C');
fluxl = HeatFlux(qe(2,1:N),qe(1,2:N+1), 'C');

% Compute aux variable q
rh = S*q - (Imat(:,m+1)*fluxr(1,:)) - Imat(:,1)*fluxl(1,:));
rhsu = (h/2*Ma)\rh;
return

```

---

The specification of the numerical flux is made in HeatFlux.m, enabling both the central flux and the alternating flux.

---

**Script 12.9. HeatFlux.m: Definition of the numerical flux for the heat equation, solved with a discontinuous Galerkin method.**

---

```

function [ flux ] = HeatFlux(u,v,type);
% function [flux] = HeatFlux(u,v,type);
% Purpose: Compute flux for heat equation.
% Type: 'L' =Upwind, 'C'=Central, 'R'=Downwind

if type=='L' flux = u; end
if type=='C' flux = (u+v)/2; end
if type=='R' flux = v; end
return

```

---

## Nonlinear viscosity term

With a stable and robust approach to implement the second order operator in place, we now seek to control the amount of dissipation through the nonlinear diffusivity  $\nu(u)$ .

The basic philosophy shall be that  $\nu(u)$  is small when the solution is smooth and large when the solution is less regular. This calls for a smoothness indicator. A local representation of the solution is

$$u_b(x) = \sum_{i=0}^m \tilde{u}_i P_i(x), \quad \tilde{u}_i = \int_{-1}^1 u_b(x) P_i(x) dx, \quad x \in [-1, 1].$$

By recalling that the Legendre polynomials satisfy (12.15), one easily proves that

$$|\tilde{u}_i| \simeq \frac{C}{i^p}, \quad (12.77)$$

provided  $u^{(p-1)} \in L^2(\Omega)$ . Hence, the decay of the expansion coefficients conveys information about the regularity of the solution. This observation, first utilized in a different context in [82], was introduced as a smoothness indicator in this context in [86].

If one defines the elementwise smoothness indicator as  $S_j = |\tilde{u}_{j,m}|^2 / \|u_j\|_2^2$  and seeks to identify elements with a nonsmooth solution, then  $S_j \simeq m^{-4}$  indicates that  $u_j$  is at least continuous. Clearly if  $S_j \ll m^{-4}$ , one can reasonably assume that the solution is smooth and  $\nu(u)$  can be taken to be zero in such an element. Based on this, [86] proposes to use

$$\nu = \nu_0 \begin{cases} 0, & s_j < s_0 - \chi, \\ \frac{1}{2} \left( 1 + \sin \frac{\pi(s_j - s_0)}{2\chi} \right), & s_0 - \chi < s_j < s_0 + \chi, \\ 1, & s_j > s_0 + \chi, \end{cases}$$

where  $\nu_0 \propto h/m$ ,  $s_j = \log S_j$ ,  $s_0 = \log(m^{-4})$ , and  $\chi$  is a free parameter.

Since this cellwise definition of the viscosity leads to a piecewise constant function, a continuous function is constructed by matching the cell values across the interfaces in a piecewise-linear manner. The implementation of the nonlinear viscosity is illustrated in `Nonvisc1.m`.

---

**Script 12.10. *Nonvisc1.m: Computation of the nonlinear viscosity following Persson and Peraire [86]***

---

```
function [nu] = Nonvisc1(x,u,iV,m,N,h,nu0,kappa);
% function [nu] = Nonvisc1(x,u,iV,m,N,h,nu0,kappa);
% Purpose: Compute nonlinear viscosity following Persson and Peraire (2006)
nu = zeros(m+1,N); S = zeros(1,N); nuh = zeros(1,N); onev = ones(m+1,1);

% Extract coefficients and compute smoothness measure
uh = iV*u; S = uh(m+1,:).^2./sum(uh.*uh); se = log(S);

% Compute elementwise viscosity
s0 = log(1/m^4);
nu1 = ((s0-kappa)<=se).*(se<=(s0+kappa)); nu2 = (se>(s0+kappa));
nuh = nu0*h/m*(nu1/2.*(1+sin(pi*(se-s0)/(2*kappa))) + nu2);

% Compute continuous viscosity
nue = zeros(1,N+2); nue = [nuh(1) nuh nuh(N)];
maxL = max(nue(1:N),nue(2:N+1)); maxR = max(nue(2:N+1),nue(3:N+2));
nu = onev*maxL + (x-onev*x(1,:))/h.*((onev*(maxR-maxL)));
return
```

---

While this approach is intuitive and easy to implement, the estimate of  $S_j$  based on the last coefficients is, in many cases, too simple. It tends to underestimate the smoothness which requires  $\nu_0$  to be too large to control oscillations. This results in over-dissipative results.

To address these shortcomings, a more involved approach for estimating the smoothness is proposed in [70]. The initial observation is again (12.77), now expressed as

$$\log |\tilde{u}_i| \simeq \log C - p \log(i).$$

The first step is to consider a least-squares problem,

$$\min_{p,C} \sum_{i=1}^m (\log |\tilde{u}_i| - (\log C - p \log(i)))^2,$$

to recover  $C$  and  $p$ . The zeroth mode has been removed to maintain conservation.

This implicitly relies on the assumption that  $|\tilde{u}_i|$  decays monotonously. Unfortunately, this is not the case for many functions and, as a result, the estimate of  $p$  is often too high. To address this, [70] proposes to modify  $\tilde{u}_i$  as

$$|\tilde{\tilde{u}}_i| := \max_{l=\min(i,m-1), \dots, m} |\tilde{u}_l|.$$

In other words,  $\tilde{\tilde{u}}_i$  is set to the maximum value of all higher modes, hence enforcing a monotone behavior. The last mode is forced to be larger than the second-largest mode to avoid even-odd effects. Using  $\tilde{\tilde{u}}_i$  to estimate  $C$  and  $p$  is found to be a robust strategy for a variety of problems.

Since we ignored the cell average in the above, very small variations around the mean may emerge as highly oscillatory. This causes the estimated of  $p$  to be close to zero. To reintroduce this sense of scale, define coefficients with the desired decay

$$|\alpha_i| = \frac{i^{-m}}{\sqrt{\sum_{i=1}^m i^{-2m}}},$$

and modify the coefficients as

$$|\bar{u}_i| := \sqrt{|\tilde{\tilde{u}}_i|^2 + \|u_j\|_h^2 |\alpha_i|^2},$$

to make sure that small scale noise is scaled out. Once  $|\bar{u}_i|$  is obtained, they are subsequently used to recover  $C$  and  $p$ . Once  $p$  is determined, the viscosity is defined as

$$\nu(p) = \nu_0 \begin{cases} 1, & p < 1, \\ 1 - (p-1)/2, & 1 \leq p \leq 3, \\ 0, & p > 3. \end{cases}$$

Finally, we set  $\nu_0 = \lambda_{\max} h/m$ , where  $\lambda_{\max}$  is chosen to be proportional to the maximum local characteristic velocity. The implementation of the nonlinear viscosity is illustrated in `Nonvisc2.m`.

**Script 12.11. `Nonvisc2.m`: Computation of the nonlinear viscosity following [70].**

---

```
function [nu] = Nonvisc2(x,u,iV,m,N,h,nu0);
% function [nu] = Nonvisc2(x,u,iV,m,N,h,nu0);
% Purpose: Compute nonlinear viscosity following Kloeckner et al. (2013)
nu = zeros(m+1,N); S = zeros(1,N); nuh = zeros(1,N); onev = ones(m+1,1);
eps0 = 1e-10;
```

```

% Special case of m=1,2
if (m<3)
    nuh = nu0*h/m*ones(1,N);
else
    % Extract coefficients and compute smoothness measure
    uh = iV*u; uh1 = uh(2:m+1,:);

    % Adjust for scaling
    bh = [1:m].^(-m) ./ sqrt(sum([1:m].^(-2*m)));
    ut = sqrt(uh1.*uh1 + (bh.*bh)*sum(uh1.*uh1));

    % Adjust for non-monotone decay
    ub = ut;
    for i=1:m-1
        ub(i,:) = max(abs(ut(i:m,:)),[],1);
    end

    % Compute decay estimate by least squares fit
    b1 = log([1:m]); h1 = -sum(b1); h2 = -b1*log(ub+eps0);
    A = [m h1; h1 h1^2]; b = [-h1*ones(1,N);h2]; coef = A\b;

    % Compute elementwise viscosity
    nu1 = (coef(2,:)<=1); nu2 = (coef(2,:)>1).* (coef(2,:)<3);
    nuh = nu0*h/m*(nu1 + nu2.* (1-(coef(2,:)-1)/2));
end

% Compute continuous viscosity
nue = zeros(1,N+2); nue = [nuh(1) nuh nuh(N)];
maxL = max(nue(1:N),nue(2:N+1)); maxR = max(nue(2:N+1),nue(3:N+2));
nu = onev*maxL + (x-onev*x(1,:))/h.* (onev*(maxR-maxL));
return

```

Finally, we discuss a different approach to estimate smoothness. It was introduced in [45] and is based on insight into the behavior of the entropy. As we have discussed at length in Chapter 2 and in subsection 8.2.2, we can typically define an entropy pair  $(\eta(u), \psi(u))$  that satisfies

$$\frac{\partial \eta}{\partial t} + \frac{\partial \psi}{\partial x} \leq 0,$$

where  $u$  is a weak solution and equality holds when the solution to the conservation law is smooth. This suggests that one constructs a smoothness estimator by considering the elementwise residual

$$R_j(x) = \frac{\partial \eta(u_j)}{\partial t} + \frac{\partial \psi(u_j)}{\partial x} = \frac{\partial \eta(u_j)}{\partial t} + f'(u_j) \frac{\partial \eta(u_j)}{\partial x},$$

where we use that  $\psi' = f'\eta'$ . Clearly, if  $u_j$  is smooth, this residual should be small.

In [45] it is proposed to use this to define a local viscosity as

$$\nu_j = \min(c_1 b \max|f'(u_j)|, c_2 b^2 D_j(u_j)), \quad D_j = \max(|R_j(u_j)|, |J_j|/N_j),$$

where

$$J_j = \frac{1}{b} \{ \{ f(u_j) \} \} \llbracket \eta_j \rrbracket$$

measures the entropy contribution from jumps across the interfaces, and

$$N_j = \max |\eta(u_j) - \bar{\eta}(u_j)|$$

is a normalization factor. Here  $\bar{\eta}$  is the cell average of the entropy.

In [45], the two constants are taken as  $c_1 = 1/(2k)$  and  $c_2 = 1$ , although their values can be modified. This work reports substantial tests to validate that the method does not destroy accuracy when approximating smooth problems. Since the discontinuous Galerkin scheme is nonlinearly stable and the nonlinear viscosity is semipositive, stability of the scheme is expected. This is confirmed in a rigorous stability analysis in [8].

Implementation of the entropy-based nonlinear viscosity is illustrated in `Entvisc.m`.

**Script 12.12. `Entvisc.m`: Computation of the nonlinear viscosity based on entropy as a smoothness indicator [45].**

---

```
function [nu] = Entvisc(x, entold, entnew, fpold, fpnew, D, iV, N, m, h, k, VtoE, c1, c2);
% function [nu]=Entvisc(x, entold, entnew, fpold, fpnew, D, iV, N, m, h, k, VtoE, c1, c2);
% Purpose: Compute nonlinear viscosity following entropy approach
nu = zeros(m+1,N); nuh = zeros(1,N); onev = ones(m+1,1);
ente = zeros(N+2,2); fpe = zeros(N+2,2);

% Compute cell wise residual by Crank–Nicholson approximation
Resi = (entnew - entold)/k + (fpnew .* (D*entnew) + fpold .* (D*entold))/h;

% Compute interface jump
[ente] = extendDG(entnew(VtoE), 'N', 0, 'N', 0);
[fpe] = extendDG(fpnew(VtoE), 'N', 0, 'N', 0);
cL = (fpe(1:N,2)+fpe(2:N+1,1)).*(ente(1:N,2)-ente(2:N+1,1))/2;
cR = (fpe(2:N+1,2)+fpe(3:N+2,1)).*(ente(2:N+1,2)-ente(3:N+2,1))/2;
Ji = max(cL', cR')/h;

% Compute normalization
Eh = iV*entnew; NormE = max(abs(entnew - onev*Eh(1,:)));

% Define numerical viscosity
Di = max(max(abs(Resi)), abs(Ji)/NormE);
nuh = min(c1*h*max(abs(fpnew)), c2*h^2*Di);

% Compute continuous viscosity
nue = zeros(1,N+2); nue = [nuh(1) nuh nuh(N)];
maxL = max(nue(1:N), nue(2:N+1)); maxR = max(nue(2:N+1), nue(3:N+2));
nu = onev*maxL + (x-onev*x(1,:))/h.* (onev*(maxR-maxL));
return
```

---

Before proceeding to evaluate the performance of these different methods in more detail, one central issue needs to be addressed. Since the conservation law is transformed into a (locally) dissipative problem, the second order operator impacts the time step. For the conservation law, we expect the time step to scale as

$$k \leq C_1 \frac{h}{\max |f'|},$$

where  $C_1$  is a constant that depends on the order of the scheme. For discontinuous Galerkin methods, the scaling is typically  $C_1 \propto m^{-2}$ , i.e., the time step must decrease as the order increases. We refer to [56] for a fuller discussion of this.

A similar situation emerges for the second order operator. Here we expect a time step scaling as

$$k \leq C_2 \frac{h^2}{\max |v_j|},$$

where  $v$  is the viscosity coefficient and  $C_2 \propto m^{-4}$ . Unless  $v$  is very small, this constraint will dominate the selection of the time step and, possibly, require a very small time step to retain stability of the scheme. This observation is, perhaps, the most important reason to apply nonlinear viscosity with care, as the overall cost may be substantial.

To gain some insight into the differences of these three methods and evaluate their efficiency, let us consider an example.

**Example 12.28.** We consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

subject to periodic boundary conditions. As initial condition we use

$$u(x, 0) = \sin(2\pi x).$$

The solution steepens and, by symmetry, a shock forms at  $x = 0.5$ . Subsequently the shock dissipates. The problem does not have an exact solution in closed form [5], so we use a highly resolved ENO solution as the reference solution, illustrated in Fig. 12.11.

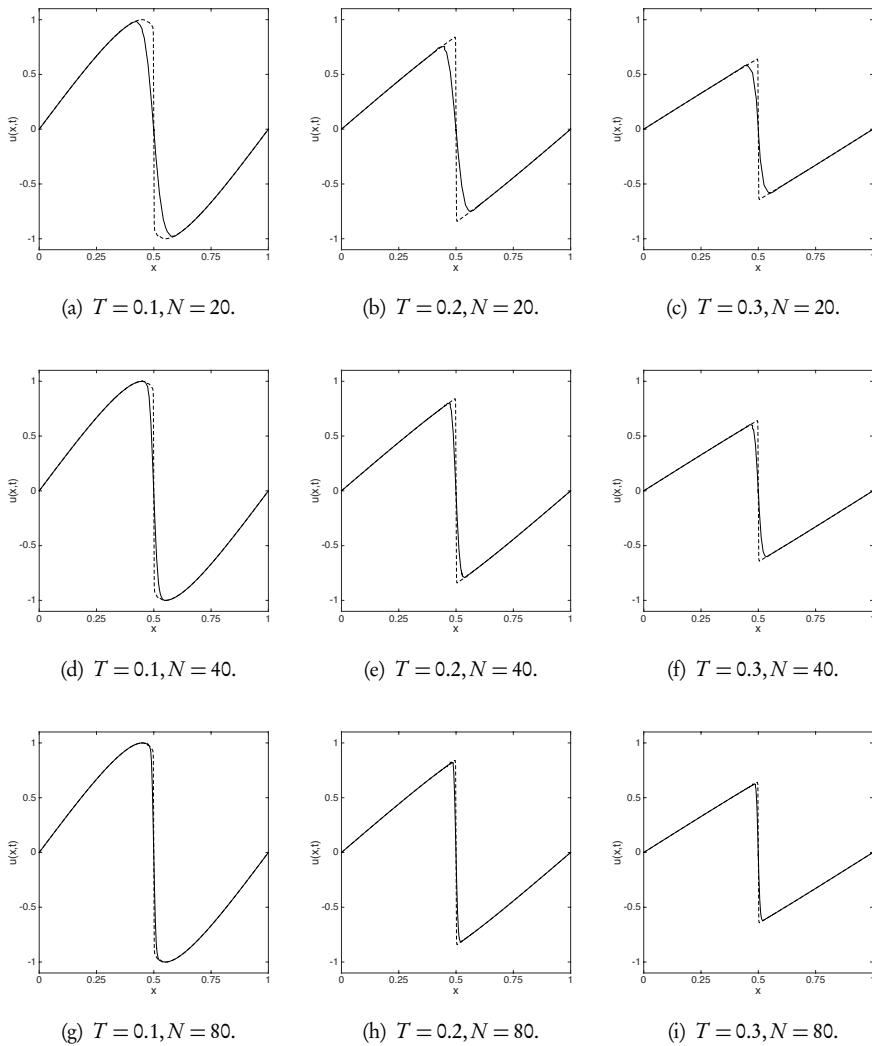
We consider numerical experiments with order  $m = 4$  and the number of cells  $N = 20, 40, 80$ , respectively. To illustrate the temporal evolution, we consider the solution at  $T = 0.1, 0.2, 0.3$ .

Figure 12.11 shows the computed solutions, using the nonlinear viscosity proposed in [86] with parameters  $v_0 = 25$  and  $\chi = -6$ . We emphasize that no systematic effort has been made to optimize these parameters but some variation suggests that these are reasonable choices for this problem. The results confirm the expected localized smoothing and show convergence to a sharply resolved shock with increasing resolution.

In Fig. 12.12, we show the same set of results, obtained with the nonlinear viscosity proposed in [70] with parameter  $v_0 = 2$ . The differences with the results in Fig. 12.11 are marginal, possibly indicating a slightly reduced smearing of the shock.

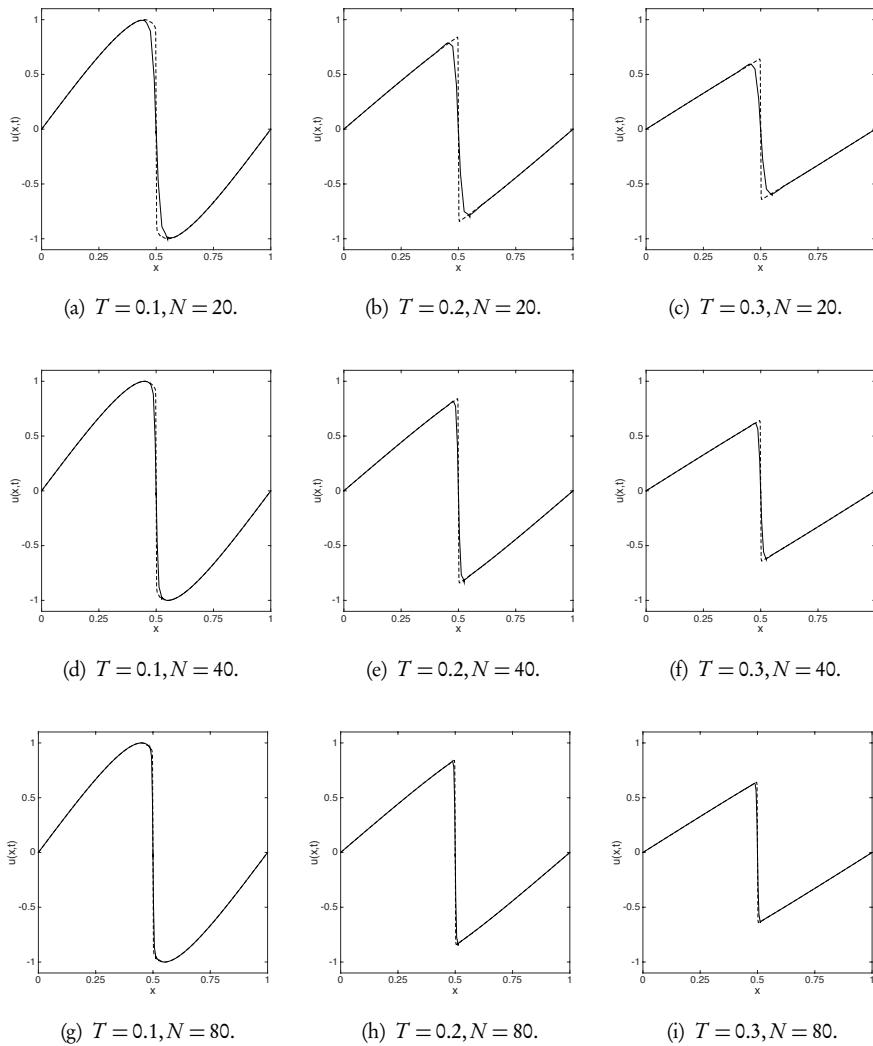
Finally, Fig. 12.13 shows the results obtained with the nonlinear viscosity defined by the entropy production. This method is essentially free of constants. Again, we observe that the differences with the previous results are marginal, although these latter results show the strongest signs of dissipation.

To illustrate the differences between the three approaches more quantitatively, we show in Fig. 12.14 the value of the nonlinear viscosity for the three schemes in the case of  $m = 4$  and  $N = 40$  during the time interval of  $[0, 0.2]$ . All three approaches behave qualitatively in the same way, showing an increase in the viscosity as the solution steepens and the shock forms. However, by taking a closer look, differences reveal themselves. In particular, the first approach results in a rapidly switching diffusivity. This is a result of the estimate of the smoothness based only on the last expansion coefficient. In contrast to this, the two latter methods feature a more smoothly varying diffusivity, as would seem preferred. More important, however, is that the magnitude



**Figure 12.11.** Nonlinear viscosity applied to the solution of Burgers' equation using a discontinuous Galerkin method with  $m = 4$ . We illustrate the solution at time  $T$  for different numbers of elements  $N$ . The dashed line reflects the solution obtained with a highly resolved ENO method. The nonlinear viscosity is defined following [86].

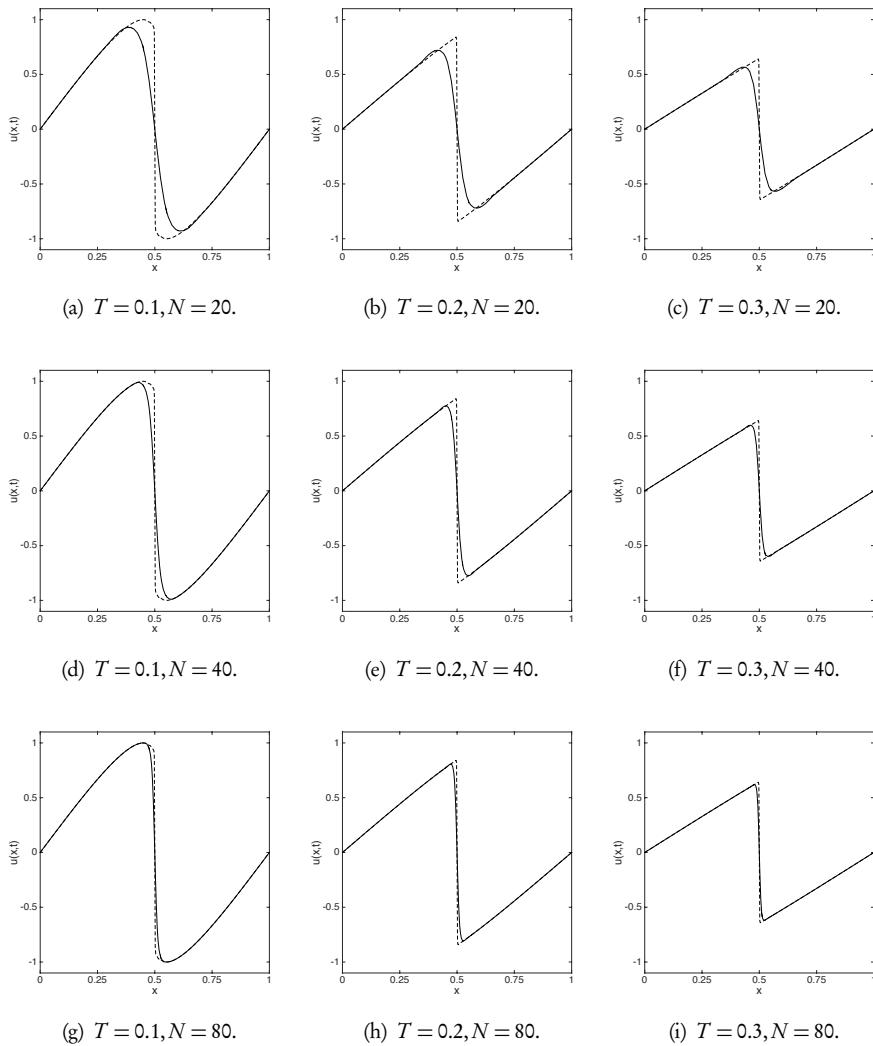
of the numerical viscosity is significantly higher for the first scheme than the latter two. Although a visual comparison of the solutions suggests that the impact of this is limited, this difference forces the use of a time step, which is almost an order of magnitude smaller than what is possible with the last two methods. However, one should keep in mind that, as for linear filters, these methods have parameters that can be tuned and the outcome may well vary for different problems. ■



**Figure 12.12.** Nonlinear viscosity applied to the solution of Burgers' equation using a discontinuous Galerkin method with  $m = 4$ . We illustrate the solution at time  $T$  for different numbers of elements  $N$ . The dashed line reflects the solution obtained with a highly resolved ENO method. The nonlinear viscosity is defined following [70].

## 12.2.4 • Slope limiters

We have, thus far, considered two techniques to address the impact of the Gibbs oscillations, both of which have limitations. Filtering is easy to apply and localizes the impact of the oscillations, but it does not eliminate the oscillations. On the other hand, nonlinear dissipation enables a subcell resolution, albeit potentially at considerable cost.

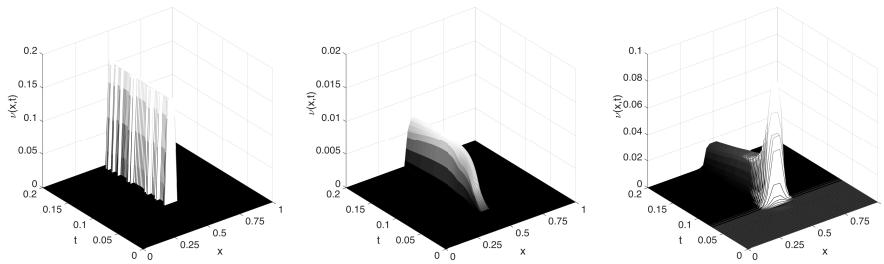


**Figure 12.13.** Nonlinear viscosity applied to the solution of Burgers' equation using a discontinuous Galerkin method with  $m = 4$ . We illustrate the solution at time  $T$  for different numbers of elements  $N$ . The dashed line reflects the solution obtained with a highly resolved ENO method. The nonlinear viscosity is defined by using an entropy measure of smoothness [45].

To seek an alternative, consider the simplest case with the cellwise solution on the form

$$u_j(x) = \bar{u}_j^n + (x - x_j)(u_j)_x = \bar{u}_j^n + \frac{x - x_j}{h} \delta u_j^n,$$

where  $\delta u_j^n/h = (u_j)_x$  is the local slope. Reaching back to section 10.2, this suggests that we seek to adapt slope limiting to the discontinuous Galerkin method.



**Figure 12.14.** Evaluation of the nonlinear viscosity for Burgers' equation as a shock develops. The solution is obtained with a discontinuous Galerkin methods with  $m = 4$  and  $N = 40$ . On the left, the diffusivity is obtained following [86], the results in the middle employ the scheme in [70], while the results on the right are obtained using an entropy-based approach [45]. Note that the vertical scales for the viscosity are different.

Let us consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to periodic boundary conditions and an appropriate initial condition. The cellwise, semidiscrete discontinuous Galerkin scheme in weak form is

$$\int_{D_j} \frac{\partial u_j}{\partial t} \psi_{j,i} - f_j \frac{d\psi_{j,i}}{dx} dx = -F_{j+1/2} \psi_{j,i}(x_{j+1/2}) + F_{j-1/2} \psi_{j,i}(x_{j-1/2}).$$

The numerical flux  $F(u_{j+1/2}^-, u_{j+1/2}^+)$  depends on  $u_{j+1/2}^\mp$  which, in the case of  $m = 1$ , takes the form

$$u_{j\pm 1/2}^\mp = \bar{u}_j^n \pm \frac{b}{2} (u_j)_x = \bar{u}_j^n \pm \frac{1}{2} \delta u_j^n.$$

Following the development in section 10.2, we seek ways to modify the slope  $\delta u_j$  to eliminate oscillations and, under some conditions, ensure total variation stability.

The semidiscrete scheme for the cell average is

$$\frac{d\bar{u}_j}{dt} = -\frac{1}{b} (F_{j+1/2} - F_{j-1/2}), \quad (12.78)$$

while the fully discrete form is

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{k}{b} (F_{j+1/2}^n - F_{j-1/2}^n), \quad F_{j+1/2}^n = F(u_{j+1/2}^-, u_{j+1/2}^+), \quad (12.79)$$

where

$$u_{j+1/2}^- = \bar{u}_j^n + \tilde{u}_j^-, \quad u_{j+1/2}^+ = \bar{u}_{j+1}^n - \tilde{u}_{j+1}^+.$$

Total variation stability is guaranteed by the following result [28].

**Theorem 12.29.** *Assume that*

$$-\theta \leq \frac{\Delta^+ \tilde{u}_j^+}{\Delta^+ \tilde{u}_j^n} \leq 1, \quad -\theta \leq -\frac{\Delta^- \tilde{u}_j^-}{\Delta^- \tilde{u}_j^n} \leq 1,$$

and the numerical flux is monotone. The semidiscrete scheme (12.78) is TVD for  $\theta \geq 0$ , and the fully discrete scheme (12.79) is TVD provided

$$\frac{k}{h} \max_u |f'(u)| \leq \frac{C}{1+\theta}.$$

Here  $C$  is a positive constant.

**Proof.** Let us express the numerical flux as

$$F(u_{j+1/2}^-, u_{j+1/2}^+) = F(\bar{u}_j^n + \tilde{u}_j^-, \bar{u}_{j+1}^n - \tilde{u}_{j+1}^+),$$

and assume that  $L_i$  is the Lipschitz constant of  $F(\cdot, \cdot)$  with respect to the  $i$ th argument. We obtain

$$-(F_{j+1/2}^n - F_{j-1/2}^n) = -C_{j-1/2} \Delta^- \bar{u}_j^n + D_{j+1/2} \Delta^+ \bar{u}_j^n,$$

where

$$C_{j-1/2} = L_1 \left( 1 + \frac{\Delta^- \tilde{u}_j^-}{\Delta^- \tilde{u}_j^n} \right), \quad D_{j+1/2} = -L_2 \left( 1 - \frac{\Delta^+ \tilde{u}_j^+}{\Delta^+ \tilde{u}_j^n} \right).$$

Relying on Theorem 8.4, we can establish TVD-stability by considering the signs of these two coefficients. Indeed, since  $F$  is monotone, we know that the Lipschitz constant

$$L_1 = \frac{F(\bar{u}_j^n + \tilde{u}_j^-, \cdot) - F(\bar{u}_{j-1}^n + \tilde{u}_{j-1}^-, \cdot)}{(\bar{u}_j^n + \tilde{u}_j^-) - (\bar{u}_{j-1}^n + \tilde{u}_{j-1}^-)}$$

is positive. Combined with the assumption

$$-\theta \leq -\frac{\Delta^- \tilde{u}_j^-}{\Delta^- \tilde{u}_j^n} \leq 1,$$

this guarantees that  $C_{j-1/2} \geq 0$ . The proof that  $D_{j+1/2} \geq 0$  follows in a similar manner, hence establishing TVD for the semidiscrete scheme.

To prove TVD-stability for the fully discrete scheme, we must additionally ensure that

$$\frac{k}{h} (C_{j+1/2} + D_{j+1/2}) \leq 1,$$

which requires

$$\frac{k}{h} (L_1 - L_2) \leq \frac{1}{1+\theta}.$$

Since  $(L_1 - L_2) \leq C \max_u |f'(u)|$ , the result follows.  $\square$

Theorem 10.5 can be applied to guarantee the limits on  $\tilde{u}_j^-$  and  $\tilde{u}_j^+$  in the above result. We recover

$$\overline{\delta u_j^n} = \text{minmod}(\delta u_j^n, \theta \Delta^+ \bar{u}_j^n, \theta \Delta^- \bar{u}_j^n),$$

where the choice of  $\theta = 1$  yields the minmod limiter, while the choice of  $\theta = 2$  yields the MUSCL limiter. Naturally, other TVD-stable limiters, discussed in section 10.2, can be applied. The reduction to  $\mathcal{O}(h)$  at local extrema can be overcome by resorting to the TVB limiter, as discussed in section 10.2.

TVD/TVB-stability suffices to prove convergence of the discontinuous Galerkin method to a weak solution [28]. The generalization of this result to enforce an entropy condition is discussed in [28] and relies on a problem-dependent modification of the slope limiter. Nevertheless, extensive computational results suggest that this is, in fact, not needed.

The development of TVD/TVB-stable discontinuous Galerkin methods is initiated in [27, 28] and continued in a series of papers which discuss their extension to systems by the use of characteristic variables [26], to multidimensional scalar problems [25] and, finally, to multidimensional systems [30].

The main limitation of the presented approach is the assumption that  $m = 1$ , i.e., it is restricted to second order accurate schemes, even in the case of smooth solutions. Since the discontinuous Galerkin method allows for schemes of arbitrary accuracy in smooth regions of the solutions, it is worth seeking a generalization.

A simple strategy, advocated in [27, 26], is to locally reduce the order of the approximation of the numerical scheme to first order and use the minmod function to identify the troubled cells that require limiting. Let us consider

$$\tilde{u}_j^+ = \bar{u}_j^n - u_{j-1/2}^+, \quad \tilde{u}_j^- = u_{j+1/2}^- - \bar{u}_j^n,$$

and evaluate the minmod-based indicator

$$\begin{aligned} \overline{\tilde{u}_j^+} &= \bar{u}_j^n - \text{minmod}(\bar{u}_j^n - u_{j-1/2}^+, \Delta^+ \bar{u}_j^n, \Delta^- \bar{u}_j^n), \\ \overline{\tilde{u}_j^-} &= \bar{u}_j^n + \text{minmod}(u_{j+1/2}^- - \bar{u}_j^n, \Delta^+ \bar{u}_j^n, \Delta^- \bar{u}_j^n). \end{aligned} \quad (12.80)$$

If  $\overline{\tilde{u}_j^\pm}$  is different from  $\tilde{u}_j^\pm$ , the local solution is reduced to a linear solution and slope-limiting applied. On the other hand, if no limiting is needed, the full polynomial solution is used. This approach retains TVD/TVB-stability in nonsmooth regions, while achieving full order accuracy in smooth parts of the solution.

The implementation of the TVD slope limiting for a general piecewise polynomial function is illustrated in `SlopeLimitCSDG.m`.

**Script 12.13. *SlopeLimitCSDG.m*: Application of TVD slope-limiting to a piecewise polynomial solution.**

---

```
function [ ulimit ] = SlopeLimitCSDG(x,u,m,h,N,V,iV);
% function ulimit = SlopeLimitCSDG(x,u,m,h,N,V,iV);
% Purpose: Apply slope limiter by Cockburn-Shu (1989)
% to u - an m'th order polynomial
eps0=1.0e-8;
```

```

% Strength of slope limiter - Minmod: theta=1, MUSCL: theta=2
theta=2.0;

% Compute cell averages and cell centers
uh = iV*u; uh(2:(m+1),:)=0; uavg = V*uh; ucell = uavg(1,:); ulimit = u;

% Extend cell averages
[ve] = extendDG(ucell, 'N', 0, 'N', 0);

% extract end values and cell averages for each element
uel = u(1,:); uer = u(end,:);
vj = ucell; vjm = ve(1:N); vjp = ve(3:N+2);

% Find elements that require limiting
vel = vj - minmod([(vj-uel);(vj-vjm);(vjp-vj)]);
ver = vj + minmod([(uer-vj);(vj-vjm);(vjp-vj)]);
ids = find(abs(vel-uel)>eps0 | abs(ver-uer)>eps0);

% Apply limiting when needed
if(~isempty(ids))
    % create piecewise linear solution for limiting on specified elements
    uhl = iV*u(:,ids); uhl(3:(m+1),:)=0; ulin = V*uhl;
    ux = 2/h*(vj(ids)-ulin(1,:));

    % Limit function
    x0h = ones(m+1,1)*(x(end,:)+x(1,:))/2;
    ulimit(:,ids) = ones(m+1,1)*vj(ids)+(x(:,ids)-x0h(:,ids)).*(ones(m+1,1)*...
        minmod([ux(1,:);theta*(vjp(ids)-vj(ids))./h;...
        theta*(vj(ids)-vjm(ids))./h]));
end
return

```

A slightly more aggressive limiter is proposed in [11]. As a first step, the slope is limited as

$$\overline{\delta u_j^n} = \text{minmod}(\delta u_j^n, \theta \Delta^+ \bar{u}_j^n, \theta \Delta^- \bar{u}_j^n).$$

Since this is similar to the TVD approach discussed above, it reduces the accuracy to first order at local extrema. To address this, it is proposed that we consider

$$\overline{\overline{\delta u_j^n}} = \text{minmod}(\delta u_j^n, \delta u_{j-1}^n, \delta u_{j+1}^n)$$

and modify the definition of the slopes as

$$\overline{\delta u_j^n} = \text{maxmod}(\overline{\delta u_j^n}, \overline{\overline{\delta u_j^n}}).$$

The maxmod function is intended to eliminate the local loss of resolution without introducing artificial oscillations. While there is no known theoretical justification, qualitative arguments and extensive computational results presented in [11] support the claim of stability.

The implementation of this more aggressive slope-limiting for a general piecewise polynomial function is illustrated in `SlopeLimitBSBDG.m`.

**Script 12.14.** *SlopeLimitBSBDG.m: Application of TVD slope-limiting to piecewise polynomial solution, following [11].*

---

```

function [ ulimit ] = SlopeLimitBSBDG (x,u,m,h,N,V,iV);
% function ulimit = SlopeLimitBSBDG (x,u,m,h,N,V,iV);
% Purpose: Apply slope limiter by Burbeau-Sagaut-Brunneau (2001)
% to u - an m'th order polynomial
eps0=1.0e-8;

% Strength of slope limiter - Minmod: theta=1, MUSCL: theta=2
theta=2.0;

% Compute cell averages and cell centers
uh = iV*u; uhx = uh; uh(2:(m+1),:)=0; uavg = V*uh; ucell = uavg(1,:);
uhx(3:(m+1),:)=0; ulin = V*uhx; ux = 2/h*(ucell - ulin(1,:)); ulimit = u;

% Extend cell averages
[ve] = extendDG(ucell,'P',0,'P',0); [vxe] = extendDG(ux,'P',0,'P',0);

% extract end values and cell averages for each element
uel = u(1,:); uer = u(end,:); vj = ucell; vjm = ve(1:N); vjp = ve(3:N+2);
vxj = ux; vxjm = vxe(1:N); vxjp = vxe(3:N+2);

% Find elements that require limiting
vel = vj - minmod([(vj-uel);(vj-vjm);(vjp-vj)]);
ver = vj + minmod([(uer-vj);(vj-vjm);(vjp-vj)]);
ids = find(abs(vel-uel)>eps0 | abs(ver-uer)>eps0);

% Apply limiting when needed
if(~isempty(ids))
    % create piecewise linear solution for limiting on specified elements
    uhl = iV*u(:,ids); uhl(3:(m+1),:)=0; ulin = V*uhl;
    x0h=ones(m+1,1)*(x(end,:)+x(1,:))/2;

    % Limit function
    ux1 = minmod([ vxj(ids); theta*(vjp(ids)-vj(ids))/h; ...
                  theta*(vj(ids)-vjm(ids))/h ]);
    ux2 = minmod([ vxj(ids); vxjm(ids); vxjp(ids) ]);
    ulimit(:,ids) = ones(m+1,1)*vj(ids)+(x(:,ids)-x0h(:,ids)).*...
        (ones(m+1,1)*maxmod([ ux1; ux2]));
end
return

```

---

An alternative approach to high-order limiting was first proposed in [7], and subsequently further developed and refined in [11]. Let us begin by expressing the local solution as

$$u_j(x) = \sum_{i=0}^m \tilde{u}_{j,i} P_i(x),$$

where  $P_i$  is the Legendre polynomial and, for the sake of simplicity only, we restrict ourselves to  $x \in [-1, 1]$ . It is clear that  $\tilde{u}_j = \tilde{u}_{j,0}$ . To ensure TVD/TVB-stability, the limiting of the slope

$$(u_j)_x(x) = \sum_{i=1}^m \tilde{u}_{j,i} P_i'(x)$$

implies that  $\tilde{u}_{j,i} = 0$  for  $i > 0$ . A slightly different approach involves the individual moments,  $\tilde{u}_{j,i}$ , of the solution. To ensure monotonicity of the moments, these are limited as

$$\begin{aligned} & \sqrt{(2i+1)(2i+3)}\tilde{u}_{j,i+1} \\ &= \minmod\left(\sqrt{(2i+1)(2i+3)}\tilde{u}_{j,i+1}, \theta(\tilde{u}_{j+1,i} - \tilde{u}_{j,i}), \theta(\tilde{u}_{j,i} - \tilde{u}_{j-1,i})\right), \end{aligned}$$

where  $\theta \geq 0$  is a free parameter. This approach is applied in a backward fashion to the moments of decreasing order, i.e.,  $i = m-1, \dots, 0$ . Furthermore, it is adaptive in the sense that once a moment is not limited, no lower moments are altered.

The implementation of hierarchical moment limiting for a general piecewise polynomial function is illustrated in `MomentLimitDG.m`.

---

**Script 12.15. *MomentLimitDG.m*: Routine for applying moment limiting to piecewise polynomial solution**

---

```
function [ ulimit ] = MomentLimitDG(x,u,m,h,N,V,iV);
% function ulimit = MomentLimitDG(x,u,m,h,N,V,iV);
% Purpose: Apply moment limiter to u - an m'th order polynomial
eps0=1.0e-8; eps1 = 1.0e-8;

% Strength of slope limiter - Minmod: theta=1, MUSCL: theta=2
theta=2.0;

% Compute cell averages and cell centers
uh = iV*u; uh(2:(m+1),:)=0; uavg = V*uh; ucell = uavg(1,:); ulimit = u;

% Extend cell averages
[ve] = extendDG(ucell , 'N' ,0 , 'N' ,0);

% extract end values and cell averages for each element
uel = u(1,:); uer = u(end,:); vj = ucell; vjm = ve(1:N); vjp = ve(3:N+2);

% Find elements that require limiting
vel = vj - minmod([(vj-uel);(vj-vjm);(vjp-vj)]);
ver = vj + minmod([(uer-vj);(vj-vjm);(vjp-vj)]);
ids = (abs(vel-uel)<eps1 & abs(ver-uer)<eps1);
mark = zeros(1,N); mark = (ids | mark);

% Compute expansion coefficients
uh = iV*u;

% Apply limiting when needed
for i=m+1:-1:2
    uh1 = uh(i,:); uh2 = uh(i-1,:);
    [uh2e] = extendDG(uh2 , 'P' ,0 , 'P' ,0); uh2m = uh2e(1:N); uh2p = uh2e(3:N+2);
    con = sqrt((2*i+1)*(2*i-1));
    uh1 = 1/con*minmod([con*uh1; theta*(uh2p - uh2); ...
        theta*(uh2 - uh2m)].*(1-mark) + mark.*uh1);
    idsh = abs(uh1-uh(i,:))<eps0; mark = (idsh | mark);
    uh(i,:)=uh1;
end
ulimit = V*uh;
return
```

---

The scheme combines simplicity with the use of all available high-order information. However, no stability theory is known.

This same approach can be applied to the more aggressive limiter, and extended to higher order by introducing the limiting function

$$\begin{aligned} & \sqrt{(2i+1)(2i+3)}\tilde{u}_{j,i+1} \\ &= \text{minmod}\left(\sqrt{(2i+1)(2i+3)}\tilde{u}_{j,i+1}, \theta(w_{j+1/2}^+ - \tilde{u}_{j,i}), \theta(\tilde{u}_{j,i} - w_{j-1/2}^-)\right), \end{aligned}$$

where

$$w_{j\pm 1/2}^\pm = \tilde{u}_{j\pm 1,i} \mp \sqrt{(2i+1)(2i+3)}\tilde{u}_{j\pm 1,i+1}.$$

In the case of systems, these techniques should generally be applied to the characteristic variables in order to avoid oscillations, as discussed in subsection 11.3.4. However, for some problems it may suffice to limit the conserved variables [7, 11]. The extension to the multidimensional case is typically achieved in a dimension-by-dimension fashion. Slope-limiters for general unstructured grids remains an active research field and we refer to some of the recent work [25, 30, 57, 74, 106, 121, 75, 79] for further details.

For a smooth problem, we compare the different options through the following example.

**Example 12.30.** We solve the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions and with  $u(x, 0) = \sin(\pi x)$ .

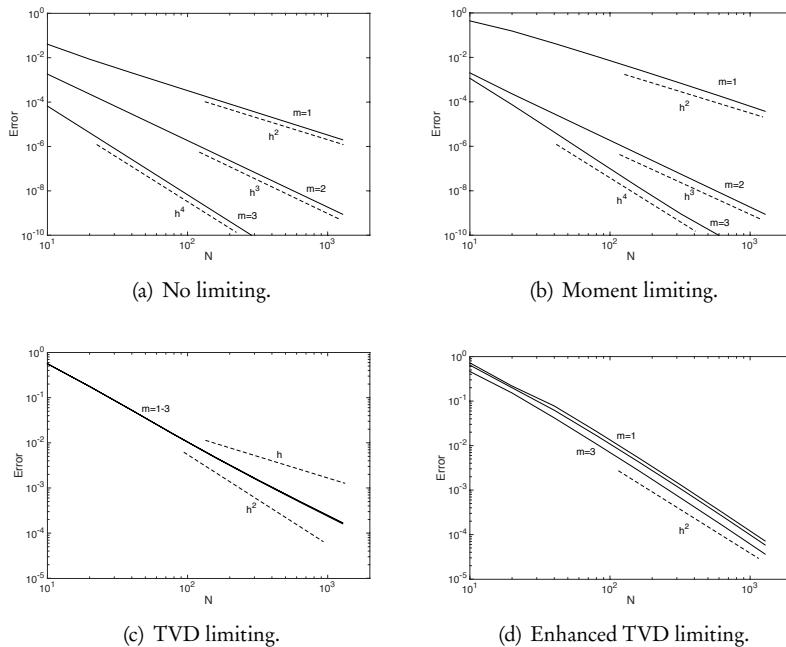
We employ a discontinuous Galerkin method of order  $m$  with an upwind flux and a third order SSPERK scheme. The problem is advanced until  $T = \pi$ .

The goal is to gain some insight into the impact of the limiter on the accuracy for a smooth solution. Figure 12.15 shows the  $L^2$ -error for the different limiters as the order of approximation increases.

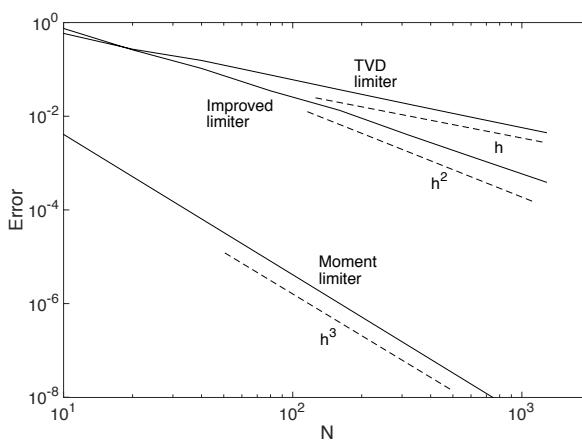
The unlimited scheme displays an order of convergence of  $\mathcal{O}(h^{m+1})$  as expected and the moment limiter [7] retains this property. On the other hand, the TVD limiter [27, 26] reduces the accuracy to  $\mathcal{O}(h^2)$  regardless of the order of approximation. As we have seen previously, this can be improved by the use of a TVB limiter. As for the improved limiter [11], the overall accuracy is improved as compared to the simple TVD limiter, although the order of approximation does not change. Similar results hold when the error is measured in  $L^1$ .

When considering the  $L^\infty$ -error, the differences between the three limiters are more pronounced, as illustrated in Fig. 12.16. As expected, the moment limiter retains full accuracy while we observe the expected  $\mathcal{O}(h)$  accuracy for the TVD limiter. The improved limiter [11] overcomes this problem and recovers  $\mathcal{O}(h^2)$ . The advantage of the improved limiter over the TVB limiter is that no problem-specific parameter is needed. ■

We return to a more detailed discussion on the performance of the limiters for discontinuous solutions in section 12.5.



**Figure 12.15.** Impact of limiting on a smooth solution. In the top row, we show the  $L^2$ -errors when using the unlimited scheme and the moment limit scheme [7] for different orders  $m$ . In the bottom row we show the  $L^2$ -errors obtained when a TVD limiter is applied [27, 26] and when an improved limiter is applied [11].



**Figure 12.16.** Behavior of the  $L^\infty$ -error at  $T = \pi$  for different limiters. We compare the error when using a TVD limiter [27, 26], an improved TVD limiter [11], and a moment-based limiter [7]. In all cases  $m = 2$ .

### 12.2.5 ■ WENO-based limiters

While slope limiting works well and, in some cases, offers a complete theory of TVD-stability, one can identify alternatives that may work equally well or, in many cases, better. If we recall the development of the WENO methods in section 11.3 and their success for problems with strong shocks, it is natural to pursue a similar approach to limiting.

Two issues need to be addressed when we attempt such an approach. First, we need to identify the cells that require reconstruction. This is achieved through the minmod-based troubled-cell indicator introduced in (12.80). This indicator is clearly not unique and a comprehensive comparison with alternatives can be found in [89]. As we shall discuss shortly, the design of this indicator function is essential to achieve optimal performance of the limiting.

Secondly, once the troubled cells are identified, we must seek the reconstruction of the solution therein through a WENO-inspired approach. Following [128], for each  $D_j$ , we define the three polynomials

$$p_{-1}(x) = u_{j-1}^n(x) - \bar{u}_{-1}^n + \bar{u}_j^n, \quad p_0(x) = u_j^n(x), \quad p_1(x) = u_{j+1}^n(x) - \bar{u}_1^n + \bar{u}_j^n,$$

where

$$\bar{u}_{\pm 1}^n = \frac{1}{b} \int_{x_{j-1/2}}^{x_{j+1/2}} u_{j\pm 1}^n(x) dx$$

are the cell averages of the polynomials  $u_{j\pm 1}^n$  extrapolated to cell  $j$ . It is easy to see that the three polynomials  $p_j$  have the same cell average. We define the reconstructed solution in cell  $D_j$  as

$$\overline{u_j^n(x)} = \omega_{-1} p_{-1}(x) + \omega_0 p_0(x) + \omega_1 p_1(x),$$

where the weights  $\omega_i$  are defined using the WENO strategy, i.e., we consider

$$\omega_i = \frac{\alpha_i}{\alpha_{-1} + \alpha_0 + \alpha_1}, \quad \alpha_i = \frac{\gamma_i}{(\varepsilon + \beta_i)^{2P}},$$

where the smoothness indicator  $\beta_i$  is defined as

$$\beta_i = \sum_{l=1}^m \int_{x_{j-1/2}}^{x_{j+1/2}} b^{2l-1} \left( \frac{d^l p_i(x)}{dx^l} \right)^2 dx.$$

The parameters  $\gamma_i$  are chosen differently from what we did for the WENO method, since the solution in the cell is a full polynomial which guarantees accuracy. While conservation requires that  $\sum_i \gamma_i = 1$ , we can otherwise choose  $\gamma_i$  with different goals in mind, e.g., upwinding ( $\gamma_{-1} \gg \gamma_1$  for a rightward propagating wave), or a strong bias towards the central element  $\gamma_0 \gg \gamma_{\pm 1}$ . A typical choice is

$$\gamma_{\pm 1} = 0.001, \quad \gamma_0 = 0.998.$$

This approach is displayed in `WENOlimitDG.m`.

---

**Script 12.16. *WENOlimitDG.m*: WENO-based limiter for the discontinuous Galerkin method, based on [128].**

---

```

function [ ulimit ] = WENOlimitDG(x,u,m,h,N,V,iV,Q,Xm,Xp);
% function ulimit = WENOlimitDG(x,u,m,h,N,V,iV,Q,Xm,Xp);
% Purpose: Apply WENO limiter by Zhong-Shu (2013)
% to u - an m'th order polynomial
eps0=1.0e-6;

% Set constants for limiting
eps1 = 1e-10; p=1;
gammam1 = 0.001; gamma0 = 0.998; gammap1 = 0.001;

% Compute cell averages and cell centers
uh = iV*u; uh(2:(m+1),:)=0; uavg = V*uh; ucell = uavg(1,:); ulimit = u;

% Compute extended polynomials with zero cell averages
[ue] = extendDG(u,'P',0,'P',0);
Pm = Xm'*ue; Pp = Xm'*ue;
Ph = iV*Pm; Ph(1,:)=0; Pm = V*Ph;
Ph = iV*Pp; Ph(1,:)=0; Pp = V*Ph;

% Extend cell averages
[ve] = extendDG(ucell,'P',0,'P',0);

% extract end values and cell averages for each element
uel = u(1,:); uer = u(end,:);
vj = ucell; vjm = ve(1:N); vjp = ve(3:N+2);

% Find elements that require limiting
vel = vj - minmod([(vj-uel)', (vj-vjm)', (vjp-vj)' ]);';
ver = vj + minmod([(uer-vj)', (vj-vjm)', (vjp-vj)' ]);';
ids = find(abs(vel-uel)>eps0 | abs(ver-uer)>eps0);

% Apply limiting when needed
if(~isempty(ids))
    % Extract local polynomials
    pm1 = Pm(:,ids) + ones(m+1,1)*vj(ids);
    p0 = u(:,ids);
    pp1 = Pp(:,ids+2) + ones(m+1,1)*vj(ids);

    % Compute smoothness indicators and WENO weights
    betam1 = diag(pm1'*Q*pm1); alpham1 = gammam1./(eps1 + betam1).^(2*p);
    beta0 = diag(p0'*Q*p0); alpha0 = gamma0./(eps1 + beta0).^(2*p);
    betap1 = diag(pp1'*Q*pp1); alphap1 = gammap1./(eps1 + betap1).^(2*p);

    alphas = alpham1 + alpha0 + alphap1;
    omm1 = alpham1./alphas; om0 = alpha0./alphas; omp1 = alphap1./alphas;

    % Compute limited function
    ulimit(:,ids) = pm1*diag(omm1) + p0*diag(om0) + pp1*diag(omp1);
end
return

```

---

The computation of the nonlinear weights is more involved, with its details outlined in the following example.

**Example 12.31.** Consider the smoothness indicator

$$\beta_i = \sum_{l=1}^m \int_{x_{i-1/2}}^{x_{i+1/2}} b^{2l-1} \left( \frac{d^l p_i(x)}{dx^l} \right)^2 dx,$$

and recall that

$$p_i(x) = \sum_{n=0}^m \tilde{p}_{i,n} P_n(r(x)), \quad x(r) = x_{j-1/2} + b \frac{1+r}{2}, \quad r \in [-1, 1].$$

Combining these expressions yields

$$\beta_i = \sum_{l=1}^m \int_{-1}^1 2^{2l-1} \left( \sum_{n=0}^m \tilde{p}_{i,n} P_n^{(l)}(r) \right)^2 dr.$$

We express this as

$$\beta_i = \sum_{l=1}^m \tilde{p}^T Q^l \tilde{p},$$

where  $\tilde{p} = (\tilde{p}_{i,0}, \dots, \tilde{p}_{i,m})^T$  and

$$Q_{ij}^l = \int_{-1}^1 P_i^{(l)}(r) P_j^{(l)}(r) dr.$$

The evaluation of  $P_i^{(l)}$  is accomplished by repeatedly applying the recurrence relation [103]:

$$P_i(r) = -\alpha_i P'_{i-1}(r) + \alpha_{i+1} P'_{i+1}(r), \quad \alpha_i = \frac{1}{\sqrt{(2i+1)(2i-1)}}.$$

This enables the exact computation of the entries of  $Q_{ij}^l$  by a quadrature of order  $m$ . Evaluating the smoothness indicator is then accomplished by

$$\beta_i = \tilde{p}^T Q \tilde{p},$$

where

$$Q = \sum_{l=1}^m 2^{2l-1} Q^l.$$

The implementation of this is illustrated in `WENOWeights.m`.

**Script 12.17. `WENODGWeights.m`: Smoothness indicator for a WENO-based limiter.**

---

```

function [Q,Xm,Xp] = WENODGWeights(m,iV);
% function [Q,Xm,Xp] = WENODGWeights(m,iV);
% Purpose: Compute operators to enable evaluation of WENO smoothness
% indicator and WENO polynomial of order m.
Q = zeros(m+1,m+1); Pmat = zeros(m+1,m+1); Xm = Pmat; Xp = Pmat;

% Compute quadrature points
[x,w] = LegendreGQ(m); Lambda = diag(w);

% Initial matrices of Legendre polynomials
for i=1:m+1;
    Pmat(i,:,:) = LegendreP(x,i-1)';
    Xm (i,:,:) = LegendreP(x-2,i-1)';
    Xp (i,:,:) = LegendreP(x+2,i-1)';
end

```

```

% Compute matrices corresponding to increasing order of derivative
for l=1:m
    % Set up operator to recover derivatives
    A = zeros(m+2-l,m+2-l); A(1,1) = 1/sqrt((2*l+1)*(2*l-1));
    A(m+2-l,m+2-l) = 1/(sqrt(2*(m+2)+1)*sqrt(2*(m+2)-1));
    for i=2:m-l+1
        Ah = 1/(sqrt(2*(l-1+i)+1)*sqrt(2*(l-1+i)-1));
        A(i,i) = Ah; A(i+1,i-1) = -Ah;
    end

    % Recover derivatives at quadrature points
    Ph1 = A\Pmat(1:m+1,:);
    Pmat(1:1,:) = 0; Pmat(1+1:m+1,:) = Ph1(1:m-l+1,:);

    % Compute smoothness operator for order l and update
    Qh = Pmat*Lambda*Pmat';
    Q = Q + 2^(2*l-1)*Qh;
end

% Initialize operator for smoothness indicator in nodal space
Q = iV'*Q*iV;

% Initialize interpolation matrices
Xp = iV'*Xp; Xm = iV'*Xm;
return

```

---

■

The development of WENO-based limiters was initiated in [91] and extended to general unstructured grids in [129]. Limiters that use the local polynomial and the evolution of its derivative have been proposed in [88, 90] and are known as Hermite-WENO limiters. These have the benefit of locality for high-order limiting, e.g., when more than the neighboring cells are used in the reconstruction.

Let us consider an example to understand the impact of WENO limiting on the accuracy of the scheme for smooth problems.

**Example 12.32.** We solve the linear wave equation

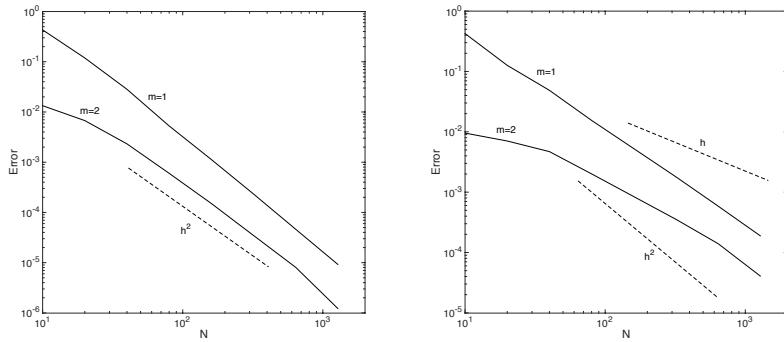
$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions and with  $u(x, 0) = \sin(\pi x)$ . The problem is solved using a discontinuous Galerkin method of order  $m$  with an upwind flux and a third order SSPRK method. We evaluate the solution at  $T = \pi$ .

Figure 12.17 shows the  $L^1$ - and  $L^\infty$ -error, obtained with a WENO limiter as the order of approximation increases.

In the  $L^1$ -error we observe  $\mathcal{O}(h^2)$  for  $m = 1$  and  $m = 2$ , i.e., a loss of accuracy occurs for  $m > 1$ . This is expected as the WENO limiter introduces an error of  $\mathcal{O}(h)$  at local extrema by using the extrapolated polynomial solutions in neighboring cells. This is also reflected in the  $L^\infty$ -error, which shows a convergence rate between one and two.

For the smooth case, there are several ways to overcome this reduction of accuracy with the natural one being to use a TVB limiter, (10.15), for the identification of troubled cells. Provided one chooses  $M$  sufficiently large, the local reduction of accuracy is eliminated. We shall discuss this further in subsection 12.5.1, and highlight the importance of choosing the troubled-cell indicator with care. ■



**Figure 12.17.** The impact of WENO limiting on a smooth solution. We show the  $L^1$ -error (left) and the  $L^\infty$ -error (right) when using the WENO limiter [128] with a TVD-based troubled-cell indicator function.

### 12.2.6 • Extrema preserving limiters

In many cases, physical considerations dictate that a computed variable cannot exceed certain values, e.g., a density must be semipositive, a temperature should not take values less than the absolute zero. With the appearance of artificial oscillations, such physical constraints may well be violated. Although some of the limiters discussed so far will alleviate this problem, their construction does not guarantee this without reducing the formal accuracy. Hence, we should seek schemes that guarantee positivity/maximum preservation, without adversely impacting their formal accuracy.

In order to pursue such a development, initiated in [125] in the context of both discontinuous Galerkin methods and WENO techniques, let us focus on the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to appropriate boundary and initial conditions. We recall from Theorem 4.5 that a monotone scheme

$$\bar{u}_j^{n+1} = G(\bar{u}^n)_j = \bar{u}_j^n - \frac{k}{h} (F_{j+1/2}^n - F_{j-1/2}^n)$$

satisfies the maximum principle

$$\min_{i \in S_j} \bar{u}_i^n \leq G(\bar{u}^n)_j \leq \max_{i \in S_j} \bar{u}_i^n,$$

where  $S_j$  is the set of cells in the stencil. Hence, a monotone scheme preserves the extrema of the solution, albeit at the cost of being first order accurate. Our goal is to extend this property to high-order schemes.

Let us recall that on each element  $D_j$ , we have  $m + 1$  quadrature points. On the standard element  $r \in [-1, 1]$ , we refer to these as  $r_i$ ,  $i = 0, \dots, m$ . We also have the associated positive quadrature weights  $w_i$ . The accuracy of the quadrature guarantees that

$$\bar{u}_j^n = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u_j(x) dx = \frac{1}{2} \int_{-1}^1 u_j(x(r)) dr = \frac{1}{2} \sum_{i=0}^m u_j(r_i) w_i,$$

where  $u_j(r_i) = u_j(x(r_i))$  for ease of notation.

In the case of a general high-order scheme in conservation form, the numerical flux is

$$F_{j \pm 1/2}^n = F(u_{j \pm 1/2}^-, u_{j \pm 1/2}^+).$$

To keep the notation simple, we introduce the function  $v_i$  as

$$v_i = \begin{cases} u_{j-1/2}^-, & i = -1, \\ u_j^n(r_i), & i = 0, \dots, m, \\ u_{j+1/2}^+, & i = m+1, \end{cases}$$

and express the flux difference as a telescoping sum

$$F_{j+1/2}^n - F_{j-1/2}^n = F(v_m, v_{m+1}) - F(v_{-1}, v_0) = \sum_{i=0}^m F(v_i, v_{i+1}) - F(v_{i-1}, v_i).$$

We are now ready to state the first of two key results [125].

**Theorem 12.33.** *Consider a discontinuous Galerkin method with a monotone flux, and let the cell solution  $u_j$  be an  $m$ th order polynomial. If there exist constants  $e$  and  $E$  such that*

$$e \leq (u_j^n(r_0), \dots, u_j^n(r_m), \bar{u}_j^n) \leq E$$

and

$$\frac{k}{h} \max_u |f'(u)| \leq \frac{1}{2} \min_i w_i,$$

then

$$e \leq \bar{u}_j^{n+1} \leq E.$$

**Proof.** Let us write the scheme as

$$\begin{aligned} \bar{u}_j^{n+1} &= G(u^n)_j = \bar{u}_j^n - \frac{k}{h} [F_{j+1/2}^n - F_{j-1/2}^n] \\ &= \frac{1}{2} \sum_{i=0}^m v_i w_i - \frac{k}{h} \sum_{i=0}^m F(v_i, v_{i+1}) - F(v_{i-1}, v_i) \\ &= \frac{1}{2} \sum_{i=0}^m \left( v_i - \frac{2k}{w_i h} (F(v_i, v_{i+1}) - F(v_{i-1}, v_i)) \right) w_i = \frac{1}{2} \sum_{i=0}^m G(v^n)_i w_i, \end{aligned}$$

where

$$\begin{aligned} G(v^n)_i &= v_i - \frac{2k}{w_i h} [F(v_i, v_{i+1}) - F(v_{i-1}, v_i)] \\ &= v_i - \lambda_i [F(v_i, v_{i+1}) - F(v_{i-1}, v_i)], \quad \lambda_i = \frac{2k}{h w_i}. \end{aligned}$$

Since the numerical flux is monotone, assuming that

$$\frac{k}{h} \max_u |f'(u)| \leq \frac{1}{2} \min_i w_i$$

guarantees that the scheme is monotone. This ensures that  $e \leq G(v^n)_i \leq E$ . Since  $w_i$  are positive and  $\sum_i w_i = 1$  for consistency, the result follows.  $\square$

In the case of Legendre–Gauss–Lobatto nodes, the CFL condition yields

$$\frac{k}{h} \max_u |f'(u)| \leq \frac{1}{2} \frac{2}{m(m+1)}.$$

Although this appears as a strong condition, its scaling is similar to that of the CFL condition for the unlimited scheme [56].

The main conclusion provided by this result is that to guarantee that the solution does not exceed the bounds, we must modify  $u_j^n$  at all quadrature points  $r_i$ . This can be achieved through the limiter

$$\tilde{u}_j^n(x) = \theta(u_j^n(x) - \bar{u}_j^n) + \bar{u}_j^n, \quad \theta = \min\left(\left|\frac{E - \bar{u}_j^n}{E_j - \bar{u}_j^n}\right|, \left|\frac{e - \bar{u}_j^n}{e_j - \bar{u}_j^n}\right|, 1\right), \quad (12.81)$$

where  $e_j$  and  $E_j$  are first defined as

$$e_j = \min_{x \in D_j} u_j^n(x), \quad E_j = \max_{x \in D_j} u_j^n(x).$$

However, as only the values at the quadrature points are relevant, it suffices to consider

$$e_j = \min_i u_j^n(r_i), \quad E_j = \max_i u_j^n(r_i).$$

We conclude that the limited scheme with the modified flux

$$F_{j \pm 1/2}^n = F(\tilde{u}_{j \pm 1/2}^-, \tilde{u}_{j \pm 1/2}^+)$$

guarantees that the solution remains bounded within the interval  $[e, E]$ , provided the time step is sufficiently small,

While this result is encouraging, the impact of this procedure on the accuracy remains unclear. To address this issue, we recall the following result [125].

**Theorem 12.34.** *If  $e \leq \bar{u}_j^n \leq E$ , then the limiter defined in (12.81) is of order  $m+1$ .*

**Proof.** If we can prove  $u_j^n(x) - \tilde{u}_j^n(x) = \mathcal{O}(h^{m+1})$ , the result follows. Let us focus on the upper bound. Since  $\bar{u}_j^n \leq E_j \leq E$ , we have

$$\tilde{u}_j^n(x) - u_j^n(x) = (\theta - 1)(u_j^n(x) - \bar{u}_j^n) = (E - E_j) \frac{u_j^n(x) - \bar{u}_j^n}{E_j - \bar{u}_j^n}.$$

The quantity  $E_j$  measures the overshoot of a polynomial approximation to the constant  $E$ , thus  $E - E_j = \mathcal{O}(h^{m+1})$ . Now assume that  $m > 0$  and recall the definition of  $E_j$  to obtain

$$\left| \frac{u_j^n(x) - \bar{u}_j^n}{E_j - \bar{u}_j^n} \right| \leq \max_x \left| \frac{u_j^n(x) - \bar{u}_j^n}{E_j - \bar{u}_j^n} \right| = \left| \frac{\min_x u_j^n(x) - \bar{u}_j^n}{\max_x u_j^n(x) - \bar{u}_j^n} \right| = \frac{|\min_x u_j^n(x) - \bar{u}_j^n|}{|\max_x u_j^n(x) - \bar{u}_j^n|}.$$

Since both the numerator and the denominator represent norms and  $u_j^n$  is a polynomial order  $m$ , the two norms are equivalent. Hence

$$\left| \frac{u_j^n(x) - \bar{u}_j^n}{E_j - \bar{u}_j^n} \right| \leq \frac{|\min_x u_j^n(x) - \bar{u}_j^n|}{|\max_x u_j^n(x) - \bar{u}_j^n|} = C_m.$$

For the special case of  $m = 0$  we recover the classic monotone finite volume method which preserves the extrema. The proof for the lower bound is similar.  $\square$

Hence, the limiter in (12.81) ensures extrema preservation without impacting the accuracy of the scheme. While our focus has been on the discontinuous Galerkin method, the extension to WENO methods follows the same approach. We refer to [125, 127] for details. Extensions to multidimensional cases and systems are, likewise, discussed in detail in [125, 127, 126].

Finally, let us discuss the implementation of the extrema preserving limiter, illustrated in `extprelimitDG.m`. As discussed previously, we rely on a simple troubled-cell indicator to minimize the impact of the limiter. However, since the limiter does not impact the overall accuracy, this is a less critical component than for the slope limiter.

---

**Script 12.18. `extprelimitDG.m`: Extrema preserving limiter for discontinuous Galerkin scheme.**

---

```

function ulimit = extprelimitDG(x,u,m,h,N,V,iV,umin,umax);
% function ulimit = extprelimitDG(x,u,m,h,N,V,iV,umin,umax);
% Purpose: Apply extrema preserving limiter to piecewise
% polynomial solution u - an m'th order polynomial
eps0=1.0e-8;

% Compute cell averages and cell centers
uh = iV*u; uh(2:(m+1),:)=0; uavg = V*uh; ucell = uavg(1,:); ulimit = u;

% Extend cell averages
[ve] = extendDG(ucell,'P',0,'P',0);

% extract end values and cell averages for each element
uel = u(1,:); uer = u(end,:);
vj = ucell; vjm = ve(1:N); vjp = ve(3:N+2);

% Find elements that require limiting
vel = vj - minmod([(vj-uel);(vj-vjm);(vjp-vj)]);
ver = vj + minmod([(uer-vj);(vj-vjm);(vjp-vj)]);
ids = find(abs(vel-uel)>eps0 | abs(ver-uer)>eps0);

% Apply limiting when needed
if(~isempty(ids))
    minu = min(u(:,ids),[],1); maxu = max(u(:,ids),[],1);
    ulimmax = abs((umax-ucell(ids))./(maxu-ucell(ids)));
    ulimmin = abs((umin-ucell(ids))./(minu-ucell(ids)));
    theta = min([ulimmax; ulimmin; ones(1,length(ids))],[],1);
    ulimit(:,ids) = (ones(m+1,1)*theta).*(u(:,ids)-uavg(:,ids)) + uavg(:,ids);
end
return

```

---

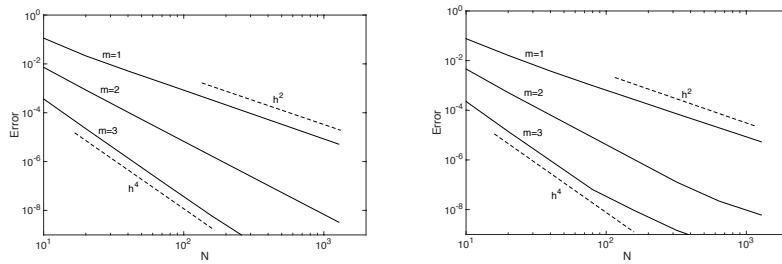
We investigate the performance of the limiter for smooth problems through an example.

**Example 12.35.** We consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad x \in [-1, 1],$$

subject to periodic boundary conditions and the initial conditions  $u(x, 0) = \sin(\pi x)$ . The problem is solved using a discontinuous Galerkin method of order  $m$  with an upwind flux and a third order SSPRK method. We consider the solution at  $T = \pi$ .

Figure 12.18 shows the  $L^1$ - and  $L^\infty$ -error, obtained when using the extrema preserving limiter. In accordance with Theorem 12.34 we observe that the rate of convergence does not deteriorate. ■



**Figure 12.18.** The impact of the extrema preserving limiting on a smooth solution. We show the  $L^1$ -error (left) and the  $L^\infty$ -error (right) using the limiter [125] with a TVD troubled-cell indicator. The slight decrease in the order of convergence in  $L^\infty$  for high values of  $N$  is caused by errors in the temporal integration.

## 12.3 • Related formulations

Before proceeding with a in-depth evaluation of discontinuous Galerkin methods, it is worth it to step back and compare them to closely related methods discussed in the literature. To establish such connections, we resort to the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad x \in \Omega_x,$$

subject to appropriate boundary and initial conditions. While we discuss the different schemes for this particular problem, the connections remain valid in general. As for the discontinuous Galerkin method, we begin by assuming that  $u(x, t) \simeq u_b(x, t)$ , where the global approximation  $u_b(x, t)$  is a piecewise polynomial of order  $m$

$$u_b(x, t) = \bigoplus_{j=1}^N u_j(x, t), \quad u_j(x, t) = \sum_{i=0}^m u_{j,i}(t) \psi_{j,i}(x),$$

where  $\psi_{j,i}$  is a polynomial basis defined on element  $\mathcal{D}_j$ . For simplicity of the comparative discussion, we assume in this section that  $\psi_{j,i} = \tilde{P}_i(r(x))$ . Here  $\tilde{P}_i(r) = \sqrt{\gamma_i} P_i(s)$  is the unnormalized Legendre polynomial, introduced in subsection 12.1.1.

For each of the  $N$  elements, we require that the equation is satisfied in the following sense:

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \left( \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x} \right) \Psi(x) dx = [\Phi(x)(G_j(x, u) - G_j^*(x, u))]_{x_{j-1/2}}^{x_{j+1/2}}. \quad (12.82)$$

We recognize this as a Petrov–Galerkin formulation in strong form, where  $\Psi$  and  $\Phi$  play the role of test functions. The functions

$$\begin{aligned} G_j(x, u) &= G_j(x, t, u_j, (u_j)_x, \dots), \\ G_j^*(x, u) &= G_j^*(x, t, u_{j-1}, u_j, u_{j+1}, (u_{j-1})_x, (u_j)_x, (u_{j+1})_x, \dots) \end{aligned}$$

are defined along the interfaces and are responsible for connecting the elements. As for the discontinuous Galerkin method, these play an essential role for stability of the scheme. It is clear that consistency of the scheme (12.82) follows from the consistency of  $G_j^*(x, u)$ , i.e.,  $G_j(x_{j\pm 1/2}, u) = G_j^*(x_{j\pm 1/2}, u)$  for an exact solution  $u$ .

The definition of the test functions  $\Psi$  and  $\Phi$ , and the interface functions  $G_j$  and  $G_j^*$ , provides a great deal of flexibility in the design of different schemes with different properties. For instance, if we choose

$$\Psi(x) = \Phi(x) = \psi_j(x), \quad G_j(x_{j+1/2}, u) = f(u_{j+1/2}^-), \quad G_j^*(x_{j+1/2}, u) = F(u_{j+1/2}^-, u_{j+1/2}^+),$$

we recover the discontinuous Galerkin method in strong form, (12.7). However, as we shall see shortly, there are several other choices to consider.

Prior to entering into a detailed discussion, let us simplify the notation by introducing

$$M_{ij} = \int_D \psi_j(x) \Psi_i(x) dx, \quad S_{ij} = \int_D (\psi_j(x))_x \Psi_i(x) dx,$$

where we have eliminated the direct reference to the element. With this notation the scheme becomes

$$M \frac{du}{dt} + Sf = [\Phi(x)(G_j(x, u) - G_j^*(x, u))]_{x_{j-1/2}}^{x_{j+1/2}}, \quad (12.83)$$

where  $u$  and  $f$  are  $(m+1)$ -vectors containing, respectively, the solution and the flux.

### 12.3.1 • Spectral penalty methods

Spectral penalty methods were first proposed in [38, 39] and later developed in a series of papers [52, 47, 49, 50, 15]. The starting point is the choice of  $m+1$  quadrature points,  $\tilde{x}_i$ , within each element. In case we employ a local nodal polynomial basis these new points may, but do not need to, be the same as the points on which the local representation is based. Furthermore, we assume that  $\tilde{x}_i$  contains the two end points of the domain, e.g., a Gauss–Lobatto quadrature, and take as  $\tilde{x}_0 = x_{j-1/2}$  and  $\tilde{x}_m = x_{j+1/2}$ .

Spectral penalty methods employ local test functions as  $\Phi_i(x) = Q_i(x)\delta(x - \tilde{x}_i)$ , where  $Q_i$  is a family of polynomials to be defined shortly. Typically, many of these are zero and only  $Q_0$  and  $Q_m$  take nontrivial values. The test functions  $\Psi_i$  can be defined in different ways. Finally, we assume that  $G_j(x, u) = f(u_j)$  and  $G_j^*(x_{j+1/2}, u) = F_{j+1/2}$

is some numerical flux. This yields a collocation penalty method of the form

$$M \frac{du}{dt} + Sf = Q_m(\tilde{x}) \left[ f(u_{j+1/2}^-) - F_{j+1/2} \right] - Q_0(\tilde{x}) \left[ f(u_{j-1/2}^+) - F_{j-1/2} \right]. \quad (12.84)$$

The similarity to the discontinuous Galerkin method is clear, although there are subtle differences resulting from the choices of the test functions  $\Psi$  and the functions  $Q_i$ .

Let us explore some of the properties of spectral penalty methods through an example.

**Example 12.36.** Consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial au}{\partial x} = 0,$$

subject to periodic boundary conditions and some appropriate initial condition.

Let us first consider the case where  $\Psi_i(x) = \delta(x - \tilde{x}_i)$  and simplify the general scheme in (12.84) by assuming that the interpolation and the collocation points are the same. We use a Lagrangian basis, based on the Legendre–Gauss–Lobatto quadrature points, to represent the solution. Furthermore, if we assume that  $a > 0$  and select an upwind flux  $F_{j+1/2} = au_{j+1/2}^-$ , we recover the elementwise scheme

$$\frac{du}{dt} + Da u = -Q_0(\tilde{x}) \left[ au_{j-1/2}^+ - au_{j-1/2}^- \right], \quad (12.85)$$

where  $D$  represents the differentiation matrix. The missing piece of the scheme is the definition of  $Q_0$ . As a first choice, let us consider

$$Q_0(x) = \tau \frac{(1 - r(x)) \tilde{P}'_m(r(x))}{2 \tilde{P}'_m(-1)},$$

where  $r(x) : [x_{j-1/2}, x_{j+1/2}] \rightarrow [-1, 1]$ . We note that  $Q_0(\tilde{x}) = \tau e_0$  by the definition of the Gauss–Lobatto points.

To understand stability in the single element, let us first assume that  $u_{j-1/2}^- = 0$ . Since we can consider the error equation for general boundary conditions, analysis of the scheme with homogeneous boundary conditions suffices. Define  $W_{ii} = w_i$  as containing the quadrature weights for the Legendre–Gauss–Lobatto quadrature and left-multiply (12.85) by  $u^T W$  to recover

$$u^T W \left( \frac{du}{dt} + Da u \right) = -w_0 \tau a (u_{j-1/2}^+)^2.$$

By the accuracy of the quadrature, we obtain

$$\frac{1}{2} \frac{d}{dt} \|u_j\|_b^2 + \frac{1}{h} [au^2]_{x_{j-1/2}}^{x_{j+1/2}} = -w_0 \tau a (u_{j-1/2}^+)^2,$$

where we define the discrete  $L^2$ -norm as

$$\|u\|_b^2 = u^T W u.$$

Hence, if we choose

$$\tau \geq \frac{1}{h w_0} = \frac{m(m+1)}{2h}, \quad (12.86)$$

we can guarantee asymptotic stability in the single element. This is the classic penalty method proposed in [38] and it is equivalent to a spectral collocation method with a weakly imposed boundary condition. While it is a powerful method, reliance on quadrature formulas to establish its stability limits the analysis to linear problems with constant coefficient.

Whereas condition (12.86) is necessary in the multi-element case, it may not be sufficient. To appreciate this, consider the contribution at  $x_{j+1/2}$  from two neighboring elements and require

$$-\frac{1}{b}(u_{j+1/2}^-)^2 + \left(\frac{1}{b} - \tau w_0\right)(u_{j+1/2}^+)^2 + \tau w_0 u_{j+1/2}^- u_{j+1/2}^+ \leq 0,$$

to ensure stability. Expressing this as a symmetric quadratic form and requiring that its eigenvalues to be nonpositive, results in the condition

$$\tau = \frac{2}{w_0 b}$$

as a sufficient condition for stability. Nevertheless, extensive computational results [52] confirm that the weaker elementwise condition (12.86) is, in fact, often sufficient.

As an alternative, consider the case with  $\Psi = \psi_j$ , which results in the scheme

$$M \frac{du}{dt} + S a u = -\tau e_0 [a u_{j-1/2}^+ - a u_{j-1/2}^-]. \quad (12.87)$$

Stability can be established as previously, although in this case we do not need a quadrature formula. This allows for more flexibility in the analysis. The bounds for stability are the same as for the collocation scheme.

If we express (12.87) in explicit form

$$\frac{du}{dt} + D a u = -\tau M^{-1} e_0 [a u_{j-1/2}^+ - a u_{j-1/2}^-],$$

it is easy to show that [15]

$$(M^{-1} e_0)_i = \frac{\tilde{P}'_{m+1}(\tilde{x}_i) + \tilde{P}'_m(\tilde{x}_i)}{2}.$$

This scheme is a spectral Galerkin penalty method and has been developed further for general grids in [51, 55].

The extension of these ideas to linear systems of equations requires the use of characteristic variables but is, otherwise, straightforward. ■

A substantial development of spectral penalty methods has taken a slightly different path than the one outlined above by exploring the freedom in the definition of  $G_j$  and  $G_j^*$ . An advantage of weakly imposed boundary conditions is that they allow one to impose very complex boundary conditions. This direction of research, initiated in [52, 47, 49], is based on the development of well posed boundary operators which, subsequently, guide the definition of  $G_j$  and  $G_j^*$ . While less relevant to purely hyperbolic systems, this enables the development of stable, multi-element schemes for advection-diffusion equations and the compressional Navier-Stokes equations, even in the limit of vanishing diffusion. Although the analysis is based on the linearized and frozen system, computational evidence suggests that the scheme remains stable also for nonlinear problems.

### 12.3.2 ■ Spectral finite volume schemes

The spectral finite volume method was developed in a series of papers [112, 113, 114, 116, 81, 101]. It relies on the use of a nodal basis for the representation of the solution and requires the equation to be satisfied in a way that mimics a sub-cell finite volume method on each element.

To develop the scheme, choose  $G_j(x_{j+1/2}, u) = f(u_{j+1/2}^-)$  and  $G_j^*(x_{j+1/2}, u) = F_{j+1/2}$  for some appropriate numerical flux. This leaves open only the specification of the test functions  $\Psi$  and  $\Phi$ . Let us term  $\{x_{j,i}\}$  as the  $m+1$  points of element  $D_j$ , i.e.,  $x_{j,0} = x_{j-1/2}$  and  $x_{j,m} = x_{j+1/2}$ , and identify  $m$  sub-cells with the  $m+1$  cell interfaces,  $x_i^c \in [x_{j,i}, x_{j,i+1}]$  for  $i = 0, \dots, m$ . This defines the space of test functions as

$$\Psi_i(x) = \Phi_i(x) = C, \quad x \in [x_i^c, x_{i+1}^c], \quad i = 0, \dots, m-1,$$

where  $x_0^c = x_{j-1/2}$  and  $x_m^c = x_{j+1/2}$ . Hence, the test functions comprise piecewise constant functions defined in the interior of the element. Using these test functions in (12.83) we recover the sub-cell scheme

$$\frac{d\bar{u}_{j,i}}{dt} = -\frac{F(x_{i+1}^c) - F(x_i^c)}{x_{i+1}^c - x_i^c},$$

for the  $m$  cells. Here

$$\bar{u}_{j,i} = \frac{1}{x_{i+1}^c - x_i^c} \int_{x_i^c}^{x_{i+1}^c} u_j(x) dx$$

is the subcell cell average. A global scheme emerges by defining  $F(x_0^c) = F_{j-1/2}$  and  $F(x_{m+1}^c) = F_{j+1/2}$ , and one can employ a monotone numerical flux at the internal interfaces.

The high-order accuracy of the scheme arises by using the cellwise polynomial for the  $\mathcal{O}(h^m)$  accurate reconstruction of the solution at the cell interfaces. Furthermore, for a nonsmooth solution, the sub-cell structure enables the use of local cellwise limiters.

At first, the scheme may seem very efficient as there is no volume term due to the sub-cell structure. However, the need to evaluate the numerical fluxes at all internal interfaces dictates the cost, which is comparable to that of a standard discontinuous Galerkin method.

The key challenge of the spectral volume method is how to identify the sub-cells, as this impacts accuracy and, in particular, the stability of the scheme. For one-dimensional problems, this task is manageable for moderate orders of the approximation. However, for multidimensional problems, this remains a substantial challenge. Furthermore, no general stability theory is available apart from studies for low-order schemes and special layouts of the sub-cells [112]. A detailed comparison between the discontinuous Galerkin method and the spectral volume method [122] indicates that the spectral volume method generally allows for a larger time step but typically results in worse errors at comparable resolution.

### 12.3.3 ■ Spectral difference schemes

Whereas the spectral volume method relies on the reconstruction of the solution, in the spirit of the finite volume method, the spectral difference method, first proposed in [80, 115], is based on a reconstruction of the flux.

To develop this family of methods, we recall that  $u_j$  is represented locally on an element as a polynomial, based on a number of grid points,  $\{x_{j,i}\}$ . These points are general and need not include the end points of the element.

Let us assume that  $f(u_j)$  is expressed as a polynomial of order  $m+1$ , i.e.,

$$f_h(u_j) = \sum_{i=0}^{m+1} f_h(u_b(\tilde{x}_{j,i})) \tilde{\ell}_i(x),$$

where we, for simplicity, have taken the local basis to be a nodal basis, based on  $m+2$  grid points  $\tilde{x}_{j,i}$  in element  $D_j$ , and  $\tilde{\ell}_i(x)$  is the associated Lagrange polynomial. The end points must be included in  $\tilde{x}_{j,i}$ . Note that for this to be exact,  $f_h(u)$  must either be the projection of  $f(u)$  onto the polynomial space of order  $m+1$  or  $f(u)$  must be linear.

In the spirit of the finite difference method, we use the solution  $u_j$  to evaluate the flux at all interior points and use a numerical flux to recover the values of the flux at the end points. We write the corrected flux  $\tilde{f}_h$  as

$$\tilde{f}_h(u) = (F_{j-1/2} - f_h(u_{j-1/2}^+)) \tilde{\ell}_0(x) + (F_{j+1/2} - f_h(u_{j+1/2}^-)) \tilde{\ell}_{m+1}(x) + f_h(u_j). \quad (12.88)$$

If we require the conservation law to be satisfied in a pointwise sense at  $x_{j,i}$ , we observe

$$\frac{d\mathbf{u}}{dt} + \tilde{\mathbf{D}}\tilde{f} = 0,$$

where  $\tilde{\mathbf{D}}$  is now a rectangular differentiation matrix to account for the transformation between  $\tilde{x}_{j,i}$  and  $x_{j,i}$ . Inserting (12.88) into this expression yields

$$\frac{d\mathbf{u}}{dt} + \tilde{\mathbf{D}}\tilde{f} = \tilde{\ell}'_{m+1}(\mathbf{x}_j) (f_h(u_{j+1/2}^-) - F_{j+1/2}) + \tilde{\ell}'_0(\mathbf{x}_j) (f_h(u_{j-1/2}^+) - F_{j-1/2}),$$

where  $\mathbf{x}_j$  is the vector of points at which the equation is satisfied. This is the spectral difference scheme as proposed in [80, 115].

To connect this to the general formulation in (12.84), we define  $\Psi_i(x) = \delta(x - x_{j,i})$ ,  $\Phi_0(x) = -\tilde{\ell}'_0(x) \delta(x - x_{j,0})$ , and  $\Phi_m(x) = \tilde{\ell}'_{m+1}(x) \delta(x - x_{j,m})$ . By inspection, this recovers the spectral difference scheme.

The use of different sets of points for the solution and the flux representation was already proposed in [73, 71, 72]. For the one-dimensional case and multidimensional tensorial elements, staggered grid spectral methods are identical to the spectral difference method.

The accuracy of the scheme is ensured by the polynomial representation of the local solution. Furthermore, conservation of the method was established in the early work. In the one-dimensional case, one can show a direct equivalence between the spectral volume method and the spectral difference method [107], much in the spirit of the connection between one-dimensional finite difference and finite volume methods.

Since we now have two sets of points, we need to understand the impact of this on stability. To gain some insight into this, let us resort to an example.

**Example 12.37.** We consider again the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial au}{\partial x} = 0, \quad a > 0,$$

subject to periodic boundary conditions and some appropriate initial condition.

In the case of an upwind flux, the spectral difference scheme becomes

$$\frac{d\mathbf{u}}{dt} + a\tilde{\mathbf{D}}\mathbf{u} = \tilde{\ell}'_0(\mathbf{x}_j)(au_{j-1/2}^+ - au_{j-1/2}^-).$$

To understand stability, let us, for simplicity, assume that  $[x_{j-1/2}, x_{j+1/2}] = [-1, 1]$  and take the solution points,  $x_{j,i}$ , to be the Legendre–Gauss quadrature points and the flux points,  $\tilde{x}_{j,i}$ , as a set of  $m+2$  points that, as a minimum, includes  $x = -1$ .

If we define  $W_{ii} = w_i$  as the matrix of Legendre–Gauss weights and left-multiply with  $\mathbf{u}^T W$  we recover

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_b^2 = -\frac{a}{2} (u_b^2(1) - u_b^2(-1)) + \mathbf{u}^T W \tilde{\ell}'_0(\mathbf{x}_j) f_C, \quad (12.89)$$

where  $f_C = au_{j-1/2}^+ - au_{j-1/2}^- = au_b(-1)^+ - au_b(-1)^-$  is the correction of the flux at the cell interface.

We realize that

$$\mathbf{u}^T W \tilde{\ell}'_0(\mathbf{x}_j) = \int_{-1}^1 u_b(x) \tilde{\ell}'_0(x) dx = -u_b(-1) - \int_{-1}^1 u'_b(x) \tilde{\ell}'_0(x) dx,$$

and recover the result

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_b^2 = -\frac{a}{2} (u_b^2(1) - u_b^2(-1)) - f_C u_b(-1) - f_C \int_{-1}^1 u'_b(x) \tilde{\ell}'_0(x) dx.$$

The first term is nonpositive and, thus, does not pose a problem in terms of stability. Likewise, the next two terms yield  $-a/2f_C^2$  at the interface. This term is equivalent to that of the discontinuous Galerkin method and, thus, stable. To establish stability we need to control the remaining term:

$$-f_C \int_{-1}^1 u'_b(x) \tilde{\ell}'_0(x) dx = -f_C \mathbf{u}^T \int_{-1}^1 \ell'(x) \tilde{\ell}'_0(x) dx,$$

where  $\ell(x) = (\ell_0(x), \dots, \ell_m(x))^T$ .

We can now exploit the choice of the points  $\{\tilde{x}_{j,i}\}$ . If we consider the Legendre–Gauss–Radau points, we have

$$\tilde{\ell}'_0(x) = \frac{(-1)^{m+1}}{2} (\tilde{P}_{m+1}(x) - \tilde{P}_m(x)),$$

by the definition of the Gauss–Radau points [53]. This implies that

$$\int_{-1}^1 \ell'(x) \tilde{\ell}'_0(x) dx = \frac{(-1)^{m+1}}{2} \int_{-1}^1 \ell'(x) (\tilde{P}_{m+1}(x) - \tilde{P}_m(x)) dx = 0,$$

since  $\ell'_i(x)$  is a polynomial of order  $m-1$  and  $P_m$  is orthogonal to all polynomials of order less than  $m$ . Hence, if we choose  $m+2$  Legendre–Gauss–Radau points as  $\tilde{x}_{j,i}$ , stability follows from stability of the discontinuous Galerkin method.

If a different choice of the  $m+2$  flux points is considered, one needs to pursue a different approach to establish stability, initiated in [65]. The basic idea is to consider stability in a different norm, characterized by

$$\|u_b\|_M^2 = \mathbf{u}^T \mathbf{M} \mathbf{u}, \quad \mathbf{M} = \mathbf{W} + \mathbf{C},$$

where  $\mathbf{W}$  is the diagonal matrix of the quadrature weights and  $\mathbf{C}$  is a positive definite correction. In order to avoid additional terms from the differentiation matrix, one constructs  $\mathbf{C}$  such that  $\mathbf{C}\mathbf{D} = 0$ .

In the case that  $u_b$  is a polynomial of order no higher than  $m$ , we observe that  $\mathbf{D}^{m+1}u_b = 0$ , and take  $\mathbf{C} = c(\mathbf{D}^m)^T \mathbf{D}^m$ , where  $c$  a suitable constant. This now yields a term in (12.89) as

$$f_C \left[ \int_{-1}^1 u'_b(x) \tilde{\ell}_0(x) dx - c \int_{-1}^1 u_b^{(m)}(x) \tilde{\ell}_0^{(m+1)}(x) dx \right]$$

or

$$f_C \mathbf{u}^T \left[ \int_{-1}^1 \ell'(x) \tilde{\ell}_0(x) dx - c \int_{-1}^1 \ell^{(m)}(x) \tilde{\ell}_0^{(m+1)}(x) dx \right].$$

If we employ the freedom to choose the flux points and, thus, define  $\tilde{\ell}_0(x)$ , one can prove that the constant  $c$  is positive and decays quickly with  $m$ . Hence,  $\mathbf{u}^T \mathbf{M} \mathbf{u}$  approaches  $\|u_b\|^2$  and is in fact a norm, provided  $u$  is analytic. We can select the interior Legendre–Gauss points, i.e., the roots of the polynomial  $\tilde{P}_{m+1}$ , as the internal flux points.

In the above discussion, the solution points  $\{x_{j,i}\}$  are taken to be the Legendre–Gauss points to ensure accuracy in the integration. This can be relaxed by considering an approach closer to that of a discontinuous Galerkin method where exact integration is accomplished through an exact mass matrix. This is developed in [65]. However, it does not yield qualitative differences.

Finally, by the direct connection between the spectral volume methods and the spectral difference method [107], the above result establishes stability for the one-dimensional spectral volume method as well. ■

The main conclusion of the above discussion is that stability of the spectral difference method requires special choices of the flux points, while the method enjoys substantial freedom in the choice of the solution points. This was also observed in [108], where heuristic choices for the points were discussed. However, in the case of multidimensional problems and unstructured grids, stability remains a substantial challenge and no analytic results are known to guide the choice of appropriate flux points. Nevertheless, the generalization of spectral difference methods at moderate order on general unstructured grids has been pursued with substantial success; see, e.g., [80, 115, 83, 102].

### 12.3.4 • Flux reconstruction schemes

Finally, let us consider schemes based on flux reconstruction which, as we shall see, is closely related to the spectral difference method. We assume that the solution  $u_b$  is a

piecewise polynomial of order  $m$ , based on the  $m + 1$  grid points  $\{x_{j,i}\}$  in each cell. These are typically taken to be Legendre–Gauss–Lobatto points, but other choices are possible. Similarly, we assume that  $f(u)$  is represented as a piecewise polynomial of order  $m + 1$ , based on the  $m + 2$  grid points  $\{\tilde{x}_{j,i}\}$ .

Following the idea of the spectral difference method, we express a reconstructed flux as

$$\tilde{f}_b(u) = f_b(u) + (F_{j+1/2} - f_b(u_{j+1/2}^-))g_R(x) + (F_{j-1/2} - f_b(u_{j-1/2}^+))g_L(x),$$

where  $F_{j\pm 1/2}$  are the numerical fluxes and  $g_{L,R}(x)$  are functions to be defined shortly. For consistency, we must have

$$g_L(x_{j-1/2}) = 1, \quad g_L(x_{j+1/2}) = 0, \quad g_R(x_{j-1/2}) = 0, \quad g_R(x_{j+1/2}) = 1, \quad (12.90)$$

and we furthermore require that  $g_L$  and  $g_R$  are polynomials of order  $m + 1$ . This yields the scheme

$$\frac{du_j}{dt} + \tilde{D}f = (F_{j+1/2} - f_b(u_{j+1/2}^-)) \frac{dg_R(x_j)}{dx} + (F_{j-1/2} - f_b(u_{j-1/2}^+)) \frac{dg_L(x_j)}{dx}.$$

If we choose

$$G_j(x_{j+1/2}, u) = f(u_j(x_{j+1/2}^-)), \quad G_j^*(x_{j+1/2}, u) = F(u_{j+1/2}^-, u_{j+1/2}^+),$$

and

$$\Psi_i(x) = \delta(x - x_{j,i}), \quad \Phi_i(x) = Q_i(x) \delta(x - x_{j,i}),$$

in (12.83) we recover

$$\frac{du_j}{dt} + \tilde{D}f = Q_m(x_j)(F_{j+1/2} - f_b(u_{j+1/2}^-)) + Q_0(x_j)(F_{j-1/2} - f_b(u_{j-1/2}^+)).$$

Defining

$$Q_0(x) = -\frac{dg_L(x)}{dx}, \quad Q_m(x) = \frac{dg_R(x)}{dx},$$

we recognize this scheme as a penalty method, discussed in subsection 12.3.1. Similarly, the definition

$$Q_0(x) = -\tilde{\ell}'_0(x), \quad Q_m(x) = \tilde{\ell}'_{m+1}(x)$$

results in the spectral difference scheme, discussed in subsection 12.3.3.

This interpretation of flux reconstruction schemes was first discussed in [64], albeit with a focus on their connections to discontinuous Galerkin methods and the spectral difference scheme.

As discussed in the context of penalty methods in subsection 12.3.1, the choice of the two functions  $g_{L,R}$  plays a central role for stability. As done previously, let us explore this through an example.

**Example 12.38.** We focus on the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial au}{\partial x} = 0, \quad a > 0,$$

subject to periodic boundary conditions and some appropriate initial condition.

In this case, the flux reconstruction scheme is given as

$$\frac{du_j}{dt} + a\tilde{D}u_j = (F_{j+1/2} - au_{j+1/2}^-) \frac{dg_R(x_j)}{dx} + (F_{j-1/2} - au_{j-1/2}^+) \frac{dg_L(x_j)}{dx},$$

which we write as

$$\frac{du_j}{dt} + a\tilde{D}u_j = f_R \frac{dg_R(x_j)}{dx} + f_L \frac{dg_L(x_j)}{dx}, \quad (12.91)$$

with the straightforward definition of  $f_{L,R}$  as the flux differences at the two cell interfaces.

To identify conditions on  $g_{L,R}$  that result in a stable family of schemes, we follow [111] and left-multiply (12.91) by  $\mathbf{u}^T \mathbf{M}$ , where  $\mathbf{M}$  is the exact mass matrix. This yields the energy statement

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_b^2 + \frac{a}{2} (u_b^2(1) - u_b^2(-1)) = f_R \int_{-1}^1 u_b \frac{dg_R}{dx} dx + f_L \int_{-1}^1 u_b \frac{dg_L}{dx} dx,$$

where we, for simplicity, have assumed that  $[x_{j-1/2}, x_{j+1/2}] = [-1, 1]$ . By recalling (12.90), integration by parts yields

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \|u_b\|_b^2 + \frac{a}{2} (u_b^2(1) - u_b^2(-1)) &= f_R \left[ u_b(1) - \int_{-1}^1 \frac{\partial u_b}{\partial x} g_R dx \right] \\ &\quad + f_L \left[ -u_b(-1) - \int_{-1}^1 g_L \frac{\partial u_b}{\partial x} dx \right]. \end{aligned}$$

Employing the same argument as in Ex. 12.37, the choice

$$g_L(x) = \frac{(-1)^{m+1}}{2} (\tilde{P}_{m+1}(x) - \tilde{P}_m(x)), \quad g_R(x) = \frac{1}{2} (\tilde{P}_{m+1}(x) + \tilde{P}_m(x))$$

ensures that the two integrals vanish by orthogonality, and we recover a discontinuous Galerkin method with a nodal basis.

To obtain alternative schemes, we recall that since  $u_b$  is a polynomial of order  $m$ , it follows that  $D^{m+1}u = 0$ . Hence, left-multiplying (12.91) with  $D^m$  yields

$$\frac{d}{dt} \mathbf{u}^{(m)} = f_R \frac{d^{m+1} g_R}{dx^{m+1}}(x) + f_L \frac{d^{m+1} g_L}{dx^{m+1}}(x).$$

If we further multiply by  $(\mathbf{u}^{(m)})^T \mathbf{M}$  we recover

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \|u_b^{(m)}\|_b^2 &= f_R \int_{-1}^1 u_b^{(m)} g_R^{(m+1)} dx + f_L \int_{-1}^1 u_b^{(m)} g_L^{(m+1)} dx \\ &= 2f_R u_b^{(m)} g_R^{(m+1)} + 2f_L u_b^{(m)} g_L^{(m+1)}. \end{aligned}$$

The last reduction follows since  $u_b$  is a polynomial of order  $m$  and  $g_{L,R}$  are polynomials of order  $m+1$ , i.e.,  $u_b^{(m)}$  and  $g_{L,R}^{(m+1)}$  are constant. Let us now consider

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \left( \|u_b\|_b^2 + \frac{c}{2} \|u_b^{(m)}\|_b^2 \right) &= -\frac{a}{2} (u_b(1)^2 - u_b(-1)^2) \\ &\quad + f_R \left[ u_b(1) - \int_{-1}^1 \frac{\partial u_b}{\partial x} g_R dx + c u_b^{(m)} g_R^{(m+1)} \right] \\ &\quad - f_L \left[ u_b(-1) + \int_{-1}^1 \frac{\partial u_b}{\partial x} g_L dx - c u_b^{(m)} g_L^{(m+1)} \right]. \end{aligned}$$

Provided that  $c \geq 0$  and  $u$  is analytic such that  $\|u_b\|_b^2 + \frac{c}{2} \|u_b^{(m)}\|_b^2$  is a norm uniformly in  $m$ , stability follows by ensuring that

$$\int_{-1}^1 \ell'(x) g_{L,R}(x) dx - c \ell^{(m)} g_{L,R}^{(m+1)} = 0.$$

To understand the bounds that must be imposed on  $c$  to ensure that the modified norm is a norm, consider

$$\|u_b\|_b^2 + \frac{c}{2} \|u_b^{(m)}\|_b^2 = \sum_{i=0}^m \tilde{u}_i^2 + \frac{c}{2} (a_m m!)^2 \tilde{u}_m^2 = \sum_{i=0}^{m-1} \tilde{u}_i^2 + \left(1 + \frac{c}{2} (a_m m!)^2\right) \tilde{u}_m^2,$$

where

$$a_m = \frac{(2m)!}{2^m (m!)^2},$$

is the leading monomial term of the Legendre polynomial [3]. Hence, we conclude that

$$\frac{-2}{(2m+1)(a_m m!)^2} \leq c < \infty.$$

As for the spectral penalty method, we recover a lower bound for the parameter to guarantee stability.

To sketch how one recovers suitable functions  $g_{L,R}$ , let us assume they are symmetric,  $g_L(x) = g_R(-x)$ . In this case, we may consider the condition

$$\int_{-1}^1 \ell'(x) g(x) dx - c \ell^{(m)} g^{(m+1)} = 0.$$

By carefully evaluating the properties of the Lagrange polynomials to find conditions for  $g$ , [111] proposes the general polynomials

$$\begin{aligned} g_L(x) &= \frac{(-1)^m}{2} \left( \tilde{P}_m(x) - \frac{\eta_m \tilde{P}_{m-1}(x) + \tilde{P}_{m+1}(x)}{1 + \eta_m} \right), \\ g_R(x) &= \frac{1}{2} \left( \tilde{P}_m(x) + \frac{\eta_m \tilde{P}_{m-1}(x) + \tilde{P}_{m+1}(x)}{1 + \eta_m} \right), \end{aligned}$$

where the constant  $\eta_m$  is given as

$$\eta_m = c \frac{(\alpha_m m!)^2}{\gamma_m}, \quad \gamma_m = \frac{2}{2m+1}.$$

By specifying the constant  $c$  we can identify different schemes, e.g., taking  $c = 0$  leads to the discontinuous Galerkin method based on the Gauss–Radau points. Similarly, with a particular choice of  $c$  as

$$c = \frac{\gamma_m m}{(m+1)(\alpha_m m!)^2} \Rightarrow \eta_m = \frac{m}{m+1},$$

we recover

$$g_L(x) = \frac{(-1)^m}{2}(1-x)\tilde{P}_m(x), \quad g_R(x) = \frac{1}{2}(1+x)\tilde{P}_m(x),$$

which is the spectral difference scheme discussed in the previous chapter and analyzed in [65].

Taking

$$c = \frac{\gamma_m(m+1)}{m(\alpha_m m!)^2} \Rightarrow \eta_m = \frac{m+1}{m}$$

leads to

$$g_L(x) = \frac{(-1)^m}{2} \left( \tilde{P}_m(x) - \frac{(m+1)\tilde{P}_{m-1}(x) + m\tilde{P}_{m+1}(x)}{2m+1} \right),$$

$$g_R(x) = \frac{1}{2} \left( \tilde{P}_m(x) + \frac{(m+1)\tilde{P}_{m-1}(x) + m\tilde{P}_{m+1}(x)}{2m+1} \right),$$

which results in a scheme proposed in [64]. Other choices are possible.

Let us finally consider schemes that explore the limits of  $c$ . Letting  $c$  approach its lower limit results in  $\eta_m \rightarrow -1$  in which case  $g_{L,R}$  becomes unbounded. This corresponds to imposing the flux correction strongly at the interface and is not likely to result in a practical scheme.

Letting  $c$  approach  $\infty$  yields

$$g_L(x) = \frac{(-1)^m}{2} \left( \tilde{P}_{m-1}(x) - \tilde{P}_m(x) \right), \quad g_R(x) = \frac{1}{2} \left( \tilde{P}_{m-1}(x) + \tilde{P}_m(x) \right),$$

which we recognize as the Gauss–Radau functions discussed above, albeit at one order lower.

A comparative study in [111] suggests that  $c = 0$  yields superior accuracy for the flux reconstruction scheme but a detailed understanding of this appears to be open. In particular, nonlinear stability is illusive due to aliasing errors in the pointwise evaluation of the nonlinear flux functions and the reliance on very smooth solutions in the above analysis. Some additional analysis can be found in [110, 66]. ■

The flux reconstruction method has been extended to multidimensional unstructured grids [17] and higher-order operators for diffusion [18, 119], and efficient software is available [120].

## 12.4 • Extension to multidimensional problems

So far, we have focused almost exclusively on the formulation and analysis of the discontinuous Galerkin schemes for one-dimensional problems. As we shall quickly realize, the generalization to multidimensional problems is, from a theoretical view point, straightforward.

Let us consider the multidimensional conservation law

$$\begin{aligned} \frac{\partial u(\mathbf{x}, t)}{\partial t} + \nabla \cdot \mathbf{f}(u(\mathbf{x}, t), \mathbf{x}, t) &= 0, \quad \mathbf{x} \in \Omega \in \mathbb{R}^d, \\ u(\mathbf{x}, t) &= g(\mathbf{x}, t), \quad \mathbf{x} \in \partial\Omega_i, \\ u(\mathbf{x}, 0) &= f(\mathbf{x}). \end{aligned} \quad (12.92)$$

For simplicity and with minimal loss of generality, we restrict attention to the scalar equation, as the issues related to the vector equations case are not specific to the multidimensional case, and have already been discussed at length. We assume that the correct number of boundary conditions is available at all inflow points of the boundary  $\partial\Omega_i$ , i.e., points where the eigenvalues of the flux-Jacobian  $\hat{\mathbf{n}} \cdot \nabla_u \mathbf{f}$  are negative along the outward pointing normal  $\hat{\mathbf{n}}$ .

To secure geometric flexibility and pave the way for the discontinuous Galerkin method, we assume that  $\Omega$  can be tiled using  $N$  nonoverlapping elements as

$$\Omega \simeq \Omega_b = \bigcup_{j=1}^N \mathbf{D}_j,$$

where  $\mathbf{D}_j$  can be any space-filling shape, e.g., triangles/quadrilaterals in two dimensions or tetrahedra/hexahedra in three dimensions. While these are the most commonly used elements, combinations of these or more general polygons/polyhedra are, likewise, possible. For simplicity, we assume that the elements are straight sided, although this too can be relaxed; see e.g., [56, 117]. Additionally, one does not need to require that the elements are conforming, further extending the generality of the scheme.

We now follow the one-dimensional approach, and assume that the solution can be approximated by

$$u(\mathbf{x}, t) \simeq u_b(\mathbf{x}, t) = \bigoplus_{j=1}^N u_j(\mathbf{x}, t) \in \mathbf{V}_b = \bigoplus_{j=1}^N \text{span} \left\{ \psi_i(\mathbf{D}_j) \right\}_{i=1}^{N_p},$$

where  $\psi_n(\mathbf{D}_j)$  is a  $d$ -dimensional polynomial basis defined on element  $\mathbf{D}_j$ . The local polynomial  $u_j(\mathbf{x}, t)$  can be expressed by

$$\mathbf{x} \in \mathbf{D}_j : u_j(\mathbf{x}, t) = \sum_{i=1}^{N_p} u_j(\mathbf{x}_{j,i}, t) \ell_{j,i}(\mathbf{x}),$$

where  $\ell_{j,i}(\mathbf{x})$  is the multidimensional Lagrange polynomial defined by some family of appropriate grid points,  $\mathbf{x}_{j,i} \in \mathbf{D}_j$ . In  $d$  dimensions, the cardinality of a basis of order  $m$  is

$$N_p = \binom{m+d}{m}.$$

We require the residual to be orthogonal to all test functions,  $\ell_{j,i} \in V_h$ , which yields the local statements

$$\forall i, j : \int_{D_j} \left[ \frac{\partial u_j}{\partial t} \ell_{j,i}(x) - f_j \cdot \nabla \ell_{j,i}(x) \right] dx = - \oint_{\partial D_j} \hat{n} \cdot F \ell_{j,i}(x) dx,$$

and, after applying Gauss' divergence theorem once again, we obtain

$$\int_{D_j} \left[ \frac{\partial u_j}{\partial t} + \nabla \cdot f_j \right] \ell_{j,i}(x) dx = \oint_{\partial D_j} \hat{n} \cdot [f_j - F] \ell_{j,i}(x) dx.$$

These are known, respectively, as the weak and strong form of the nodal discontinuous Galerkin method in the multidimensional case.

As a final step, we need to specify the numerical flux  $F$ . This is essentially a generalization of the one-dimensional case and one often resorts to the monotone local Lax–Friedrichs flux

$$F(a, b) = \frac{f(a) + f(b)}{2} + \frac{C}{2} \hat{n}(a - b),$$

where  $C$  is the local absolute maximum of the eigenvalues of the directional flux Jacobian

$$C = \max_{u \in [a, b]} |\lambda(\hat{n} \cdot \nabla_u f)|.$$

As for the one-dimensional case, the actual implementation of the scheme is somewhat complex and requires attention to a variety of details. We refer to [56, 69] for a comprehensive discussion of this aspect, exemplified by a variety of problems.

One crucial difference between the one-dimensional case and the multidimensional case is the attainable accuracy of the scheme. We recall that for the one-dimensional case, we can achieve an optimal convergence rate of  $\mathcal{O}(h^{m+1})$  for linear systems, provided the numerical flux is based on exact upwinding.

To understand the situation in the multidimensional case, let us consider the linear neutron transport equation

$$\alpha \cdot \nabla u + \beta u = f, \quad x \in \Omega \subset \mathbb{R}^2. \quad (12.93)$$

In [77], it was shown that if  $u$  is smooth and  $\Omega$  is tiled with rectangular elements, each using an  $m$ th order tensor basis and an upwind flux, the scheme exhibits optimal convergence:

$$\|u - u_h\|_h \leq C h^{m+1} \|u\|_{h,m+2}.$$

This is in line with the one-dimensional results discussed in subsection 12.1.3. However, in the case of general triangular grids, the analysis indicates a reduced  $\mathcal{O}(h^m)$ . This estimate can be improved to [68]

$$\|u - u_h\|_h \leq C h^{m+1/2} \|u\|_{h,m+1}, \quad (12.94)$$

provided the triangulation is quasi-uniform, i.e., the angles of all triangles are bounded from below by a constant, independent of the element size  $h$ . In [93] this result is

further improved to

$$\|u - u_b\|_b \leq C b^{m+1} \|u\|_{b,m+2},$$

for quasi-uniform grids for which all edges are bounded away from the characteristic direction, i.e.,  $|\alpha \cdot \hat{n}| > 0$  for all outward pointing normals on the element edges. In other words, this analysis does not apply to the case where edges are aligned with the characteristic direction.

To understand this special case, let us consider an example, taken from [87].

**Example 12.39.** Consider the neutron equation (12.93) with  $\alpha = (0, 1)$  and  $\beta = 0$  as

$$\frac{\partial u}{\partial y} = 0, \quad x \in [0, 1]^2,$$

and the boundary condition

$$u(x, 0) = x.$$

It is immediate that the exact solution is  $u(x, y) = x$ .

Assume that  $\Omega = [0, 1]^2$  is triangulated with a simple grid such that  $b_x = h = 1/I$  and  $b_y = h/2$ , where  $I$  is the number of cells along the  $x$ -axis. In other words, the horizontal edge length is twice that of the vertical edge length. Otherwise the grid is entirely regular.

We employ the simplest possible formulation with an  $m = 0$  basis. Hence, the finite volume scheme is

$$\oint_{\partial D_j} \hat{n}_y u^* dx = 0.$$

If we define the grid function

$$u_{i,j} = u_b(i b_x, j b_y) \text{ for } \begin{cases} i = 1/2, 3/2, 5/2, \dots, I-1/2, & j = 0, 2, 4, \dots, J, \\ i = 0, 1, 2, \dots, I, & j = 1, 3, 5, \dots, J-1, \end{cases}$$

use upwinding, and exploit the geometry of the grid, the local scheme becomes

$$u_{i,j+1} = \frac{1}{2} [u_{i-1/2,j} + u_{i+1/2,j}].$$

If we further assume the symmetry  $u_{i,j} = u_{-i,j}$ , this also extends to the boundary as

$$u_{0,j+1} = u_{1/2,j}.$$

By induction, the exact numerical solution can be obtained exactly as

$$u_{i,j} = \frac{1}{2^j} \sum_{q=0}^j \binom{j}{q} u_{j/2-q+i,0} = \frac{1}{2^j} \sum_{q=0}^j \binom{j}{q} |b(j/2 - q + i)|, \quad i < I - \frac{j}{2},$$

since  $u(x, 0) = x$ . The restriction on the  $i$  index is to avoid complications caused by the finite domain.

Along  $x = 0$  and  $x = 1$ , the grid is aligned with the characteristic direction. Let us therefore consider the solution at  $(0, 1 - b_y)$  as ( $J' = J - 1$ ):

$$u_{0,J'} = \frac{1}{2^{J'}} \sum_{q=0}^{J'} \binom{J'}{q} |b(j/2 - q)| = \sum_{j=1,3,5,\dots}^{J'} \frac{b}{2^j} \binom{j-1}{(j-1)/2},$$

where we refer to [87] for the proof by induction of the last reduction. For an even integer, using Stirling's formula, we recover that

$$\binom{a}{a/2} = \frac{a!}{(a/2)!(a/2)!} \geq c 2^{a+1} a^{-1/2},$$

where  $c$  is a positive constant independent of  $a$ . As  $J' \propto h^{-1}$ , we thus obtain

$$u_{0,J'} \geq c b \sum_{j=1,3,5,\dots}^{J'} \frac{1}{\sqrt{j-1}} \geq c b \int_1^{J'/2} \frac{1}{\sqrt{2s}} ds \geq c b \sqrt{J'} \geq c b^{1/2}.$$

Since the exact solution is  $u(0, y) = 0$ , this result reflects the pointwise error and suggests a loss of optimal convergence for certain special grids. ■

As the example highlights, if  $\hat{n} \cdot \alpha = 0$  there may be a loss of the optimal rate of convergence. In [87], this was used to construct an example that demonstrates the loss of optimality in  $L^2$  for higher order and confirms that (12.94) is sharp. It should be emphasized that this suboptimal behavior is observed only for very specific grids and one can generally expect optimal rates of convergence when solving problems with smooth solutions on general unstructured grids. For linear problems one can prevent this special case from happening by constructing the grid appropriately.

These results can be expected to carry over to nonlinear problems with smooth solutions as long as a monotone flux is used. In the multidimensional case, this is proven in [123, 124], under the assumption that upwind fluxes are used.

For the general nonlinear case with nonsmooth solutions, the main complication lies in the incompatibility between the solution being high-order accurate and TVD at the same time. Therefore, the best one can hope for is that the solutions are TVB. The development of such results, including both the multidimensional scalar case and the system case, are presented in [25, 30].

## 12.5 • Discontinuous Galerkin methods in action

We now begin our discussion of the implementation of discontinuous Galerkin methods for one-dimensional problems. These methods utilize the operators and local nodal points discussed throughout this chapter. For boundary conditions, we need to enable the extension of the computational domain. However, since we are now duplicating interface points, the procedure, although based on the same principles, is slightly different from the approach introduced in subsection 5.3.1. This is outlined in `extendDG.m`.

**Script 12.19. `extendDG.m`: Impose boundary conditions in the discontinuous Galerkin method through extension.**

---

```
function [ue] = extendDG(u,BCl,ul,BCr,ur)
% Purpose: Extend dependent and independent vectors u with m+1 entries
% subject to appropriate boundary conditions for DG formulation
% BC = "D" - Dirichlet; BC = "N" - Neumann; BC = "P" - periodic
% u - BC value - only active for Dirichlet BC
```

```

dim = size(u); m = dim(1)-1; N = dim(2);
ue = zeros(m+1,N+2); ue(:,2:N+1) = u;

% Periodic extension of u
if (BCl=='P') | (BCr=='P')
    ue(:,1) = u(:,N); ue(:,N+2) = u(:,1);
    return;
end;

% Left extension
if BCl=='D'
    ue(:,1) = -flipud(u(:,1))+2*u1;
else
    ue(:,1) = flipud(u(:,1));
end

% Right extension
if BCr=='D'
    ue(:,N+2) = -flipud(u(:,N))+2*ur;
else
    ue(:,N+2) = flipud(u(:,N));
end
return

```

---

Another issue to consider is how to choose a time step  $k$  that ensures discrete stability. So far, we have relied of the CFL condition in the form

$$k \leq Ch,$$

where  $C$  mainly depends on the fastest characteristic speed in the system and, to a lesser extent, on the order of the scheme. For the discontinuous Galerkin method, the richer polynomial structure on each element changes this. The exact relationship between the local order of approximation and the constant in the CFL condition is not known. However, experience and some heuristics, see [44, 56], suggest that a robust choice is

$$k \leq Ch_{min}, \quad h_{min} = h \min_i \Delta^+ r_i,$$

where  $r_i$  are the Gauss–Lobatto points used in each element. In other words, rather than being defined by the cell size, the CFL number depends on the minimal distance between any two nodes in the cell.

### 12.5.1 • Linear wave equation

As previously, let us begin by solving the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0,$$

subject to appropriate initial and boundary conditions. The grid, the initial conditions, and the general description of the computational model are set in LinwaveDGDriver1D.m.

---

**Script 12.20.** *LinwaveDGDriver1D.m: Problem setup and generation of the grid for solving the one-dimensional linear wave equation using a discontinuous Galerkin method.*

---

```
% Driver script for solving the 1D wave equations using an DG scheme
clear all

% Order of method (m), number of elements (N)
m=1; N=128;

% Set problem parameters
FinalTime = 4.0; CFL = 0.20;
xmin = -1; xmax = 1;

% Generate mesh
VX = (xmax-xmin)*(0:N)/N + xmin; r = LegendreGL(m);
x = ones(m+1,1)*VX(1:N) + 0.5*(r+1)*(VX(2:N+1)-VX(1:N));
h = (xmax-xmin)/N;

% Define initial conditions
uh = wavetest(x(:), 0.5, -0.7, 0.005, 10, log(2)/(36*0.005^2));
u = reshape(uh,m+1,N);

% Solve Problem
[u] = LinwaveDG1D(x,u,h,m,N,CFL,FinalTime);
```

---

In LinwaveDG1D.m, a third order SSPERK scheme is implemented for temporal integration, relying on the evaluation of the right-hand side of the linear wave problem, discretized by a discontinuous Galerkin method. This is also where limiting is applied, if needed.

---

**Script 12.21.** *LinwaveDG1D.m: Time integration of the one-dimensional linear wave equation with the discontinuous Galerkin method.*

---

```
function [u] = LinwaveDG1D(x,u,h,m,N,CFL,FinalTime)
% function [u] = LinwaveDG1D(x,u,h,m,N,CFL,FinalTime)
% Purpose : Integrate 1D linear wave equation until FinalTime using a DG
%            scheme and 3rd order SSP-RK method
% Initialize operators at Legendre Gauss Lobatto grid
r = LegendreGL(m); V = VandermondeDG(m, r); D = DmatrixDG(m, r, V);
Ma = inv(V*V'); S = Ma*D; iV = inv(V);

% Compute operator for WENO smoothness evaluator
[Q,Xm,Xp] = WENODGWeights(m,iV);

% Initialize extraction vector
VtoE = zeros(2,N);
for j=1:N
    VtoE(1,j) = (j-1)*(m+1)+1; VtoE(2,j) = j*(m+1);
end

% Compute smallest spatial scale timestep
rLGLmin = abs(r(1)-r(2));
time = 0; tstep = 0;

% Set timestep
k = CFL*rLGLmin*h;
```

---

```
% integrate scheme
while (time<FinalTime)
    if (time+k>FinalTime) k = FinalTime-time; end
    % Update solution
    rhsu = LinwaveDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,1.0);
    u1 = u + k*rhsu;
    u1 = WENOlimitDG(x,u1,m,h,N,V,iV,Q,Xm,Xp);

    rhsu = LinwaveDGrhs1D(x,u1,h,k,m,N,Ma,S,VtoE,1.0);
    u2 = (3*u + u1 + k*rhsu)/4;
    u2 = WENOlimitDG(x,u2,m,h,N,V,iV,Q,Xm,Xp);

    rhsu = LinwaveDGrhs1D(x,u2,h,k,m,N,Ma,S,VtoE,1.0);
    u = (u + 2*u2 + 2*k*rhsu)/3;
    u = WENOlimitDG(x,u,m,h,N,V,iV,Q,Xm,Xp);
    time = time+k; tstep = tstep+1;
end
return
```

---

The spatial approximation of the right-hand side, utilizing a discontinuous Galerkin formulation and expressed in weak form, is illustrated in `LinwaveDGrhs1D.m`. In this case, the global Lax–Friedrichs flux is used.

---

**Script 12.22. *LinwaveDGrhs1D.m: Evaluation of the right-hand side for solving the one-dimensional linear wave equation using a discontinuous Galerkin method.***

---

```
function [rhsu] = LinwaveDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,maxvel)
% function [rhsu] = LinwaveDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,maxvel)
% Purpose : Evaluate the RHS of the linear wave equations using a DG method
Imat = eye(m+1); ue = zeros(2,N+2);

% Extend data and assign boundary conditions
[ue] = extendDG(u(VtoE),'P',0,'P',0);

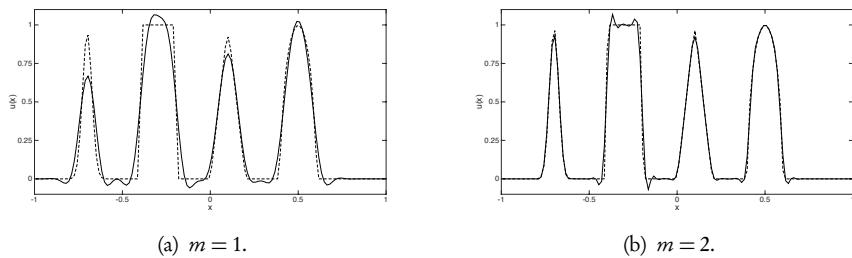
% Compute numerical fluxes at interfaces
fluxr = LinwaveLF(ue(2,2:N+1),ue(1,3:N+2),0,maxvel);
fluxl = LinwaveLF(ue(2,1:N),ue(1,2:N+1),0,maxvel);

% Compute right hand side of Maxwell's equation
ru = S'*u - (Imat(:,m+1)*fluxr(1,:)-Imat(:,1)*fluxl(1,:));
rhsu = (h/2*Ma)\ru;
return
```

---

We begin by considering the test case discussed in subsection 1.4.1. Figure 12.19 shows the computed solution for different orders of approximation. While the overall benefits of the high-order approximation on the propagation are clear, the oscillations created by the insufficient smoothness are likewise clear.

We explore the use of limiters, discussed in subsections 12.2.4–12.2.5, to eliminate the artificial oscillations. Figure 12.20 shows the results of a direct comparison between four different limiters. In all cases, we employ  $N = 128$  and orders of approximation  $m = 1, 2$  as in Fig. 12.19. A minmod function is used as a troubled-cell indicator. As a slope limiter, a MUSCL limiter is used.



**Figure 12.19.** Solution of linear wave test case using a discontinuous Galerkin method with a Lax-Friedrichs flux. We show the computed solution (solid line) along with the exact solution (dashed line). In all cases,  $N = 128$  and the result is shown at  $T = 4.0$ .

While the elimination of the artificial oscillations and the results generally improve the accuracy as the order of approximation increases, we also observe substantial differences in the performance of the different limiters. The standard TVD-limiter is the most dissipative, while the improved limiter [11] performs better at the extrema; the moment-based limiter [7] performs very well in this case, whereas the WENO-based limiter appears to be only marginally better than the simpler MUSCL-based limiter. The most pronounced improvement is observed at the smooth extrema.

To improve on the performance of the WENO limiter, let us consider the use of a different troubled-cell indicator. As an example, we use the TVB-based indicator, discussed in section 10.2. Figure 12.21 shows the results for orders  $m = 2, 3$ , and different choices of the parameter  $M$  in the TVB indicator. A comparison of Fig. 12.20 and Fig. 12.21 highlights the substantial improvements made possible by carefully selecting the troubled-cell indicator. The consistent behavior when compared to alternative limiters in Fig. 12.20 is encouraging. Naturally, one can also consider the use of improved troubled-cell indicators with other limiters to observe further improvements. A direct comparison of the impact of the choice of the troubled-cell indicator in the context of WENO-limited discontinuous Galerkin methods can be found in [89].

Finally, let us also consider the impact of the extrema preserving limiter on this more complicated, albeit linear, test case. Figure 12.22 shows the results for increasing orders of approximation. In agreement with the analysis, we observe excellent performance. When comparing with the results in Fig. 12.19, we clearly observe the elimination of the artificial oscillations, without any visual impact on the accuracy of the scheme.

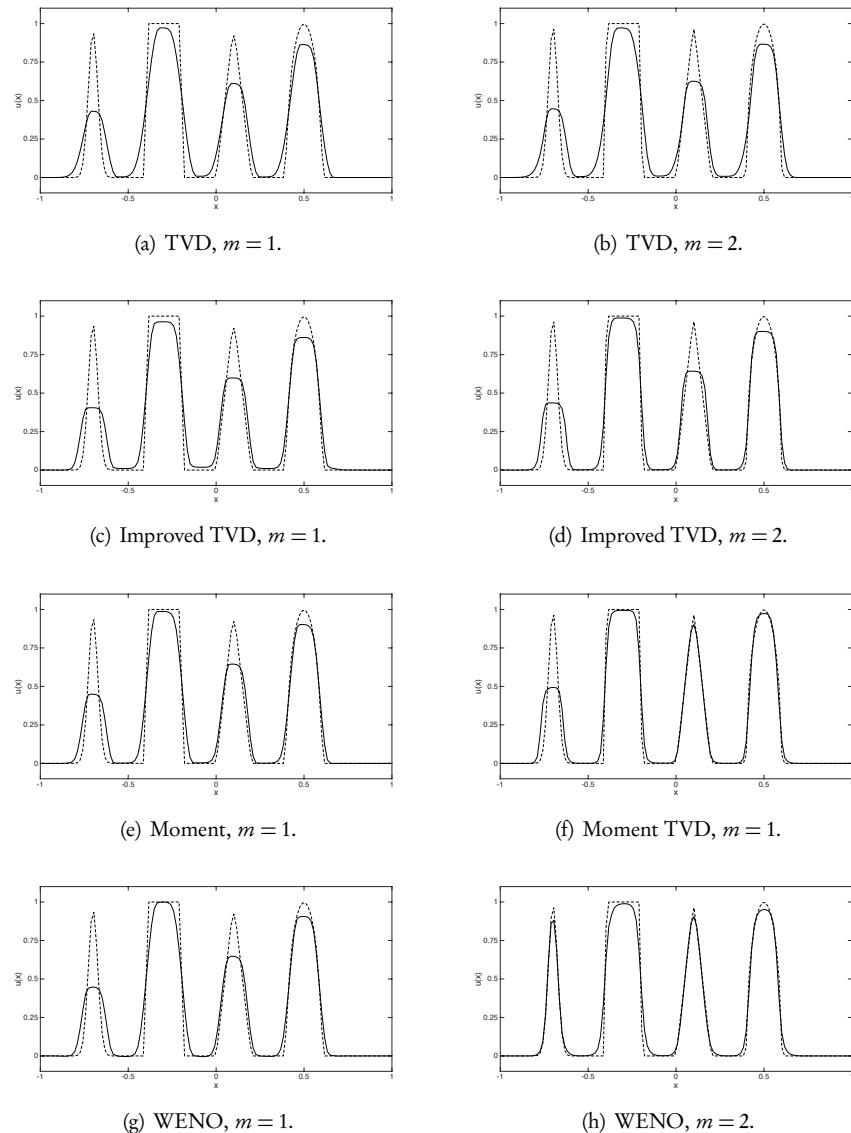
### 12.5.2 • Burgers' equation

With excellent results for the linear wave equation, let us consider Burgers' equation

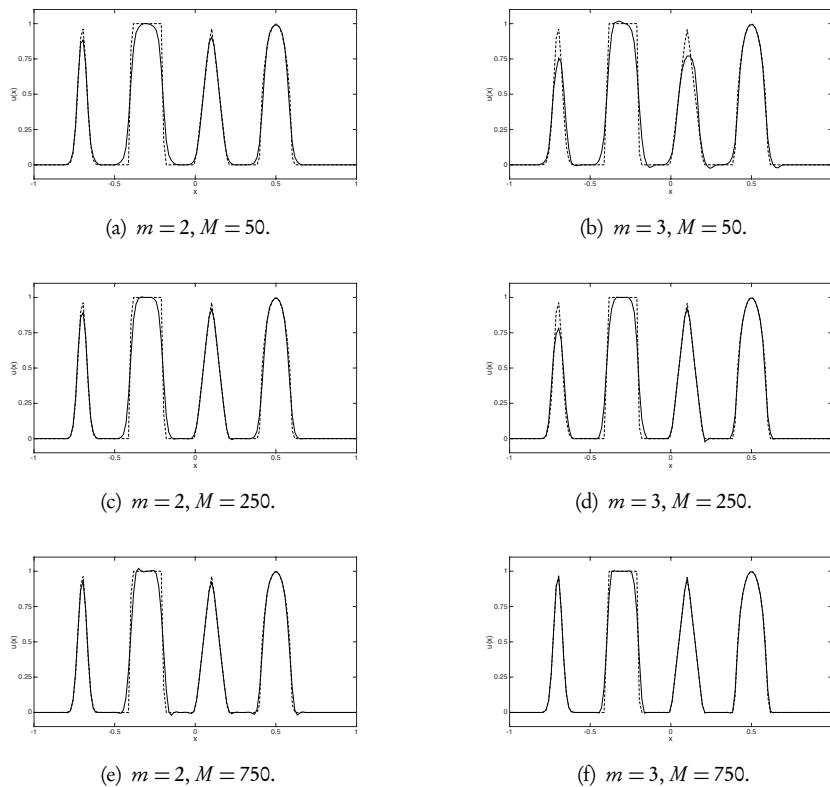
$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to appropriate boundary and initial conditions.

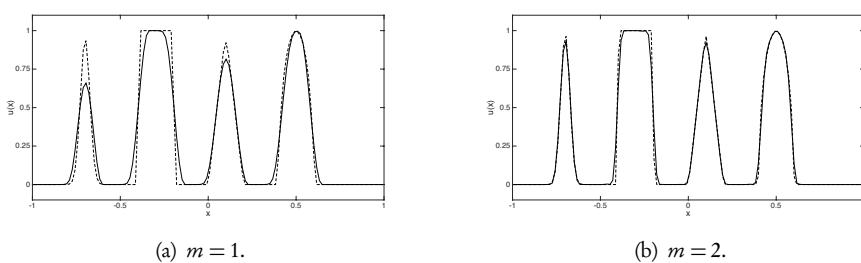
In `BurgersDGDriver1D.m`, we illustrate the setup of the grid and the initial conditions for solving Burgers' equation using a discontinuous Galerkin method.



**Figure 12.20.** The impact of limiting on a solution of the linear wave equation, solved using a limited discontinuous Galerkin method of order  $m$ . We consider a TVD limiter [27, 26], an improved TVD-limiter [11], a moment-based limiter [7], and a WENO-based limiter [128]. In all cases,  $N = 128$  and a minmod function is used as the indicator function. A MUSCL limiter is used in the reconstruction except for the WENO-based limiter where the smoothness indicators are used for the reconstruction.



**Figure 12.21.** The impact of changing the troubled-cell indicator on a solution (solid line), advected by the linear wave equation, along with the exact solution (dashed line). In all cases, we used  $N = 128$ , a TVB-based indicator and a WENO limiter. The results reflect variations when changing order of approximation  $m$  and  $M$  in the TVB minmod limiter.



**Figure 12.22.** Solution of the linear wave test case using an  $m$ th order discontinuous Galerkin method with an extrema preserving limiter. The computed solution (solid line) along with the exact solution (dashed line) are shown. In all cases,  $N = 128$  and the result is shown at  $T = 4.0$ .

---

**Script 12.23. BurgersDGDriver1D.m: Problem setup and grid generation for solving the one-dimensional Burgers' equation using a discontinuous Galerkin method.**

---

```
% Driver script for solving the 1D Burgers equations using an DG scheme
clear all

% Order of method (m), number of elements (N)
m=1; N=40;

% Set problem parameters
xmin = 0.0; xmax = 1.0;
FinalTime = 0.5; CFL = 0.1;

% Generate mesh
VX = (xmax-xmin)*(0:N)/N + xmin; r = LegendreGL(m);
x = ones(m+1,1)*VX(1:N) + 0.5*(r+1)*(VX(2:N+1)-VX(1:N));
h = (xmax-xmin)/N;

% Define initial conditions
u = sin(2*pi*x)+0.8; %periodic BC needed
% u = (1-sign(x-0.2))/2+1; % Constant BC needed

% Solve Problem
[u] = BurgersDG1D(x,u,h,m,N,CFL,FinalTime);
```

---

The equation is integrated in time using a third order SSPERK scheme, as illustrated in BurgersDG1D.m, where filtering or limiting is likewise applied.

---

**Script 12.24. BurgersDG1D.m: Time integration for Burgers' equation using the discontinuous Galerkin method.**

---

```
function [u] = BurgersDG1D(x,u,h,m,N,CFL,FinalTime)
% function [u] = BurgersDG1D(x,u,h,m,N,CFL,FinalTime)
% Purpose : Integrate 1D Burgers equation until FinalTime using a DG
% scheme and 3rd order SSP-RK method
% Initialize operators at Legendre Gauss Lobatto grid
r = LegendreGL(m); V = VandermondeDG(m, r); D = DmatrixDG(m, r, V);
Ma = inv(V*V'); S = Ma*D; iV = inv(V);

% Compute operator for WENO smoothness evaluator
[Q,Xm,Xp] = WENODGWeights(m, iV);

% Initialize extraction vector
VtoE = zeros(2,N);
for j=1:N
    VtoE(1,j) = (j-1)*(m+1)+1; VtoE(2,j) = j*(m+1);
end

% Initialize filter matrix
% F = FilterDG(m,0,10,V);

% Compute smallest spatial scale timestep
rLGLmin = min(abs(r(1)-r(2)));
time = 0; tstep = 0;

% Initialize parameters for nonlinear viscosity
nu = zeros(m+1,N); nu0 = 2; kappa = -6; c2 = 1;

% integrate scheme
```

```

while (time<FinalTime)
  % Decide on timestep
  maxvel = 2*max(max(abs(u))); k = CFL*rLGLmin*h/maxvel;
  if (time+k>FinalTime) k = FinalTime-time; end
  % Update solution - stage 1
  rhsu = BurgersDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,maxvel);
  u1 = u + k*rhsu;
  u1 = WENOlimitDG(x,u1,m,h,N,V,iV,Xm,Xp);
  % Update solution - stage 2
  rhsu = BurgersDGrhs1D(x,u1,h,k,m,N,Ma,S,VtoE,maxvel);
  u2 = (3*u + u1 + k*rhsu)/4;
  u2 = WENOlimitDG(x,u2,m,h,N,V,iV,Xm,Xp);
  % Update solution - stage 3
  rhsu = BurgersDGrhs1D(x,u2,h,k,m,N,Ma,S,VtoE,maxvel);
  u = (u + 2*u2 + 2*k*rhsu)/3;
  u = WENOlimitDG(x,u,m,h,N,V,iV,Q,Xm,Xp);
  plot(x,u);
  pause(0.1);
  time = time+k; tstep = tstep+1;
end
return

```

---

Finally, we illustrate in *BurgersDGrhs1D.m* the evaluation of the right-hand side based on the discontinuous Galerkin method for Burgers' equation, using a Lax-Friedrichs numerical flux.

**Script 12.25. *BurgersDGrhs1D.m*: Evaluation of the right-hand side for solving the one-dimensional Burgers' equation using a discontinuous Galerkin method.**

---

```

function [rhsu] = BurgersDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,maxvel)
% function [rhsu] = BurgersDGrhs1D(x,u,h,k,m,N,Ma,S,VtoE,maxvel)
% Purpose : Evaluate the RHS of Burgers equations using a DG method
Imat = eye(m+1); ue = zeros(N+2,2);

% Extend data and assign boundary conditions
[ue] = extendDG(u(VtoE), 'P', 0, 'P', 0);

% Compute numerical fluxes at interfaces
fluxr = BurgersLF(ue(2,2:N+1), ue(1,3:N+2), 0, maxvel);
fluxl = BurgersLF(ue(2,1:N), ue(1,2:N+1), 0, maxvel);

% Compute right hand side of Maxwell's equation
ru = S*(u.^2) - (Imat(:,m+1)*fluxr(1,:)) - (Imat(:,1)*fluxl(1,:));
rhsu = (h/2*Ma)\ru;
return

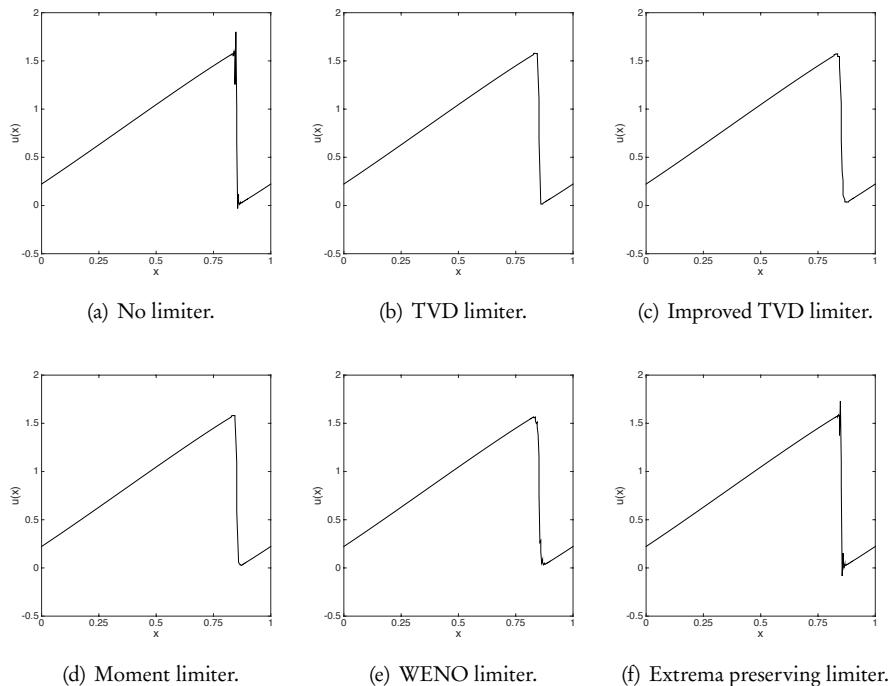
```

---

We demonstrate the performance of the discontinuous Galerkin method for the case of a smooth initial condition, steepening into a moving shock. The initial condition is

$$u(x,0) = \sin(2\pi x) + 0.8.$$

Figure 12.23 shows the result at  $T = 0.22$ , obtained for  $N = 128$  elements and order of approximation  $m = 2$  in combination with various limiters. In the absence of limiting we observe the expected oscillations. At the same time we also find that the different limiters perform more or less equivalently for this simple test case. The extrema preserving limiter does not eliminate the oscillations but simply guarantees



**Figure 12.23.** Solution of Burgers' equation using a discontinuous Galerkin method with different limiters. The full polynomial solution in each cell is plotted. In the top row we show the unlimited solution (left), the solution when a TVD limiter [27, 26] is used (middle), and the solution obtained with the improved limiter [11]. In the bottom row, we show the result obtained with a moment limiter (left) [7], a WENO-based limiter (middle) [128], and an extrema preserving limiter (right) [125]. In all cases  $m = 2$ ,  $N = 128$ , and we use a TVD-based troubled-cell indicator. The solution is shown at  $T = 0.22$ .

that extrema are not exceeded. Hence, if shocks appear within the defined bounds, the oscillations will remain, as highlighted in this example. In such cases, additional slope limiting is required to control the oscillations.

### 12.5.3 • Maxwell's equations

Before moving on to the Euler equations, let us briefly consider the one-dimensional Maxwell's equations [118]

$$\varepsilon(x) \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x}, \quad x \in [-1, 1],$$

where the coefficient  $\varepsilon$  is assumed to be piecewise-constant, as discussed in detail in subsection 1.4.1. We assume the boundary condition is  $E(-1, t) = E(1, t) = 0$ , corresponding to a closed metallic cavity, partially filled with a material, specified by  $\varepsilon$ . The exact solution is given in subsection 1.4.1. While the solution itself is continuous,

the discontinuous material parameter implies that the E-field loses smoothness and its derivative is discontinuous at the material interface. As we found in subsection 5.3.4, this may result in a reduced order of convergence.

The driver routine for the discontinuous Galerkin method to solve the one-dimensional Maxwell's equations is illustrated in `MaxwellDGDriver1D.m`. It initializes the grid, the initial conditions, and various other parameters for the problem.

**Script 12.26.** *MaxwellDGDriver1D.m: Problem setup and generation of the grid for solving the one-dimensional Maxwell's equations using a discontinuous Galerkin method.*

---

```
% Driver script for solving the 1D Maxwell's equations using DG scheme
clear all

% Order of method (m), number of elements (N)
m=4; N=10;

% Set problem parameters
xmin = -1.0; xmax = 1.0;
FinalTime = sqrt(2.0); CFL = 0.25;
ep1 = 1.0; mul = 1.0; epr = 2.0; mur = 1.0;

% Generate mesh
VX = (xmax-xmin)*(0:N)/N + xmin; r = LegendreGL(m);
x = ones(m+1,1)*VX(1:N) + 0.5*(r+1)*(VX(2:N+1)-VX(1:N));
h = (xmax-xmin)/N;

% Define domain, materials and initial conditions
Ef = zeros(m+1,N); Hf = zeros(m+1,N);
for k = 1:N
    [Ef(:,k), Hf(:,k), ep(:,k), mu(:,k)] = ...
        CavityExact(x(:,k), ep1, epr, mul, mur, 0);
end

% Set up material parameters
eps1 = [ep1*ones(1,N/2), epr*ones(1,N/2)];
mu1 = [mul*ones(1,N/2), mur*ones(1,N/2)];
ep = ones(m+1,1)*eps1; mu = ones(m+1,1)*mu1;

% Solve Problem
q = [Ef Hf];
[q] = MaxwellDG1D(x,q,ep, mu, h, m, N, CFL, FinalTime);
```

---

In `MaxwellDG1D.m`, the system is integrated in time using a third order SSPERK scheme. This relies on the evaluation of the right-hand side, discretized using the discontinuous Galerkin method, as illustrated in `MaxwellDGrhs1D.m`.

**Script 12.27.** *MaxwellDG1D.m: Time integration routine for Maxwell's equations using the discontinuous Galerkin method.*

---

```
function [q] = MaxwellDG1D(x,q,ep, mu, h, m, N, CFL, FinalTime)
% function [q] = MaxwellDG1D(x,q,ep, mu, h, m, N, CFL, FinalTime)
% Purpose : Integrate 1D Maxwell's equation until FinalTime using a DG
%             scheme and a 3rd order SSP-RK method.
% Initialize operators at Legendre Gauss Lobatto grid
r = LegendreGL(m); V = VandermondeDG(m, r); D = DmatrixDG(m, r, V);
Ma = inv(V'*V'); S = Ma*D;
```

---

```

% Initialize extraction vector
VtoE = zeros(2,N);
for j=1:N
    VtoE(1,j) = (j-1)*(m+1)+1; VtoE(2,j) = j*(m+1);
end

% Compute smallest spatial scale timestep
rLGLmin = min(abs(r(1)-r(2)));
time = 0; tstep = 0;

% Set timestep
cvel = 1./sqrt(ep.*mu); maxvel = max(max(cvel));
k = CFL*rLGLmin*h/2/maxvel;

% integrate scheme
while (time<FinalTime)
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    [rhsq] = MaxwellDGrhs1D(x,q,ep,mu,h,k,m,N,Ma,S,VtoE,maxvel);
    q1 = q + k*rhsq;
    [rhsq] = MaxwellDGrhs1D(x,q1,ep,mu,h,k,m,N,Ma,S,VtoE,maxvel);
    q2 = (3*q + q1 + k*rhsq)/4;
    [rhsq] = MaxwellDGrhs1D(x,q2,ep,mu,h,k,m,N,Ma,S,VtoE,maxvel);
    q = (q + 2*q2 + 2*k*rhsq)/3;
    time = time+k; tstep = tstep+1;
end
return

```

---

**Script 12.28. MaxwellDGrhs1D.m: Evaluation of the right-hand side for solving the one-dimensional Maxwell's equations using a discontinuous Galerkin method.**

---

```

function [rhsq] = MaxwellDGrhs1D(x,q,ep,mu,h,k,m,N,Ma,S,VtoE,maxvel);
% function [dq] = MaxwellDGrhs1D(x,q,ep,mu,h,k,m,Ma,Sr,VtoE,maxvel);
% Purpose: Evaluate right hand side for Maxwell's equation using DG method
Imat = eye(m+1); Ee = zeros(2,N+2); He = zeros(2,N+2);
EMl = zeros(N,2); EMr = zeros(N,2); EMm = zeros(N,2); EMp = zeros(N,2);

% Impose boundary conditions
[Ee] = extendDG(q(VtoE,1), 'D', 0, 'D', 0);
[He] = extendDG(q(VtoE,2), 'N', 0, 'N', 0);

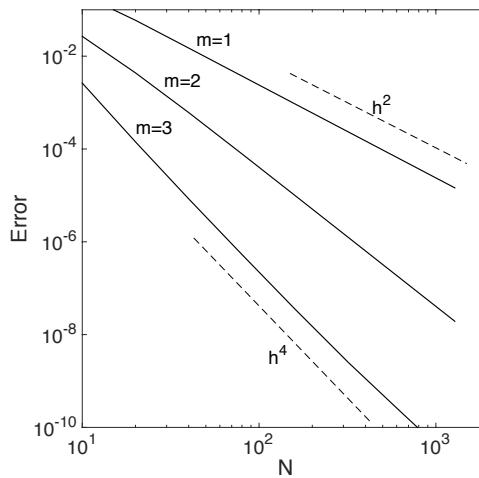
% Compute numerical fluxes at interfaces
EMr = [Ee(2,2:N+1)' He(2,2:N+1)']; EMl = [Ee(1,2:N+1)' He(1,2:N+1)'];
EMm = [Ee(2,1:N)' He(2,1:N)']; EMp = [Ee(1,3:N+2)' He(1,3:N+2)'];
fluxr = MaxwellLF(EMr,EMp,ep(1,:)',mu(1,:)',k/h,maxvel)';
fluxl = MaxwellLF(EMm,EMl,ep(1,:)',mu(1,:)',k/h,maxvel)';

% Compute right hand side of Maxwell's equation
rE = S'*H./ep - (Imat(:,m+1)*fluxr(1,:)) - Imat(:,1)*fluxl(1,:));
rH = S'*E./mu - (Imat(:,m+1)*fluxr(2,:)) - Imat(:,1)*fluxl(2,:));
rhsq = (h/2*Ma)\[rE rH];
return

```

---

It is clear that if we design the grid such that the material interface is within an element, the limited regularity of the solution will impact the achievable accuracy. Based on the theoretical developments in subsection 12.1.1, we expect  $\mathcal{O}(h^{3/2})$  in such a case, in agreement with the computational results in subsection 5.3.4.



**Figure 12.24.**  $L^2$ -error of the electric field in a cavity with a discontinuous material interface at  $T = \sqrt{2}$ . The result is obtained with a discontinuous Galerkin method of order  $m$  and with  $N$  elements, aligned with the material interface.

However, the discontinuous nature of the discontinuous Galerkin method suggests that we can eliminate the impact of the limited regularity if we ensure that the material interface coincides with a cell interface. In such a case, the solution on either side of the interface is smooth and we expect to recover an optimal order of convergence. This is confirmed by the results in Fig. 12.24, where  $\mathcal{O}(h^{m+1})$  order of convergence is observed. Additional discussions on the importance of the grid being geometry-conforming to ensure full order of accuracy can be found in [56].

#### 12.5.4 • The Euler equations

As a final example, let us consider the performance of the discontinuous Galerkin method for the more challenging Euler equations, discussed in detail in subsection 1.4.1 and given in conservation form as

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f} = 0,$$

where the conserved variables  $\mathbf{q}$  and the flux  $\mathbf{f}$  are defined as

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho \mathbf{u} \\ E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho \mathbf{u} \\ \rho \mathbf{u}^2 + p \\ (\mathbf{E} + p) \mathbf{u} \end{bmatrix}.$$

The system of equations is closed by the ideal gas law as

$$p = (\gamma - 1) \left( E - \frac{1}{2} \rho \mathbf{u}^2 \right), \quad c = \sqrt{\frac{\gamma p}{\rho}},$$

where  $c$  represents the local speed of sound, and  $\gamma$  is a fluid-dependent constant which we take to be  $\gamma = 7/5$ , as is typical for atmospheric gases.

In EulerDGDriver1D.m we initialize the grid and the initial conditions for the problem.

**Script 12.29.** *EulerDGDriver1D.m: Routine for setting up the problem and generating the grid for solving the one-dimensional Euler equations using a discontinuous Galerkin method.*

---

```
% Driver script for solving the 1D wave equations using an DG scheme
clear all

% Order of method (m), number of elements (N)
m=1; N=256;

% Set problem parameters
FinalTime = 1.8; CFL = 0.1; gamma = 1.4;

% Define domain, materials and initial conditions
r = zeros(m+1,N); ru = zeros(m+1,N); E = zeros(m+1,N);

% Define spatial grid
xmin = -5; xmax = 5; L = xmax - xmin;

% Generate mesh
VX = (xmax-xmin)*(0:N)/N + xmin; rv = LegendreGL(m);
x = ones(m+1,1)*VX(1:N) + 0.5*(rv+1)*(VX(2:N+1)-VX(1:N));
h = (xmax-xmin)/N;

% Initialize for Sod's problem
% r = (x<0.5) + (x>=0.5)*0.125;
% E = ((x<0.5) + (x>=0.5)*0.1)/(gamma-1);

% Initialize for shock entropy problem
r = (x<-4)*3.857143 + (x>=-4).*(1+0.2*sin(pi*x));
ru = (x<-4)*3.857143*2.629369;
p = (x<-4)*10.33333 + (x>=-4);
E = p/(gamma-1) + 0.5*ru.^2./r;

q = zeros(m+1,N,3); q(:,:,1)=r; q(:,:,2)=ru; q(:,:,3)=E;
% Solve Problem
[q] = EulerDG1D(x,q,h,m,N,CFL,gamma,FinalTime);
```

---

Temporal integration of the system is done in EulerDG1D.m with a third order SSPERK scheme and with limiting/nonlinear viscosity added to control oscillations. In case a limiter is used, we apply this on the characteristic variables.

**Script 12.30.** *EulerDG1D.m: Time integration of the Euler equations for the discontinuous Galerkin method. Limiting is done at the end of each stage of the SSPERK scheme.*

---

```
function [q] = EulerDG1D(x,q,h,m,N,CFL,gamma,FinalTime)
% function [q] = EulerDG1D(x,q,h,m,N,CFL,gamma,FinalTime)
% Purpose : Integrate 1D Euler equation until FinalTime using a DG
%             scheme and 3rd order SSP-RK method
% Initialize operators at Legendre Gauss Lobatto grid
r = LegendreGL(m); V = VandermondeDG(m, r); D = DmatrixDG(m, r, V);
```

```

Ma = inv (V*V') ; S = Ma*D; iV = inv (V) ;

% Compute operator for WENO smoothness evaluator
[qW,Xm,Xp] = WENODGWeights(m,iV) ;

% Initialize extraction vector
VtoE = zeros(2,N) ;
for j=1:N
    VtoE(1,j) = (j-1)*(m+1)+1; VtoE(2,j) = j*(m+1) ;
end

% Compute smallest spatial scale timestep
rLGLmin = abs(r(1)-r(2)) ;
time = 0; tstep = 0;

% integrate scheme
while (time<FinalTime)
    % Set timestep
    p = (gamma-1)*(q(:,:,3) - 0.5*q(:,:,2).^2./q(:,:,1)) ;
    c = sqrt(gamma*p./q(:,:,1)) ;
    maxvel = max(max(c+abs(q(:,:,2)./q(:,:,1)))); k = CFL*h*rLGLmin/maxvel ;
    if (time+k>FinalTime) k = FinalTime-time; end

    % Stage 1 of SSPRK
    rhsq = EulerDGrhs1D(x,q,h,k,m,N, gamma, S, Ma, VtoE, maxvel) ;
    q1 = q + k*rhsq ;

    % Limit solution through characteristics
    [qc,R] = EulerQtoRDG(q1, gamma, V, iV) ;
    R(:,:,1) = SlopeLimitCSDG(x,R(:,:,1),m,h,N,V,iV) ;
    R(:,:,2) = SlopeLimitCSDG(x,R(:,:,2),m,h,N,V,iV) ;
    R(:,:,3) = SlopeLimitCSDG(x,R(:,:,3),m,h,N,V,iV) ;
    q1 = EulerRtoQDG(R, qc, gamma, V, iV) ;

    % Stage 2 of SSPRK
    rhsq = EulerDGrhs1D(x,q1,h,k,m,N, gamma, S, Ma, VtoE, maxvel) ;
    q2 = (3*q + q1 + k*rhsq)/4;

    % Limit solution through characteristics
    [qc,R] = EulerQtoRDG(q2, gamma, V, iV) ;
    R(:,:,1) = SlopeLimitCSDG(x,R(:,:,1),m,h,N,V,iV) ;
    R(:,:,2) = SlopeLimitCSDG(x,R(:,:,2),m,h,N,V,iV) ;
    R(:,:,3) = SlopeLimitCSDG(x,R(:,:,3),m,h,N,V,iV) ;
    q2 = EulerRtoQDG(R, qc, gamma, V, iV) ;

    % Stage 3 of SSPRK
    rhsq = EulerDGrhs1D(x,q2,h,k,m,N, gamma, S, Ma, VtoE, maxvel) ;
    q = (q + 2*q2 + 2*k*rhsq)/3;

    % Limit solution through characteristics
    [qc,R] = EulerQtoRDG(q, gamma, V, iV) ;
    R(:,:,1) = SlopeLimitCSDG(x,R(:,:,1),m,h,N,V,iV) ;
    R(:,:,2) = SlopeLimitCSDG(x,R(:,:,2),m,h,N,V,iV) ;
    R(:,:,3) = SlopeLimitCSDG(x,R(:,:,3),m,h,N,V,iV) ;
    q = EulerRtoQDG(R, qc, gamma, V, iV) ;

    time = time+k; tstep = tstep+1;
end
return

```

In EulerDGrhs1D.m we illustrate the evaluation of the right-hand side of the discontinuous Galerkin method with a Lax–Friedrichs numerical flux.

**Script 12.31. EulerDGrhs1D.m: Evaluation of the right-hand side for solving the one-dimensional Euler equations using a discontinuous Galerkin method.**

---

```

function [rhsq] = EulerDGrhs1D(x,q,h,k,m,N, gamma, S, Ma, VtoE, maxvel)
% function [rhsq] = EulerDGrhs1D(x,q,h,k,m,N, gamma, S, Ma, VtoE, maxvel)
% Purpose : Evaluate the RHS of the Euler equations using a DG method
Imat = eye(m+1); re = zeros(2,N+2); rue = zeros(2,N+2); Ee = zeros(2,N+2);

% Extract solution
r = q(:,:,1); ru = q(:,:,2); E = q(:,:,3);

% Extend data and assign boundary conditions
% [re] = extendDG( r(VtoE) , 'D' , 1.0 , 'D' , 0.125 );
% [rue] = extendDG( ru(VtoE) , 'D' , 0 , 'N' , 0 );
% [Ee] = extendDG( E(VtoE) , 'D' , 2.5 , 'N' , 0 );

[re] = extendDG( r(VtoE) , 'D' , 3.857143 , 'N' , 0 );
[rue] = extendDG( ru(VtoE) , 'D' , 10.141852 , 'D' , 0 );
[Ee] = extendDG( E(VtoE) , 'D' , 39.166661 , 'N' , 0 );

% Compute volume fluxes
p = (gamma-1)*(E - 0.5*ru.^2./r);
fr = ru; fru = ru.^2./r + p; fE = (E+p).*ru./r;

% Compute surface fluxes
fluxr = EulerLF([ re(2,2:N+1)' rue(2,2:N+1)' Ee(2,2:N+1)' ] ,...
    [ re(1,3:N+2)' rue(1,3:N+2)' Ee(1,3:N+2)' ] ,...
    gamma, 0, maxvel );
frr = fluxr(1,:); frur = fluxr(2,:); fEr = fluxr(3,:);
fluxl = EulerLF([ re(2,1:N)' rue(2,1:N)' Ee(2,1:N)' ] ,...
    [ re(1,2:N+1)' rue(1,2:N+1)' Ee(1,2:N+1)' ] ,...
    gamma, 0, maxvel );
frl = fluxl(1,:); frul = fluxl(2,:); fEl = fluxl(3,:);

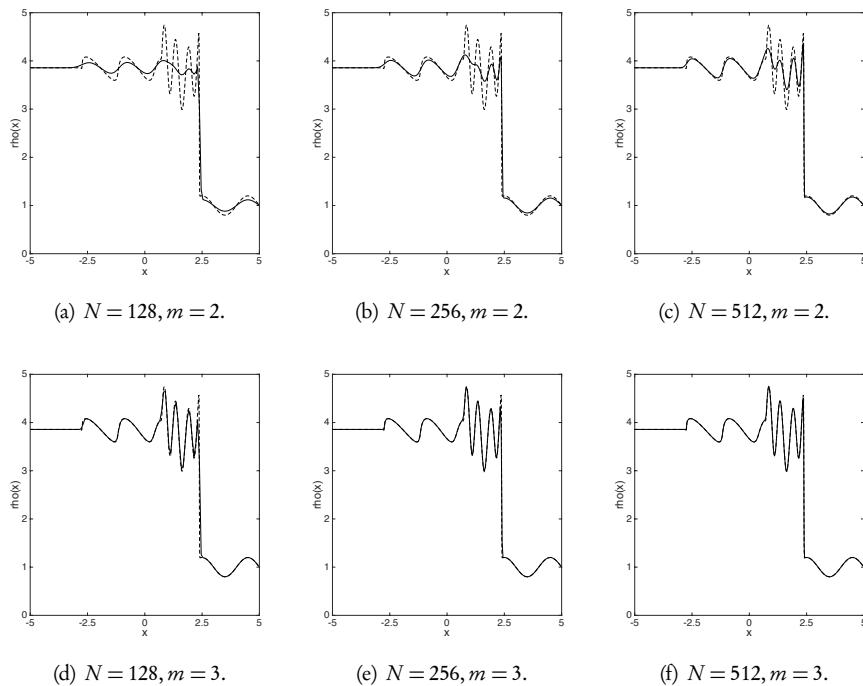
% Compute residual
qh = S'*fr - (Imat(:,m+1)*frr(1,:)-Imat(:,1)*frl(1,:));
rhsq(:,:,1) = (h/2*Ma)\qh;
qh = S'*fru - (Imat(:,m+1)*frur(1,:)-Imat(:,1)*frul(1,:));
rhsq(:,:,2) = (h/2*Ma)\qh;
qh = S'*fE - (Imat(:,m+1)*fEr(1,:)-Imat(:,1)*fEl(1,:));
rhsq(:,:,3) = (h/2*Ma)\qh;
return

```

---

As a test case we consider the shock-entropy test [99], illustrated in Fig. 1.8. The combination of smooth, rapidly varying waves and a shock makes it a complex test case that is particularly well suited to benchmark high-order methods.

We first consider the solution of the shock-entropy problem using a discontinuous Galerkin method with a nonlinear dissipation to control the oscillations. In this particular case we use the method proposed in [70] with  $\nu_0 = 0.75$ , discussed in detail in subsection 12.2.3. Figure 12.25 shows the results of the computation for different choices of  $N$  and  $m$ . While the quality of the results for a coarse resolution is moderate, the accuracy quickly improves for increasing  $N$  and/or  $m$ . For  $N = 512$  and  $m = 3$  the



**Figure 12.25.** Computed density of the Euler equation for the shock-entropy problem using an  $m$ th order discontinuous Galerkin method with  $N$  elements and a nonlinear viscosity [70]. We use  $\nu_0 = 0.75$  and  $CFL = 0.1$  in all tests. The dashed line indicates the reference solution.

agreement with the reference solution is excellent. A direct comparison with the high-order accurate WENO scheme in Fig. 11.17 shows excellent agreement at a comparable level of resolution.

While the nonlinear dissipation works very well, even for this relatively complex test case, let us also consider the performance of the limiters. For scalar problems, the limiter is applied directly to the scalar field. On the other hand, for the system case, limiting should be applied on the characteristic variables to avoid the introduction of additional oscillations. This is particularly important for high-order accurate schemes, as was discussed in detail in subsection 11.3.4 and illustrated in Fig. 11.16.

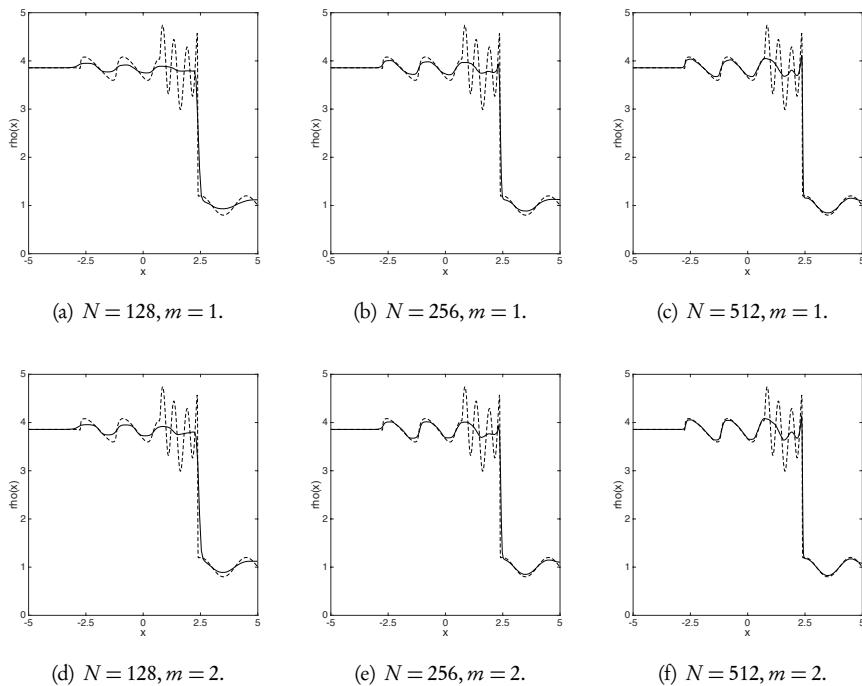
**Script 12.32.** *EulerQtoRDG.m: Recovery of characteristic variables from conserved variables for the Euler equations.*

```

function [qc,R] = EulerQtoRDG(q,gamma,V,iV);
% function [qc,R] = EulerQtoRDG(q,gamma,V,iV);
% Purpose: Compute characteristic values from conserved values with
% transformation based on cell average in DG formulation.
dim = size(q); m = dim(1)-1; N = dim(2); R = zeros(m+1,N,3); qc = zeros(3,N);

% Extract conserved values and compute cell averages
r = q(:,:,1); mu = q(:,:,2); E = q(:,:,3);
rh = iV*r; rh(2:(m+1),:) = 0; ra = V*rh; qc(1,:) = ra(1,:);

% Compute cell averages
qc(2,:) = (qc(1,:)+qc(3,:))/2;
qc(3,:) = (qc(2,:)+qc(4,:))/2;
qc(4,:) = (qc(3,:)+qc(5,:))/2;
qc(5,:) = (qc(4,:)+qc(6,:))/2;
qc(6,:) = (qc(5,:)+qc(7,:))/2;
qc(7,:) = (qc(6,:)+qc(8,:))/2;
qc(8,:) = (qc(7,:)+qc(9,:))/2;
qc(9,:) = (qc(8,:)+qc(10,:))/2;
qc(10,:) = (qc(9,:)+qc(11,:))/2;
qc(11,:) = (qc(10,:)+qc(12,:))/2;
qc(12,:) = (qc(11,:)+qc(13,:))/2;
qc(13,:) = (qc(12,:)+qc(14,:))/2;
qc(14,:) = (qc(13,:)+qc(15,:))/2;
qc(15,:) = (qc(14,:)+qc(16,:))/2;
qc(16,:) = (qc(15,:)+qc(17,:))/2;
qc(17,:) = (qc(16,:)+qc(18,:))/2;
qc(18,:) = (qc(17,:)+qc(19,:))/2;
qc(19,:) = (qc(18,:)+qc(20,:))/2;
qc(20,:) = (qc(19,:)+qc(21,:))/2;
qc(21,:) = (qc(20,:)+qc(22,:))/2;
qc(22,:) = (qc(21,:)+qc(23,:))/2;
qc(23,:) = (qc(22,:)+qc(24,:))/2;
qc(24,:) = (qc(23,:)+qc(25,:))/2;
qc(25,:) = (qc(24,:)+qc(26,:))/2;
qc(26,:) = (qc(25,:)+qc(27,:))/2;
qc(27,:) = (qc(26,:)+qc(28,:))/2;
qc(28,:) = (qc(27,:)+qc(29,:))/2;
qc(29,:) = (qc(28,:)+qc(30,:))/2;
qc(30,:) = (qc(29,:)+qc(31,:))/2;
qc(31,:) = (qc(30,:)+qc(32,:))/2;
qc(32,:) = (qc(31,:)+qc(33,:))/2;
qc(33,:) = (qc(32,:)+qc(34,:))/2;
qc(34,:) = (qc(33,:)+qc(35,:))/2;
qc(35,:) = (qc(34,:)+qc(36,:))/2;
qc(36,:) = (qc(35,:)+qc(37,:))/2;
qc(37,:) = (qc(36,:)+qc(38,:))/2;
qc(38,:) = (qc(37,:)+qc(39,:))/2;
qc(39,:) = (qc(38,:)+qc(40,:))/2;
qc(40,:) = (qc(39,:)+qc(41,:))/2;
qc(41,:) = (qc(40,:)+qc(42,:))/2;
qc(42,:) = (qc(41,:)+qc(43,:))/2;
qc(43,:) = (qc(42,:)+qc(44,:))/2;
qc(44,:) = (qc(43,:)+qc(45,:))/2;
qc(45,:) = (qc(44,:)+qc(46,:))/2;
qc(46,:) = (qc(45,:)+qc(47,:))/2;
qc(47,:) = (qc(46,:)+qc(48,:))/2;
qc(48,:) = (qc(47,:)+qc(49,:))/2;
qc(49,:) = (qc(48,:)+qc(50,:))/2;
qc(50,:) = (qc(49,:)+qc(51,:))/2;
qc(51,:) = (qc(50,:)+qc(52,:))/2;
qc(52,:) = (qc(51,:)+qc(53,:))/2;
qc(53,:) = (qc(52,:)+qc(54,:))/2;
qc(54,:) = (qc(53,:)+qc(55,:))/2;
qc(55,:) = (qc(54,:)+qc(56,:))/2;
qc(56,:) = (qc(55,:)+qc(57,:))/2;
qc(57,:) = (qc(56,:)+qc(58,:))/2;
qc(58,:) = (qc(57,:)+qc(59,:))/2;
qc(59,:) = (qc(58,:)+qc(60,:))/2;
qc(60,:) = (qc(59,:)+qc(61,:))/2;
qc(61,:) = (qc(60,:)+qc(62,:))/2;
qc(62,:) = (qc(61,:)+qc(63,:))/2;
qc(63,:) = (qc(62,:)+qc(64,:))/2;
qc(64,:) = (qc(63,:)+qc(65,:))/2;
qc(65,:) = (qc(64,:)+qc(66,:))/2;
qc(66,:) = (qc(65,:)+qc(67,:))/2;
qc(67,:) = (qc(66,:)+qc(68,:))/2;
qc(68,:) = (qc(67,:)+qc(69,:))/2;
qc(69,:) = (qc(68,:)+qc(70,:))/2;
qc(70,:) = (qc(69,:)+qc(71,:))/2;
qc(71,:) = (qc(70,:)+qc(72,:))/2;
qc(72,:) = (qc(71,:)+qc(73,:))/2;
qc(73,:) = (qc(72,:)+qc(74,:))/2;
qc(74,:) = (qc(73,:)+qc(75,:))/2;
qc(75,:) = (qc(74,:)+qc(76,:))/2;
qc(76,:) = (qc(75,:)+qc(77,:))/2;
qc(77,:) = (qc(76,:)+qc(78,:))/2;
qc(78,:) = (qc(77,:)+qc(79,:))/2;
qc(79,:) = (qc(78,:)+qc(80,:))/2;
qc(80,:) = (qc(79,:)+qc(81,:))/2;
qc(81,:) = (qc(80,:)+qc(82,:))/2;
qc(82,:) = (qc(81,:)+qc(83,:))/2;
qc(83,:) = (qc(82,:)+qc(84,:))/2;
qc(84,:) = (qc(83,:)+qc(85,:))/2;
qc(85,:) = (qc(84,:)+qc(86,:))/2;
qc(86,:) = (qc(85,:)+qc(87,:))/2;
qc(87,:) = (qc(86,:)+qc(88,:))/2;
qc(88,:) = (qc(87,:)+qc(89,:))/2;
qc(89,:) = (qc(88,:)+qc(90,:))/2;
qc(90,:) = (qc(89,:)+qc(91,:))/2;
qc(91,:) = (qc(90,:)+qc(92,:))/2;
qc(92,:) = (qc(91,:)+qc(93,:))/2;
qc(93,:) = (qc(92,:)+qc(94,:))/2;
qc(94,:) = (qc(93,:)+qc(95,:))/2;
qc(95,:) = (qc(94,:)+qc(96,:))/2;
qc(96,:) = (qc(95,:)+qc(97,:))/2;
qc(97,:) = (qc(96,:)+qc(98,:))/2;
qc(98,:) = (qc(97,:)+qc(99,:))/2;
qc(99,:) = (qc(98,:)+qc(100,:))/2;
qc(100,:) = (qc(99,:)+qc(101,:))/2;
qc(101,:) = (qc(100,:)+qc(102,:))/2;
qc(102,:) = (qc(101,:)+qc(103,:))/2;
qc(103,:) = (qc(102,:)+qc(104,:))/2;
qc(104,:) = (qc(103,:)+qc(105,:))/2;
qc(105,:) = (qc(104,:)+qc(106,:))/2;
qc(106,:) = (qc(105,:)+qc(107,:))/2;
qc(107,:) = (qc(106,:)+qc(108,:))/2;
qc(108,:) = (qc(107,:)+qc(109,:))/2;
qc(109,:) = (qc(108,:)+qc(110,:))/2;
qc(110,:) = (qc(109,:)+qc(111,:))/2;
qc(111,:) = (qc(110,:)+qc(112,:))/2;
qc(112,:) = (qc(111,:)+qc(113,:))/2;
qc(113,:) = (qc(112,:)+qc(114,:))/2;
qc(114,:) = (qc(113,:)+qc(115,:))/2;
qc(115,:) = (qc(114,:)+qc(116,:))/2;
qc(116,:) = (qc(115,:)+qc(117,:))/2;
qc(117,:) = (qc(116,:)+qc(118,:))/2;
qc(118,:) = (qc(117,:)+qc(119,:))/2;
qc(119,:) = (qc(118,:)+qc(120,:))/2;
qc(120,:) = (qc(119,:)+qc(121,:))/2;
qc(121,:) = (qc(120,:)+qc(122,:))/2;
qc(122,:) = (qc(121,:)+qc(123,:))/2;
qc(123,:) = (qc(122,:)+qc(124,:))/2;
qc(124,:) = (qc(123,:)+qc(125,:))/2;
qc(125,:) = (qc(124,:)+qc(126,:))/2;
qc(126,:) = (qc(125,:)+qc(127,:))/2;
qc(127,:) = (qc(126,:)+qc(128,:))/2;
qc(128,:) = (qc(127,:)+qc(129,:))/2;
qc(129,:) = (qc(128,:)+qc(130,:))/2;
qc(130,:) = (qc(129,:)+qc(131,:))/2;
qc(131,:) = (qc(130,:)+qc(132,:))/2;
qc(132,:) = (qc(131,:)+qc(133,:))/2;
qc(133,:) = (qc(132,:)+qc(134,:))/2;
qc(134,:) = (qc(133,:)+qc(135,:))/2;
qc(135,:) = (qc(134,:)+qc(136,:))/2;
qc(136,:) = (qc(135,:)+qc(137,:))/2;
qc(137,:) = (qc(136,:)+qc(138,:))/2;
qc(138,:) = (qc(137,:)+qc(139,:))/2;
qc(139,:) = (qc(138,:)+qc(140,:))/2;
qc(140,:) = (qc(139,:)+qc(141,:))/2;
qc(141,:) = (qc(140,:)+qc(142,:))/2;
qc(142,:) = (qc(141,:)+qc(143,:))/2;
qc(143,:) = (qc(142,:)+qc(144,:))/2;
qc(144,:) = (qc(143,:)+qc(145,:))/2;
qc(145,:) = (qc(144,:)+qc(146,:))/2;
qc(146,:) = (qc(145,:)+qc(147,:))/2;
qc(147,:) = (qc(146,:)+qc(148,:))/2;
qc(148,:) = (qc(147,:)+qc(149,:))/2;
qc(149,:) = (qc(148,:)+qc(150,:))/2;
qc(150,:) = (qc(149,:)+qc(151,:))/2;
qc(151,:) = (qc(150,:)+qc(152,:))/2;
qc(152,:) = (qc(151,:)+qc(153,:))/2;
qc(153,:) = (qc(152,:)+qc(154,:))/2;
qc(154,:) = (qc(153,:)+qc(155,:))/2;
qc(155,:) = (qc(154,:)+qc(156,:))/2;
qc(156,:) = (qc(155,:)+qc(157,:))/2;
qc(157,:) = (qc(156,:)+qc(158,:))/2;
qc(158,:) = (qc(157,:)+qc(159,:))/2;
qc(159,:) = (qc(158,:)+qc(160,:))/2;
qc(160,:) = (qc(159,:)+qc(161,:))/2;
qc(161,:) = (qc(160,:)+qc(162,:))/2;
qc(162,:) = (qc(161,:)+qc(163,:))/2;
qc(163,:) = (qc(162,:)+qc(164,:))/2;
qc(164,:) = (qc(163,:)+qc(165,:))/2;
qc(165,:) = (qc(164,:)+qc(166,:))/2;
qc(166,:) = (qc(165,:)+qc(167,:))/2;
qc(167,:) = (qc(166,:)+qc(168,:))/2;
qc(168,:) = (qc(167,:)+qc(169,:))/2;
qc(169,:) = (qc(168,:)+qc(170,:))/2;
qc(170,:) = (qc(169,:)+qc(171,:))/2;
qc(171,:) = (qc(170,:)+qc(172,:))/2;
qc(172,:) = (qc(171,:)+qc(173,:))/2;
qc(173,:) = (qc(172,:)+qc(174,:))/2;
qc(174,:) = (qc(173,:)+qc(175,:))/2;
qc(175,:) = (qc(174,:)+qc(176,:))/2;
qc(176,:) = (qc(175,:)+qc(177,:))/2;
qc(177,:) = (qc(176,:)+qc(178,:))/2;
qc(178,:) = (qc(177,:)+qc(179,:))/2;
qc(179,:) = (qc(178,:)+qc(180,:))/2;
qc(180,:) = (qc(179,:)+qc(181,:))/2;
qc(181,:) = (qc(180,:)+qc(182,:))/2;
qc(182,:) = (qc(181,:)+qc(183,:))/2;
qc(183,:) = (qc(182,:)+qc(184,:))/2;
qc(184,:) = (qc(183,:)+qc(185,:))/2;
qc(185,:) = (qc(184,:)+qc(186,:))/2;
qc(186,:) = (qc(185,:)+qc(187,:))/2;
qc(187,:) = (qc(186,:)+qc(188,:))/2;
qc(188,:) = (qc(187,:)+qc(189,:))/2;
qc(189,:) = (qc(188,:)+qc(190,:))/2;
qc(190,:) = (qc(189,:)+qc(191,:))/2;
qc(191,:) = (qc(190,:)+qc(192,:))/2;
qc(192,:) = (qc(191,:)+qc(193,:))/2;
qc(193,:) = (qc(192,:)+qc(194,:))/2;
qc(194,:) = (qc(193,:)+qc(195,:))/2;
qc(195,:) = (qc(194,:)+qc(196,:))/2;
qc(196,:) = (qc(195,:)+qc(197,:))/2;
qc(197,:) = (qc(196,:)+qc(198,:))/2;
qc(198,:) = (qc(197,:)+qc(199,:))/2;
qc(199,:) = (qc(198,:)+qc(200,:))/2;
qc(200,:) = (qc(199,:)+qc(201,:))/2;
qc(201,:) = (qc(200,:)+qc(202,:))/2;
qc(202,:) = (qc(201,:)+qc(203,:))/2;
qc(203,:) = (qc(202,:)+qc(204,:))/2;
qc(204,:) = (qc(203,:)+qc(205,:))/2;
qc(205,:) = (qc(204,:)+qc(206,:))/2;
qc(206,:) = (qc(205,:)+qc(207,:))/2;
qc(207,:) = (qc(206,:)+qc(208,:))/2;
qc(208,:) = (qc(207,:)+qc(209,:))/2;
qc(209,:) = (qc(208,:)+qc(210,:))/2;
qc(210,:) = (qc(209,:)+qc(211,:))/2;
qc(211,:) = (qc(210,:)+qc(212,:))/2;
qc(212,:) = (qc(211,:)+qc(213,:))/2;
qc(213,:) = (qc(212,:)+qc(214,:))/2;
qc(214,:) = (qc(213,:)+qc(215,:))/2;
qc(215,:) = (qc(214,:)+qc(216,:))/2;
qc(216,:) = (qc(215,:)+qc(217,:))/2;
qc(217,:) = (qc(216,:)+qc(218,:))/2;
qc(218,:) = (qc(217,:)+qc(219,:))/2;
qc(219,:) = (qc(218,:)+qc(220,:))/2;
qc(220,:) = (qc(219,:)+qc(221,:))/2;
qc(221,:) = (qc(220,:)+qc(222,:))/2;
qc(222,:) = (qc(221,:)+qc(223,:))/2;
qc(223,:) = (qc(222,:)+qc(224,:))/2;
qc(224,:) = (qc(223,:)+qc(225,:))/2;
qc(225,:) = (qc(224,:)+qc(226,:))/2;
qc(226,:) = (qc(225,:)+qc(227,:))/2;
qc(227,:) = (qc(226,:)+qc(228,:))/2;
qc(228,:) = (qc(227,:)+qc(229,:))/2;
qc(229,:) = (qc(228,:)+qc(230,:))/2;
qc(230,:) = (qc(229,:)+qc(231,:))/2;
qc(231,:) = (qc(230,:)+qc(232,:))/2;
qc(232,:) = (qc(231,:)+qc(233,:))/2;
qc(233,:) = (qc(232,:)+qc(234,:))/2;
qc(234,:) = (qc(233,:)+qc(235,:))/2;
qc(235,:) = (qc(234,:)+qc(236,:))/2;
qc(236,:) = (qc(235,:)+qc(237,:))/2;
qc(237,:) = (qc(236,:)+qc(238,:))/2;
qc(238,:) = (qc(237,:)+qc(239,:))/2;
qc(239,:) = (qc(238,:)+qc(240,:))/2;
qc(240,:) = (qc(239,:)+qc(241,:))/2;
qc(241,:) = (qc(240,:)+qc(242,:))/2;
qc(242,:) = (qc(241,:)+qc(243,:))/2;
qc(243,:) = (qc(242,:)+qc(244,:))/2;
qc(244,:) = (qc(243,:)+qc(245,:))/2;
qc(245,:) = (qc(244,:)+qc(246,:))/2;
qc(246,:) = (qc(245,:)+qc(247,:))/2;
qc(247,:) = (qc(246,:)+qc(248,:))/2;
qc(248,:) = (qc(247,:)+qc(249,:))/2;
qc(249,:) = (qc(248,:)+qc(250,:))/2;
qc(250,:) = (qc(249,:)+qc(251,:))/2;
qc(251,:) = (qc(250,:)+qc(252,:))/2;
qc(252,:) = (qc(251,:)+qc(253,:))/2;
qc(253,:) = (qc(252,:)+qc(254,:))/2;
qc(254,:) = (qc(253,:)+qc(255,:))/2;
qc(255,:) = (qc(254,:)+qc(256,:))/2;
qc(256,:) = (qc(255,:)+qc(257,:))/2;
qc(257,:) = (qc(256,:)+qc(258,:))/2;
qc(258,:) = (qc(257,:)+qc(259,:))/2;
qc(259,:) = (qc(258,:)+qc(260,:))/2;
qc(260,:) = (qc(259,:)+qc(261,:))/2;
qc(261,:) = (qc(260,:)+qc(262,:))/2;
qc(262,:) = (qc(261,:)+qc(263,:))/2;
qc(263,:) = (qc(262,:)+qc(264,:))/2;
qc(264,:) = (qc(263,:)+qc(265,:))/2;
qc(265,:) = (qc(264,:)+qc(266,:))/2;
qc(266,:) = (qc(265,:)+qc(267,:))/2;
qc(267,:) = (qc(266,:)+qc(268,:))/2;
qc(268,:) = (qc(267,:)+qc(269,:))/2;
qc(269,:) = (qc(268,:)+qc(270,:))/2;
qc(270,:) = (qc(269,:)+qc(271,:))/2;
qc(271,:) = (qc(270,:)+qc(272,:))/2;
qc(272,:) = (qc(271,:)+qc(273,:))/2;
qc(273,:) = (qc(272,:)+qc(274,:))/2;
qc(274,:) = (qc(273,:)+qc(275,:))/2;
qc(275,:) = (qc(274,:)+qc(276,:))/2;
qc(276,:) = (qc(275,:)+qc(277,:))/2;
qc(277,:) = (qc(276,:)+qc(278,:))/2;
qc(278,:) = (qc(277,:)+qc(279,:))/2;
qc(279,:) = (qc(278,:)+qc(280,:))/2;
qc(280,:) = (qc(279,:)+qc(281,:))/2;
qc(281,:) = (qc(280,:)+qc(282,:))/2;
qc(282,:) = (qc(281,:)+qc(283,:))/2;
qc(283,:) = (qc(282,:)+qc(284,:))/2;
qc(284,:) = (qc(283,:)+qc(285,:))/2;
qc(285,:) = (qc(284,:)+qc(286,:))/2;
qc(286,:) = (qc(285,:)+qc(287,:))/2;
qc(287,:) = (qc(286,:)+qc(288,:))/2;
qc(288,:) = (qc(287,:)+qc(289,:))/2;
qc(289,:) = (qc(288,:)+qc(290,:))/2;
qc(290,:) = (qc(289,:)+qc(291,:))/2;
qc(291,:) = (qc(290,:)+qc(292,:))/2;
qc(292,:) = (qc(291,:)+qc(293,:))/2;
qc(293,:) = (qc(292,:)+qc(294,:))/2;
qc(294,:) = (qc(293,:)+qc(295,:))/2;
qc(295,:) = (qc(294,:)+qc(296,:))/2;
qc(296,:) = (qc(295,:)+qc(297,:))/2;
qc(297,:) = (qc(296,:)+qc(298,:))/2;
qc(298,:) = (qc(297,:)+qc(299,:))/2;
qc(299,:) = (qc(298,:)+qc(300,:))/2;
qc(300,:) = (qc(299,:)+qc(301,:))/2;
qc(301,:) = (qc(300,:)+qc(302,:))/2;
qc(302,:) = (qc(301,:)+qc(303,:))/2;
qc(303,:) = (qc(302,:)+qc(304,:))/2;
qc(304,:) = (qc(303,:)+qc(305,:))/2;
qc(305,:) = (qc(304,:)+qc(306,:))/2;
qc(306,:) = (qc(305,:)+qc(307,:))/2;
qc(307,:) = (qc(306,:)+qc(308,:))/2;
qc(308,:) = (qc(307,:)+qc(309,:))/2;
qc(309,:) = (qc(308,:)+qc(310,:))/2;
qc(310,:) = (qc(309,:)+qc(311,:))/2;
qc(311,:) = (qc(310,:)+qc(312,:))/2;
qc(312,:) = (qc(311,:)+qc(313,:))/2;
qc(313,:) = (qc(312,:)+qc(314,:))/2;
qc(314,:) = (qc(313,:)+qc(315,:))/2;
qc(315,:) = (qc(314,:)+qc(316,:))/2;
qc(316,:) = (qc(315,:)+qc(317,:))/2;
qc(317,:) = (qc(316,:)+qc(318,:))/2;
qc(318,:) = (qc(317,:)+qc(319,:))/2;
qc(319,:) = (qc(318,:)+qc(320,:))/2;
qc(320,:) = (qc(319,:)+qc(321,:))/2;
qc(321,:) = (qc(320,:)+qc(322,:))/2;
qc(322,:) = (qc(321,:)+qc(323,:))/2;
qc(323,:) = (qc(322,:)+qc(324,:))/2;
qc(324,:) = (qc(323,:)+qc(325,:))/2;
qc(325,:) = (qc(324,:)+qc(326,:))/2;
qc(326,:) = (qc(325,:)+qc(327,:))/2;
qc(327,:) = (qc(326,:)+qc(328,:))/2;
qc(328,:) = (qc(327,:)+qc(329,:))/2;
qc(329,:) = (qc(328,:)+qc(330,:))/2;
qc(330,:) = (qc(329,:)+qc(331,:))/2;
qc(331,:) = (qc(330,:)+qc(332,:))/2;
qc(332,:) = (qc(331,:)+qc(333,:))/2;
qc(333,:) = (qc(332,:)+qc(334,:))/2;
qc(334,:) = (qc(333,:)+qc(335,:))/2;
qc(335,:) = (qc(334,:)+qc(336,:))/2;
qc(336,:) = (qc(335,:)+qc(337,:))/2;
qc(337,:) = (qc(336,:)+qc(338,:))/2;
qc(338,:) = (qc(337,:)+qc(339,:))/2;
qc(339,:) = (qc(338,:)+qc(340,:))/2;
qc(340,:) = (qc(339,:)+qc(341,:))/2;
qc(341,:) = (qc(340,:)+qc(342,:))/2;
qc(342,:) = (qc(341,:)+qc(343,:))/2;
qc(343,:) = (qc(342,:)+qc(344,:))/2;
qc(344,:) = (qc(343,:)+qc(345,:))/2;
qc(345,:) = (qc(344,:)+qc(346,:))/2;
qc(346,:) = (qc(345,:)+qc(347,:))/2;
qc(347,:) = (qc(346,:)+qc(348,:))/2;
qc(348,:) = (qc(347,:)+qc(349,:))/2;
qc(349,:) = (qc(348,:)+qc(350,:))/2;
qc(350,:) = (qc(349,:)+qc(351,:))/2;
qc(351,:) = (qc(350,:)+qc(352,:))/2;
qc(352,:) = (qc(351,:)+qc(353,:))/2;
qc(353,:) = (qc(352,:)+qc(354,:))/2;
qc(354,:) = (qc(353,:)+qc(355,:))/2;
qc(355,:) = (qc(354,:)+qc(356,:))/2;
qc(356,:) = (qc(355,:)+qc(357,:))/2;
qc(357,:) = (qc(356,:)+qc(358,:))/2;
qc(358,:) = (qc(357,:)+qc(359,:))/2;
qc(359,:) = (qc(358,:)+qc(360,:))/2;
qc(360,:) = (qc(359,:)+qc(361,:))/2;
qc(361,:) = (qc(360,:)+qc(362,:))/2;
qc(362,:) = (qc(361,:)+qc(363,:))/2;
qc(363,:) = (qc(362,:)+qc(364,:))/2;
qc(364,:) = (qc(363,:)+qc(365,:))/2;
qc(365,:) = (qc(364,:)+qc(366,:))/2;
qc(366,:) = (qc(365,:)+qc(367,:))/2;
qc(367,:) = (qc(366,:)+qc(368,:))/2;
qc(368,:) = (qc(367,:)+qc(369,:))/2;
qc(369,:) = (qc(368,:)+qc(370,:))/2;
qc(370,:) = (qc(369,:)+qc(371,:))/2;
qc(371,:) = (qc(370,:)+qc(372,:))/2;
qc(372,:) = (qc(371,:)+qc(373,:))/2;
qc(373,:) = (qc(372,:)+qc(374,:))/2;
qc(374,:) = (qc(373,:)+qc(375,:))/2;
qc(375,:) = (qc(374,:)+qc(376,:))/2;
qc(376,:) = (qc(375,:)+qc(377,:))/2;
qc(377,:) = (qc(376,:)+qc(378,:))/2;
qc(378,:) = (qc(377,:)+qc(379,:))/2;
qc(379,:) = (qc(378,:)+qc(380,:))/2;
qc(380,:) = (qc(379,:)+qc(381,:))/2;
qc(381,:) = (qc(380,:)+qc(382,:))/2;
qc(382,:) = (qc(381,:)+qc(383,:))/2;
qc(383,:) = (qc(382,:)+qc(384,:))/2;
qc(384,:) = (qc(383,:)+qc(385,:))/2;
qc(385,:) = (qc(384,:)+qc(386,:))/2;
qc(386,:) = (qc(385,:)+qc(387,:))/2;
qc(387,:) = (qc(386,:)+qc(388,:))/2;
qc(388,:) = (qc(387,:)+qc(389,:))/2;
qc(389,:) = (qc(388,:)+qc(390,:))/2;
qc(390,:) = (qc(389,:)+qc(391,:))/2;
qc(391,:) = (qc(390,:)+qc(392,:))/2;
qc(392,:) = (qc(391,:)+qc(393,:))/2;
qc(393,:) = (qc(392,:)+qc(394,:))/2;
qc(394,:) = (qc(393,:)+qc(395,:))/2;
qc(395,:) = (qc(394,:)+qc(396,:))/2;
qc(396,:) = (qc(395,:)+qc(397,:))/2;
qc(397,:) = (qc(396,:)+qc(398,:))/2;
qc(398,:) = (qc(397,:)+qc(399,:))/2;
qc(399,:) = (qc(398,:)+qc(400,:))/2;
qc(400,:) = (qc(399,:)+qc(401,:))/2;
qc(401,:) = (qc(400,:)+qc(402,:))/2;
qc(402,:) = (qc(401,:)+qc(403,:))/2;
qc(403,:) = (qc(402,:)+qc(404,:))/2;
qc(404,:) = (qc(403,:)+qc(405,:))/2;
qc(405,:) = (qc(404,:)+qc(406,:))/2;
qc(406,:) = (qc(405,:)+qc(407,:))/2;
qc(407,:) = (qc(406,:)+qc(408,:))/2;
qc(408,:) = (qc(407,:)+qc(409,:))/2;
qc(409,:) = (qc(408,:)+qc(410,:))/2;
qc(410,:) = (qc(409,:)+qc(411,:))/2;
qc(411,:) = (qc(410,:)+qc(412,:))/2;
qc(412,:) = (qc(411,:)+qc(413,:))/2;
qc(413,:) = (qc(412,:)+qc(414,:))/2;
qc(414,:) = (qc(413,:)+qc(415,:))/2;
qc(415,:) = (qc(414,:)+qc(416,:))/2;
qc(416,:) = (qc(415,:)+qc(417,:))/2;
qc(417,:) = (qc(416,:)+qc(418,:))/2;
qc(418,:) = (qc(417,:)+qc(419,:))/2;
qc(419,:) = (qc(418,:)+qc(420,:))/2;
qc(420,:) = (qc(419,:)+qc(421,:))/2;
qc(421,:) = (qc(420,:)+qc(422,:))/2;
qc(422,:) = (qc(421,:)+qc(423,:))/2;
qc(423,:) = (qc(422,:)+qc(424,:))/2;
qc(424,:) = (qc(423,:)+qc(425,:))/2;
qc(425,:) = (qc(424,:)+qc(426,:))/2;
qc(426,:) = (qc(425,:)+qc(427,:))/2;
qc(427,:) = (qc(426,:)+qc(428,:))/2;
qc(428,:) = (qc(427,:)+qc(429,:))/2;
qc(429,:) = (qc(428,:)+qc(430,:))/2;
qc(430,:) = (qc(429,:)+qc(431,:))/2;
qc(431,:) = (qc(430,:)+qc(432,:))/2;
qc(432,:) = (qc(431,:)+qc(433,:))/2;
qc(433,:) = (qc(432,:)+qc(434,:))/2;
qc(434,:) = (qc(433,:)+qc(435,:))/2;
qc(435,:) = (qc(434,:)+qc(436,:))/2;
qc(436,:) = (qc(435,:)+qc(437,:))/2;
qc(437,:) = (qc(436,:)+qc(438,:))/2;
qc(438,:) = (qc(437,:)+qc(439,:))/2;
qc(439,:) = (qc(438,:)+qc(440,:))/2;
qc(440,:) = (qc(439,:)+qc(441,:))/2;
qc(441,:) = (qc(440,:)+qc(442,:))/2;
qc(442,:) = (qc(441,:)+qc(443,:))/2;
qc(443,:) = (qc(442,:)+qc(444,:))/2;
qc(444,:) = (qc(443,:)+qc(445,:))/2;
qc(445,:) = (qc(444,:)+qc(446,:))/2;
qc(446,:) = (qc(445,:)+qc(447,:))/2;
qc(447,:) = (qc(446,:)+qc(448,:))/2;
qc(448,:) = (qc(447,:)+qc(449,:))/2;
qc(449,:) = (qc(448,:)+qc(450,:))/2;
qc(450,:) = (qc(449,:)+qc(451,:))/2;
qc(451,:) = (qc(450,:)+qc(452,:))/2;
qc(452,:) = (qc(451,:)+qc(453,:))/2;
qc(453,:) = (qc(452,:)+qc(454,:))/2;
qc(454,:) = (qc(453,:)+qc(455,:))/2;
qc(455,:) = (qc(454,:)+qc(456,:))/2;
qc(456,:) = (qc(455,:)+qc(457,:))/2;
qc(457,:) = (qc(456,:)+qc(458,:))/2;
qc(458,:) = (qc(457,:)+qc(459,:))/2;
qc(459,:) = (qc(458,:)+qc(460,:))/2;
qc(460,:) = (qc(459,:)+qc(461,:))/2;
qc(461,:) = (qc(460,:)+qc(462,:))/2;
qc(462,:) = (qc(461,:)+qc(463,:))/2;
qc(463,:) = (qc(462,:)+qc(464,:))/2;
qc(464,:) = (qc(463,:)+qc(465,:))/2;
qc(465,:) = (qc(464,:)+qc(466,:))/2;
qc(466,:) = (qc(465,:)+qc(467,:))/2;
qc(467,:) = (qc(466,:)+qc(468,:))/2;
qc(468,:) = (qc(467,:)+qc(469,:))/2;
qc(469,:) = (qc(468,:)+qc(470,:))/2;
qc(470,:) = (qc(469,:)+qc(471,:))/2;
qc(471,:) = (qc(470,:)+qc(472,:))/2;
qc(472,:) = (qc(471,:)+qc(473,:))/2;
qc(473,:) = (qc(472,:)+qc(474,:))/2;
qc(474,:) = (qc(473,:)+qc(475,:))/2;
qc(475,:) = (qc(474,:)+qc(476,:))/2;
qc(476,:) = (qc(475,:)+qc(477,:))/2;
qc(477,:) = (qc(476,:)+qc(478,:))/2;
qc(478,:) = (qc(477,:)+qc(479,:))/2;
qc(479,:) = (qc(478,:)+qc(480,:))/2;
qc(480,:) = (qc(479,:)+qc(481,:))/2;
qc(481,:) = (qc(480,:)+qc(482,:))/2;
qc(482,:) = (qc(481,:)+qc(483,:))/2;
qc(483,:) = (qc(482,:)+qc(484,:))/2;
qc(484,:) = (qc(483,:)+qc(485,:))/2;
qc(485,:) = (qc(484,:)+qc(486,:))/2;
qc(486,:) = (qc(485,:)+qc(487,:))/2;
qc(487,:) = (qc(486,:)+qc(488,:))/2;
qc(488,:) = (qc(487,:)+qc(489,:))/
```



**Figure 12.26.** Computed density of the Euler equation for the shock-entropy problem using an  $m$ th order discontinuous Galerkin method with  $N$  elements and a TVD-based, troubled-cell indicator. We use a TVD limiter [27, 26] and  $CFL = 0.1$  in all tests. The dashed line indicates the reference solution.

```
muh = iV*mu; muh(2:(m+1),:) = 0; mua = V*muh; qc(2,:) = mua(1,:);
```

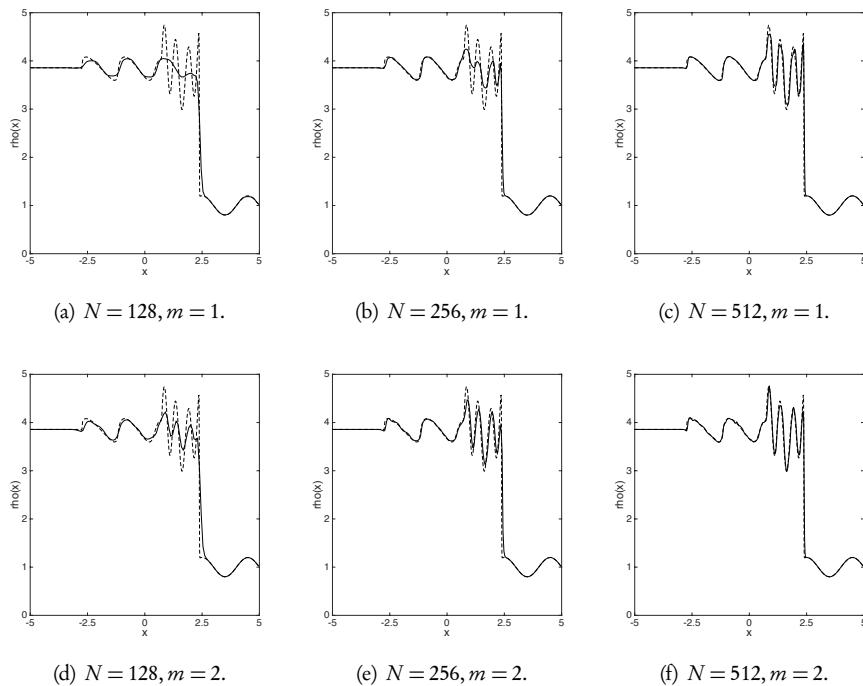
```
Eh = iV*E; Eh(2:(m+1),:) = 0; Ea = V*Eh; qc(3,:) = Ea(1,:);
```

```
% Compute characteristic values
for i=1:N
    [S, iS, Lam] = EulerChar([qc(1,i) qc(2,i) qc(3,i)],gamma);
    qh = [r(:,i) mu(:,i) E(:,i)]; Ch = (iS*qh')';
    R(:,i,1) = Ch(:,1); R(:,i,2) = Ch(:,2); R(:,i,3) = Ch(:,3);
end
return
```

In EulerQtoRDG.m we show recovery of the characteristic variables from the conserved variables through a similarity transform, Ex. 3.3, based on the cell average. The reverse action is enabled by EulerRtoQDG.m. As illustrated in EulerDG1D.m, these are used before and after limiting.

**Script 12.33.** EulerRtoQDG.m: Routine to recover conserved variables from characteristic variables for the Euler equations.

```
function [q] = EulerRtoQDG(R, qc, gamma, V, iV);
% function [q] = EulerRtoQ(R, qc, gamma, V, iV);
```



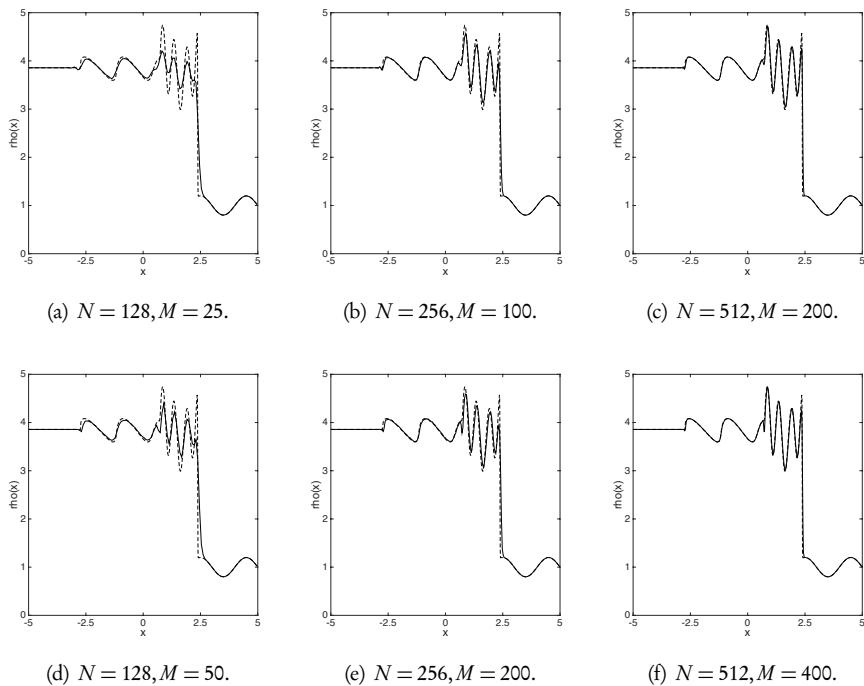
**Figure 12.27.** Computed density of the Euler equation for the shock-entropy problem using an  $m$ th order discontinuous Galerkin method with  $N$  elements and a TVD-based troubled-cell indicator. We use a WENO-based limiter [128] and  $CFL = 0.1$  in all tests. The dashed line indicates the reference solution.

```
% Purpose: Compute conserved values from characteristic values with
% transformation based on cell average in DG formulation.
dim = size(R); m = dim(1)-1; N = dim(2); q = zeros(m+1,N,3);

% Compute conserved values
for i=1:N
    [S, iS, Lam] = EulerChar([qc(1,i) qc(2,i) qc(3,i)],gamma);
    qh = [R(:,i,1) R(:,i,2) R(:,i,3)]; Ch = (S*qh)';
    q(:,i,1) = Ch(:,1); q(:,i,2) = Ch(:,2); q(:,i,3) = Ch(:,3);
end
return
```

We first consider the use of the TVD limiter [27, 26] with a simple minmod-based, troubled-cell indicator. The results are shown in Fig. 12.26 for different choices of  $N$  and  $m$ . While we do not observe any artificial oscillations and do see clear signs of convergence for increasing resolution, the results are inferior to those obtained with the nonlinear dissipation, shown in Fig. 12.25, or with a WENO method, as shown in Fig. 11.16.

Figure 12.27 shows the results of the same experiment performed with a WENO-based limiter. Compared to the results in Fig. 12.26, we see a substantial improvement



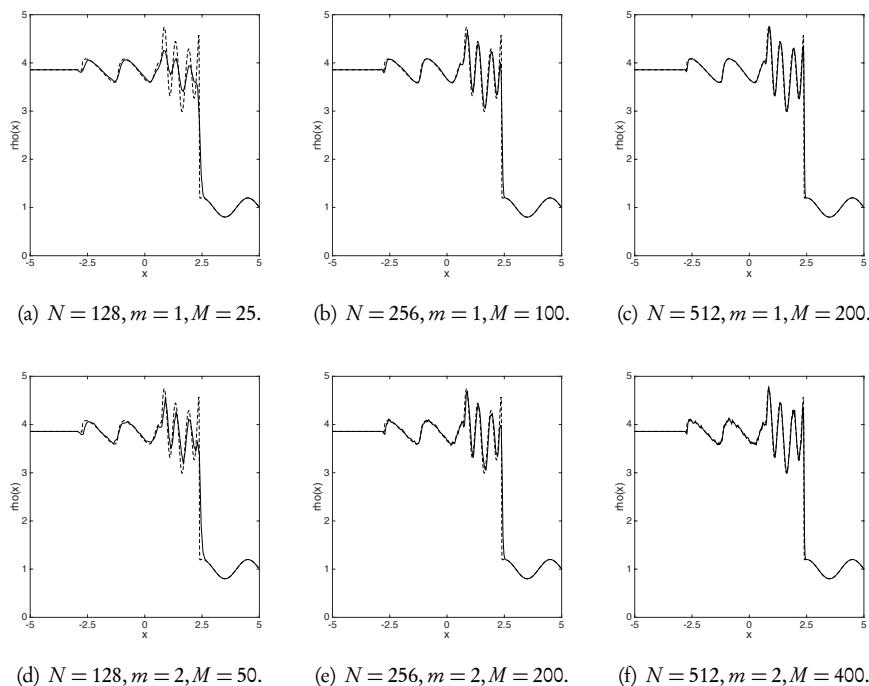
**Figure 12.28.** Computed density of the Euler equation for the shock-entropy problem using an  $m$ th order discontinuous Galerkin method with  $N$  elements and a TVB-based troubled-cell indicator. We use a TVD limiter [27, 26] and  $CFL = 0.1$  in all tests. The dashed line indicates the reference solution.

in the overall accuracy of the results along with enhanced convergence when increasing  $N$  and/or  $m$ . However, when compared to the results in Fig. 12.25 or Fig. 11.16, the overall quality of the results remains inferior.

To improve on this situation, let us recall the results in Fig. 12.21, where we used the TVB-minmod as a troubled-cell indicator and observed substantial improvements. It is natural to pursue a similar approach here, even if this requires an experimental determination of the value of  $M$  in the TVB-minmod function (10.15).

We first repeat the computations for the TVD-based limiter, employing the TVB-minmod function as the troubled cell indicator. The results are shown in Fig. 12.28 for increasing resolution. When compared to the results in Fig. 12.26, we observe a very substantial improvement in the accuracy and convergence of the scheme. The values of  $M$  used in the troubled-cell indicator are given in Fig. 12.28, and have been optimized experimentally. We observe a weak dependence of  $M$  on  $N$  and  $m$ , as one would expect.

In Fig. 12.29 we show the same set of results obtained with a WENO-based limiter in combination with a TVB-minmod-based troubled-cell indicator. We again observe a substantial improvement over the results in Fig. 12.27, and an enhanced convergence when increasing  $N$  or  $m$ . When compared to the results in Fig. 12.25 or Fig. 11.16, the accuracy of the results are comparable across the different methods.



**Figure 12.29.** Computed density of the Euler equation for the shock-entropy problem using an  $m$ th order discontinuous Galerkin method with  $N$  elements and a TVB-based troubled-cell indicator. We use a WENO limiter [128] and  $CFL = 0.1$  in all tests. The dashed line indicates the reference solution.

## References

- [1] Saul Abarbanel, David Gottlieb, and Eitan Tadmor. *Spectral Methods for Discontinuous Problems*, ICASE Report NASA-CA-177974. Langley Research Center, NASA, 1985.
- [2] Rémi Abgrall. On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation. *Journal of Computational Physics*, 114(1):45–58, 1994.
- [3] Milton Abramowitz and Irene A Stegun. *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards, Applied Mathematics Series, volume 55. Courier Corporation, 1964.
- [4] Mark Ainsworth. Dispersive and dissipative behaviour of high order discontinuous Galerkin finite element methods. *Journal of Computational Physics*, 198(1):106–130, 2004.
- [5] Cea Basdevant, M. Deville, P. Haldenwang, J. M. Lacroix, J. Ouazzani, R. Peyret, P. Orlandi, and A. T. Patera. Spectral and finite difference solutions of the Burgers equation. *Computers & Fluids*, 14(1):23–41, 1986.

- [6] Christine Bernardi and Yvon Maday. Polynomial interpolation results in Sobolev spaces. *Journal of Computational and Applied Mathematics*, 43(1):53–80, 1992.
- [7] Rupak Biswas, Karen D. Devine, and Joseph E. Flaherty. Parallel, adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14(1):255–283, 1994.
- [8] Andrea Bonito, Jean-Luc Guermond, and Bojan Popov. Stability analysis of explicit entropy viscosity methods for non-linear scalar conservation equations. *Mathematics of Computation*, 83:1039–1062, 2014.
- [9] Susanne C. Brenner and Ridgway Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of Texts in Applied Mathematics. Springer Science+Business Media, 2008.
- [10] Franco Brezzi, L. Donatella Marini, and E. Süli. Discontinuous Galerkin methods for first-order hyperbolic problems. *Mathematical Models and Methods in Applied Sciences*, 14(12):1893–1903, 2004.
- [11] Anne Burbeau, Pierre Sagaut, and Charles-Henri Bruneau. A problem-independent limiter for high-order Runge–Kutta discontinuous Galerkin methods. *Journal of Computational Physics*, 169(1):111–150, 2001.
- [12] Claudio Canuto, M. Yousuff Hussaini, Alfio Quarteroni, and Thomas A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1988.
- [13] Claudio Canuto and Alfio Quarteroni. Approximation results for orthogonal polynomials in Sobolev spaces. *Mathematics of Computation*, 38(157):67–86, 1982.
- [14] Mark H. Carpenter, Travis C. Fisher, Eric J. Nielsen, and Steven H. Frankel. Entropy stable spectral collocation schemes for the Navier–Stokes equations: Discontinuous interfaces. *SIAM Journal on Scientific Computing*, 36(5):B835–B867, 2014.
- [15] Mark H. Carpenter and David Gottlieb. Spectral methods on arbitrary grids. *Journal of Computational Physics*, 129:74–86, 1996.
- [16] Mark H. Carpenter, David Gottlieb, and Chi-Wang Shu. On the conservation and convergence to weak solutions of global schemes. *Journal of Scientific Computing*, 18(1):111–132, 2003.
- [17] Patrice Castonguay, Peter E. Vincent, and Antony Jameson. A new class of high-order energy stable flux reconstruction schemes for triangular elements. *Journal of Scientific Computing*, 51(1):224–256, 2012.
- [18] Patrice Castonguay, David M. Williams, Peter E. Vincent, and Antony Jameson. Energy stable flux reconstruction schemes for advection–diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 267:400–417, 2013.
- [19] Guy Chavent and Bernardo Cockburn. The local projection  $p^0 - p^1$  discontinuous Galerkin finite element method for scalar conservation laws. *RAIRO-Modélisation Mathématique et Analyse Numérique*, 23(4):565–592, 1989.

- [20] Guy Chavent and G. Salzano. A finite-element method for the 1-D water flooding problem with gravity. *Journal of Computational Physics*, 45(3):307–344, 1982.
- [21] Tianheng Chen and Chi-Wang Shu. Entropy stable high order discontinuous Galerkin methods with suitable quadrature rules for hyperbolic conservation laws. *Journal of Computational Physics*, 345:427–461, 2017.
- [22] Philippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*, volume 40 of Classics in Applied Mathematics. SIAM, 2002.
- [23] Bernardo Cockburn. Discontinuous Galerkin methods for convection-dominated problems. In: *High-order Methods for Computational Physics*, pages 69–224. Springer, 1999.
- [24] Bernardo Cockburn and Johnny Guzmán. Error estimates for the Runge–Kutta discontinuous Galerkin method for the transport equation with discontinuous initial data. *SIAM Journal on Numerical Analysis*, 46(3):1364–1398, 2008.
- [25] Bernardo Cockburn, Suchung Hou, and Chi-Wang Shu. The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV. The multidimensional case. *Mathematics of Computation*, 54(190):545–581, 1990.
- [26] Bernardo Cockburn, San-Yih Lin, and Chi-Wang Shu. TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, 1989.
- [27] Bernardo Cockburn and Chi-Wang Shu. TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework. *Mathematics of Computation*, 52(186):411–435, 1989.
- [28] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta local projection  $p^1$  discontinuous Galerkin finite element method for scalar conservation laws. *RAIRO-Modélisation Mathématique et Analyse Numérique*, 25(3):337–361, 1991.
- [29] Bernardo Cockburn and Chi-Wang Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, 1998.
- [30] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. *Journal of Computational Physics*, 141(2):199–224, 1998.
- [31] Bernardo Cockburn and Chi-Wang Shu. Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, 2001.
- [32] Philip J. Davis and Philip Rabinowitz. *Numerical Integration*. Blaisdell Publishing Company, 1967.
- [33] Michael Dumbser and Martin Käser. Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems. *Journal of Computational Physics*, 221(2):693–723, 2007.

- [34] Alexandre Ern and Jean-Luc Guermond. Discontinuous Galerkin methods for Friedrichs' systems. I. General theory. *SIAM Journal on Numerical Analysis*, 44(2):753–778, 2006.
- [35] Alexandre Ern and Jean-Luc Guermond. *Theory and Practice of Finite Elements*, volume 159. Springer Science+Business Media, 2013.
- [36] Jinzhi Fan and Hua Li. *A Shock Capturing Technique Using an Adaptive Filter for CPR Schemes*. Technical report, 2015.
- [37] Travic C. Fisher and Mark H. Carpenter. High-order entropy stable finite difference schemes for nonlinear conservation laws: Finite domains. *Journal of Computational Physics*, 252:518–557, 2013.
- [38] Daniele Funaro and David Gottlieb. A new method of imposing boundary conditions in pseudospectral approximations of hyperbolic equations. *Mathematics of Computation*, 51(184):599–613, 1988.
- [39] Daniele Funaro and David Gottlieb. Convergence results for pseudospectral approximations of hyperbolic systems by a penalty-type boundary treatment. *Mathematics of Computation*, 57(196):585–596, 1991.
- [40] Gregor J. Gassner. A skew-symmetric discontinuous Galerkin spectral element discretization and its relation to SBP-SAT finite difference methods. *SIAM Journal of Scientific Computing*, 35(3):A1233–A1253, 2013.
- [41] Gene H. Golub and John H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation*, 23(106):221–230, 1969.
- [42] David Gottlieb and Jan S. Hesthaven. Spectral methods for hyperbolic problems. *Journal of Computational and Applied Mathematics*, 128(1):83–131, 2001.
- [43] David Gottlieb and Steven A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*, volume 26 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1977.
- [44] David Gottlieb and Eitan Tadmor. The CFL condition for spectral approximations to hyperbolic initial-boundary value problems. *Mathematics of Computation*, 56(194):565–588, 1991.
- [45] Jean-Luc Guermond, Richard Pasquetti, and Bojan Popov. Entropy viscosity method for nonlinear conservation laws. *Journal of Computational Physics*, 230(11):4248–4267, 2011.
- [46] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. *Time Dependent Problems and Difference Methods*, volume 24 of Pure and Applied Mathematics. John Wiley & Sons, 1995.
- [47] Jan S. Hesthaven. A stable penalty method for the compressible Navier–Stokes equations: II. One-dimensional domain decomposition schemes. *SIAM Journal on Scientific Computing*, 18(3):658–685, 1997.
- [48] Jan S. Hesthaven. From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex. *SIAM Journal on Numerical Analysis*, 35(2):655–676, 1998.

- [49] Jan S. Hesthaven. A stable penalty method for the compressible Navier–Stokes equations: III. Multidimensional domain decomposition schemes. *SIAM Journal on Scientific Computing*, 20(1):62–93, 1998.
- [50] Jan S. Hesthaven. Spectral penalty methods. *Applied Numerical Mathematics*, 33(1):23–41, 2000.
- [51] Jan S. Hesthaven and David Gottlieb. Stable spectral methods for conservation laws on triangles with unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 175(3):361–381, 1999.
- [52] Jan S. Hesthaven and David Gottlieb. A stable penalty method for the compressible Navier–Stokes equations: I. Open boundary conditions. *SIAM Journal on Scientific Computing*, 17(3):579–612, 1996.
- [53] Jan S. Hesthaven, Sigal Gottlieb, and David Gottlieb. *Spectral Methods for Time-Dependent Problems*, volume 21 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2007.
- [54] Jan S. Hesthaven and Robert Kirby. Filtering in Legendre spectral methods. *Mathematics of Computation*, 77(263):1425–1452, 2008.
- [55] Jan S. Hesthaven and Chun-Hao Teng. Stable spectral methods on tetrahedral elements. *SIAM Journal on Scientific Computing*, 21(6):2352–2380, 2000.
- [56] Jan S. Hesthaven and Tim Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*. Springer Science+Business Media, 2007.
- [57] Hussein Hoteit, Ph. Ackerer, Robert Mosé, Jocelyne Erhel, and Bernard Philippe. New two-dimensional slope limiters for discontinuous Galerkin methods on arbitrary meshes. *International Journal for Numerical Methods in Engineering*, 61(14):2566–2593, 2004.
- [58] Songming Hou and Xu-Dong Liu. Solutions of multi-dimensional hyperbolic systems of conservation laws by square entropy condition satisfying discontinuous Galerkin method. *Journal of Scientific Computing*, 31:127–151, 2007.
- [59] Changqing Hu and Chi-Wang Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150(1):97–127, 1999.
- [60] Fang Q. Hu and Harold L. Atkins. Eigensolution analysis of the discontinuous Galerkin method with nonuniform grids: I. One space dimension. *Journal of Computational Physics*, 182(2):516–545, 2002.
- [61] Fang Q. Hu and Harold L. Atkins. Two-dimensional wave analysis of the discontinuous Galerkin method with non-uniform grids and boundary conditions. In: *Proceedings of the 8th AIAA/CEAS Aeroacoustics Conference and Exhibit*, 2002, paper AIAA 2002-2514.
- [62] Fang Q. Hu, M. Y. Hussaini, and Patrick Rasetarinera. An analysis of the discontinuous Galerkin method for wave propagation problems. *Journal of Computational Physics*, 151(2):921–946, 1999.
- [63] Thomas J. R. Hughes. *The finite element method: Linear static and dynamic finite element analysis*. Courier Corporation, 2012.

- [64] Hung T. Huynh. A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods. In: *Proceedings of the 18th AIAA Computational Fluid Dynamics Conference*, 2007, paper AIAA 2007-4079.
- [65] Antony Jameson. A proof of the stability of the spectral difference method for all orders of accuracy. *Journal of Scientific Computing*, 45(1-3):348–358, 2010.
- [66] Antony Jameson, Peter E. Vincent, and Patrice Castonguay. On the nonlinear stability of flux reconstruction schemes. *Journal of Scientific Computing*, 50(2):434–445, 2012.
- [67] Guang Shan Jiang and Chi-Wang Shu. On a cell entropy inequality for discontinuous Galerkin methods. *Mathematics of Computation*, 62(206):531–538, 1994.
- [68] Claes Johnson and Juhani Pitkäranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Mathematics of Computation*, 46(173):1–26, 1986.
- [69] George Karniadakis and Spencer Sherwin. *Spectral/hp Element Methods for Computational Fluid Dynamics*. Oxford University Press, 2013.
- [70] Andreas Klöckner, Tim Warburton, and Jan S. Hesthaven. Viscous shock capturing in a time-explicit discontinuous Galerkin method. *Mathematical Modelling of Natural Phenomena*, 6(3):57–83, 2011.
- [71] David A. Kopriva. A conservative staggered-grid Chebyshev multidomain method for compressible flows. II: A semi-structured method. *Journal of Computational Physics*, 128(2):475–488, 1996.
- [72] David A. Kopriva. A staggered-grid multidomain spectral method for the compressible Navier–Stokes equations. *Journal of Computational Physics*, 143(1):125–158, 1998.
- [73] David A. Kopriva and John H. Kolias. A conservative staggered-grid Chebyshev multidomain method for compressible flows. *Journal of Computational Physics*, 125(1):244–261, 1996.
- [74] Lilia Krivodonova, J. Xin, J.-F. Remacle, Nicolas Chevaugeon, and Joseph E. Flaherty. Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Applied Numerical Mathematics*, 48(3):323–338, 2004.
- [75] Dmitri Kuzmin. A vertex-based hierarchical slope limiter for  $p$ -adaptive discontinuous Galerkin methods. *Journal of Computational and Applied Mathematics*, 233(12):3077–3085, 2010.
- [76] Peter D. Lax. Accuracy and resolution in the computation of solutions of linear and nonlinear equations. Springer Collected Works in Mathematics, *Selected Papers Volume I*, pages 184–194, 2005.
- [77] Pierre Lesaint and Pierre-Arnaud Raviart. On a finite element method for solving the neutron transport equation. *Mathematical Aspects of Finite Elements in Partial Differential Equations*, 33:89–123, 1974.

- [78] Ben Q. Li. *Discontinuous Finite Elements in Fluid Dynamics and Heat Transfer*. Springer Science+Business Media, 2006.
- [79] Wanai Li, Yu-Xin Ren, Guodong Lei, and Hong Luo. The multi-dimensional limiters for solving hyperbolic conservation laws on unstructured grids. *Journal of Computational Physics*, 230(21):7775–7795, 2011.
- [80] Yen Liu, Marcel Vinokur, and Zhi Jian Wang. Spectral difference method for unstructured grids I: Basic formulation. *Journal of Computational Physics*, 216(2):780–801, 2006.
- [81] Yen Liu, Marcel Vinokur, and Zhi Jian Wang. Spectral (finite) volume method for conservation laws on unstructured grids V: Extension to three-dimensional systems. *Journal of Computational Physics*, 212(2):454–472, 2006.
- [82] Catherine Mavriplis. Adaptive mesh strategies for the spectral element method. *Computer Methods in Applied Mechanics and Engineering*, 116(1):77–86, 1994.
- [83] Georg May and Antony Jameson. A spectral difference method for the Euler and Navier–Stokes equations on unstructured meshes. In: *Proceedings of the 44th Aerospace Sciences Meeting and Exhibit*, 2006, paper AIAA 2006-304.
- [84] Michael S. Mock and Peter D. Lax. The computation of discontinuous solutions of linear hyperbolic equations. *Communications on Pure and Applied Mathematics*, 31(4):423–430, 1978.
- [85] Stanley Osher. Riemann solvers, the entropy condition, and difference. *SIAM Journal on Numerical Analysis*, 21(2):217–235, 1984.
- [86] Per-Olof Persson and Jaime Peraire. Sub-cell shock capturing for discontinuous Galerkin methods. In: *Proceedings of the 44th Aerospace Sciences Meeting and Exhibit*, 2006, paper AIAA 2006-112.
- [87] Todd E. Peterson. A note on the convergence of the discontinuous Galerkin method for a scalar hyperbolic equation. *SIAM Journal on Numerical Analysis*, 28(1):133–140, 1991.
- [88] Jianxian Qiu and Chi-Wang Shu. Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method: One-dimensional case. *Journal of Computational Physics*, 193(1):115–135, 2004.
- [89] Jianxian Qiu and Chi-Wang Shu. A comparison of troubled-cell indicators for Runge–Kutta discontinuous Galerkin methods using weighted essentially nonoscillatory limiters. *SIAM Journal on Scientific Computing*, 27(3):995–1013, 2005.
- [90] Jianxian Qiu and Chi-Wang Shu. Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method II: Two dimensional case. *Computers & Fluids*, 34(6):642–663, 2005.
- [91] Jianxian Qiu and Chi-Wang Shu. Runge–Kutta discontinuous Galerkin method using WENO limiters. *SIAM Journal on Scientific Computing*, 26(3):907–929, 2005.

- [92] Wm H. Reed and T. R. Hill. *Triangular Mesh Methods for the Neutron Transport Equation*. Technical Report LA-UR-73-479, Los Alamos National Lab, 1973.
- [93] Gerard R. Richter. An optimal-order error estimate for the discontinuous Galerkin method. *Mathematics of Computation*, 50(181):75–88, 1988.
- [94] Béatrice Rivière. *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation*, volume 35 of *Frontiers in Applied Mathematics*. SIAM, 2008.
- [95] Christoph Schwab. *p-and hp-Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics*. Oxford University Press, 1998.
- [96] Spencer Sherwin. Dispersion analysis of the continuous and discontinuous Galerkin formulations. In: *Discontinuous Galerkin Methods*, pages 425–431. Springer, 2000.
- [97] Jing Shi, Changqing Hu, and Chi-Wang Shu. A technique of treating negative weights in WENO schemes. *Journal of Computational Physics*, 175(1):108–127, 2002.
- [98] Chi-Wang Shu. TVB uniformly high-order schemes for conservation laws. *Mathematics of Computation*, 49(179):105–121, 1987.
- [99] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 83(1):32–78, 1989.
- [100] Gilbert Strang and George J. Fix. *An Analysis of the Finite Element Method*, Series in Automatic Computation, volume 212. Prentice-Hall, 1973.
- [101] Yuzhi Sun, Zhi Jian Wang, and Yen Liu. Spectral (finite) volume method for conservation laws on unstructured grids VI: Extension to viscous flow. *Journal of Computational Physics*, 215(1):41–58, 2006.
- [102] Yuzhi Sun, Zhi Jian Wang, and Yen Liu. High-order multidomain spectral difference method for the Navier-Stokes equations on unstructured hexahedral grids. *Communications in Computational Physics*, 2(2):310–333, 2007.
- [103] Gabor Szego. *Orthogonal Polynomials*, volume 23. American Mathematical Society, Colloquim Publications, 1939.
- [104] Eitan Tadmor. The exponential accuracy of Fourier and Chebyshev differencing methods. *SIAM Journal on Numerical Analysis*, 23(1):1–10, 1986.
- [105] Tayfun E. Tezduyar. *Stabilized Finite Element Formulations for Incompressible Flow Computations*. Academic Press, 1992.
- [106] Shuangzhang Tu and Shahrouz Aliabadi. A slope limiting procedure in discontinuous Galerkin finite element method for gasdynamics applications. *International Journal of Numerical Analysis and Modeling.* 2(2):163–178, 2005.
- [107] Kris Van den Abeele, Chris Lacor, and Zhi Jian Wang. On the connection between the spectral volume and the spectral difference method. *Journal of Computational Physics*, 227(2):877–885, 2007.

- [108] Kris Van den Abeele, Chris Lacor, and Zhi Jian Wang. On the stability and accuracy of the spectral difference method. *Journal of Scientific Computing*, 37(2):162–188, 2008.
- [109] Hervé Vandeven. Family of spectral filters for discontinuous problems. *Journal of Scientific Computing*, 6(2):159–192, 1991.
- [110] Peter E. Vincent, Patrice Castonguay, and Antony Jameson. Insights from von Neumann analysis of high-order flux reconstruction schemes. *Journal of Computational Physics*, 230(22):8134–8154, 2011.
- [111] Peter E. Vincent, Patrice Castonguay, and Antony Jameson. A new class of high-order energy stable flux reconstruction schemes. *Journal of Scientific Computing*, 47(1):50–72, 2011.
- [112] Zhi Jian Wang. Spectral (finite) volume method for conservation laws on unstructured grids: Basic formulation. *Journal of Computational Physics*, 178(1):210–251, 2002.
- [113] Zhi Jian Wang and Yen Liu. Spectral (finite) volume method for conservation laws on unstructured grids: II. Extension to two-dimensional scalar equation. *Journal of Computational Physics*, 179(2):665–697, 2002.
- [114] Zhi Jian Wang and Yen Liu. Spectral (finite) volume method for conservation laws on unstructured grids III: One dimensional systems and partition optimization. *Journal of Scientific Computing*, 20(1):137–157, 2004.
- [115] Zhi Jian Wang, Yen Liu, Georg May, and Antony Jameson. Spectral difference method for unstructured grids II: Extension to the Euler equations. *Journal of Scientific Computing*, 32(1):45–71, 2007.
- [116] Zhi Jian Wang, Laiping Zhang, and Yen Liu. Spectral (finite) volume method for conservation laws on unstructured grids IV: Extension to two-dimensional systems. *Journal of Computational Physics*, 194(2):716–741, 2004.
- [117] T. Warburton. A low-storage curvilinear discontinuous Galerkin method for wave problems. *SIAM Journal on Scientific Computing*, 35(4):A1987–A2012, 2013.
- [118] Wikipedia contributors. *Maxwell's equations*. Wikipedia, The Free Encyclopedia, [https://en.wikipedia.org/wiki/Maxwell%27s\\_equations](https://en.wikipedia.org/wiki/Maxwell%27s_equations) (accessed August 2016).
- [119] David M. Williams, Patrice Castonguay, Peter E. Vincent, and Antony Jameson. Energy stable flux reconstruction schemes for advection–diffusion problems on triangles. *Journal of Computational Physics*, 250:53–76, 2013.
- [120] Freddie D. Witherden, Antony M. Farrington, and Peter E. Vincent. Pyfr: An open source framework for solving advection–diffusion type problems on streaming architectures using the flux reconstruction approach. *Computer Physics Communications*, 185(11):3028–3040, 2014.
- [121] Michael Yang and Zhi-Jian Wang. A parameter-free generalized moment limiter for high-order methods on unstructured grids. *Advanced in Mathematical Modeling and Mechanics*, 1(4):451–480, 2009.

- 
- [122] Mengping Zhang and Chi-Wang Shu. An analysis of and a comparison between the discontinuous Galerkin and the spectral finite volume methods. *Computers & Fluids*, 34(4):581–592, 2005.
  - [123] Qiang Zhang and Chi-Wang Shu. Error estimates to smooth solutions of Runge–Kutta discontinuous Galerkin methods for scalar conservation laws. *SIAM Journal on Numerical Analysis*, 42(2):641–666, 2004.
  - [124] Qiang Zhang and Chi-Wang Shu. Error estimates to smooth solutions of Runge–Kutta discontinuous Galerkin method for symmetrizable systems of conservation laws. *SIAM Journal on Numerical Analysis*, 44(4):1703–1720, 2006.
  - [125] Xiangxiong Zhang and Chi-Wang Shu. On maximum-principle-satisfying high order schemes for scalar conservation laws. *Journal of Computational Physics*, 229(9):3091–3120, 2010.
  - [126] Xiangxiong Zhang and Chi-Wang Shu. On positivity-preserving high order discontinuous Galerkin schemes for compressible euler equations on rectangular meshes. *Journal of Computational Physics*, 229(23):8918–8934, 2010.
  - [127] Xiangxiong Zhang and Chi-Wang Shu. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: Survey and new developments. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 467:2752–2776, 2011.
  - [128] Xinghui Zhong and Chi-Wang Shu. A simple weighted essentially nonoscillatory limiter for Runge–Kutta discontinuous Galerkin methods. *Journal of Computational Physics*, 232(1):397–415, 2013.
  - [129] Jun Zhu, Xinghui Zhong, Chi-Wang Shu, and Jianxian Qiu. Runge–Kutta discontinuous Galerkin method using a new type of WENO limiters on unstructured meshes. *Journal of Computational Physics*, 248:200–220, 2013.
  - [130] Olgierd Cecil Zienkiewicz and Robert Leroy Taylor. *The Finite Element Method: Solid Mechanics*, volume 2. Butterworth-Heinemann, 2000.
  - [131] Olgierd Cecil Zienkiewicz, Robert Leroy Taylor, Olgierd Cecil Zienkiewicz, and Robert Lee Taylor. *The Finite Element Method*, volume 3. McGraw-Hill, 1977.
  - [132] Jens Zudrop and Jan S. Hesthaven. Accuracy of high order and spectral methods for hyperbolic conservation laws with discontinuous solutions. *SIAM Journal of Numerical Analysis*, 53(4):1857–1875, 2015.

## Chapter 13

# Spectral methods

This final chapter is dedicated to methods that take the notion of high-order to the extreme, and rely on all available information to represent the solution. Such methods, known as spectral methods, are global methods and are, in spirit, close to finite difference methods, although with distinct differences, since the notion of numerical fluxes is abandoned.

Although the development of spectral methods for smooth problems is a mature field, see e.g., [24, 6, 16, 3, 34, 51], their application to solve conservation laws is less developed and remains an area of active research. As has been a main theme throughout this text, the main challenge is the spontaneous formation of discontinuous solutions, and the need to balance high-order accuracy with the control of numerical oscillations.

In order to keep the exposition focused, we restrict the discussion to Fourier spectral methods. While this limits the scope to problems with periodic solutions, it allows us to develop a solid foundation for some of the results we have quoted previously, e.g., the impact of filtering on the accuracy, and the development of postprocessing techniques that enable the reduction and elimination of the Gibbs phenomenon.

It should be emphasized that by using orthogonal polynomials rather than Fourier series, many of the results discussed here can be extended to problems with nonperiodic solutions. For details we refer to some of the reference texts devoted to this topic [6, 16, 3, 34, 51], including texts with an applied focus [10, 7, 42].

We consider the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad x \in [0, L],$$

and assume periodic boundary conditions and a suitable initial condition. In a Fourier spectral method one assumes the existence of a global solution of the form

$$u(x, t) \simeq u_b(x, t) = \sum_{n=-N}^N \hat{u}_n(t) \exp\left(i 2\pi n \frac{x}{L}\right),$$

where  $i = \sqrt{-1}$ . We form the residual

$$R_b(x, t) = \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x},$$

and define how this vanishes. As we shall see, different families of methods with slightly different properties arise from this last step.

## 13.1 • Fourier modes and nodes

Following the approach of Chapter 12, we first seek to understand the relation between the accuracy of the approximation and the regularity of the solution. Since we restrict our attention to problems with some degree of periodicity, it is natural to use trigonometric polynomials, i.e., Fourier series, to represent the unknown solution. For the sake of simplicity and without loss of generality, we consider functions defined on  $[0, 2\pi]$ .

Let us recall that for periodic functions  $u \in L^2([0, 2\pi])$ , the continuous Fourier expansion

$$u(x) = \sum_{n=-\infty}^{\infty} \hat{u}_n \exp(inx), \quad \hat{u}_n = \frac{1}{2\pi} \int_0^{2\pi} u(x) \exp(-inx) dx,$$

exists and converges almost everywhere [8]. Here  $\hat{u}_n$  are the expansion coefficients. For any  $u \in H^p([0, 2\pi])$  we recover

$$u^{(m)}(x) = \sum_{n=-\infty}^{\infty} (in)^m \hat{u}_n \exp(inx),$$

where  $0 \leq m \leq p$ .

The Sobolev space  $\|\cdot\|_p$  is characterized by the unweighted Sobolev norm

$$\|u\|_p^2 = \sum_{m=0}^p \int_0^{2\pi} \|u^{(m)}(x)\|^2 dx.$$

As we shall see, it is useful to introduce a new norm,  $\|\cdot\|_{W,p}$ , equivalent to  $\|\cdot\|_p$ , defined as [47, 43]

$$\|u\|_{W,p} = \left( \sum_{n=-\infty}^{\infty} (1+|n|)^{2p} |\hat{u}_n|^2 \right)^{1/2}.$$

The associated Sobolev space is indicated with  $W^p([0, 2\pi])$ . We observe that this definition can be extended to include real values of  $p$ .

### 13.1.1 • The continuous Fourier expansion

For a function  $u \in L^2([0, 2\pi])$  the classic continuous series of trigonometric polynomials, known as the Fourier series  $F[u]$ , is defined as

$$F[u] = \hat{a}_0 + \sum_{n=1}^{\infty} \hat{a}_n \cos(nx) + \sum_{n=1}^{\infty} \hat{b}_n \sin(nx), \quad (13.1)$$

where the expansion coefficients are

$$\hat{a}_n = \frac{1}{c_n \pi} \int_0^{2\pi} u(x) \cos(nx) dx, \quad c_n = \begin{cases} 2, & n=0, \\ 1, & n>0, \end{cases}$$

and

$$\hat{b}_n = \frac{1}{\pi} \int_0^{2\pi} u(x) \sin(nx) dx, \quad n > 0.$$

We have the natural inner product and the associated norm

$$(u, v) = \int_0^{2\pi} u(x) \bar{v}(x) dx, \quad \|u\|_2 = \left( \int_0^{2\pi} |u(x)|^2 dx \right)^{1/2},$$

where  $\bar{v}$  refers to the complex conjugate of  $v$ . An equivalent way to express the Fourier series employs the Fourier basis functions

$$\phi_n(x) = \exp(inx),$$

which are orthogonal over  $[0, 2\pi]$ . By introducing the complex coefficients,

$$\hat{u}_n = \begin{cases} \hat{a}_0, & n = 0, \\ (\hat{a}_n - i\hat{b}_n)/2, & n > 0, \\ (\hat{a}_n + i\hat{b}_n)/2, & n < 0, \end{cases} \quad (13.2)$$

the trigonometric series (13.1) can be expressed as

$$F[u] = \sum_{n=-\infty}^{\infty} \hat{u}_n \phi_n(x), \quad (13.3)$$

and the expansion coefficients  $\hat{u}_n$  are obtained by orthogonality as

$$\hat{u}_n = \frac{1}{2\pi} \int_0^{2\pi} u(x) \exp(-inx) dx. \quad (13.4)$$

When  $u(x)$  is a real function then  $\hat{a}_n$  and  $\hat{b}_n$  are real numbers and, consequently,  $\hat{u}_{-n} = \overline{\hat{u}_n}$ . We refer to (13.3)–(13.4) as the continuous Fourier series.

Let us now consider the convergence behavior of the truncated Fourier series

$$u_b(x) = \mathcal{P}_N u(x) = \sum_{n=-N}^N \hat{u}_n \exp(inx), \quad (13.5)$$

where we have introduced the projection operator  $\mathcal{P}_N$ . The convergence behavior of (13.5) has been given a thorough mathematical treatment in the past; see, e.g., [6].

Since  $\mathcal{P}_N u$  is periodic, a sufficient condition for uniform convergence is that  $u$  is periodic and possesses a minimum amount of smoothness [8].

**Theorem 13.1.** *Every periodic function  $u \in C^1([0, 2\pi])$  has a uniformly convergent Fourier series  $\mathcal{P}_N u$ , i.e.,*

$$\|u - \mathcal{P}_N u\|_{\infty} \rightarrow 0 \quad \text{as} \quad N \rightarrow \infty.$$

The condition on smoothness is needed to ensure that

$$\sum_{n=-\infty}^{\infty} |\hat{u}_n| < \infty.$$

A weaker result for convergence in the mean is the following.

**Theorem 13.2.** *Every piecewise continuous function  $u$  permits a Fourier series  $\mathcal{P}_N u$  such that*

$$\|u - \mathcal{P}_N u\|_2 \rightarrow 0 \quad \text{as} \quad N \rightarrow \infty.$$

This implies

$$\begin{aligned} \|u - \mathcal{P}_N u\|_2^2 &= \|u\|_2^2 - (u, \mathcal{P}_N u) - (\mathcal{P}_N u, u) + \|\mathcal{P}_N u\|_2^2 \\ &= \|u\|_2^2 - 2\pi \sum_{n=-N}^N |\hat{u}_n|^2. \end{aligned}$$

Letting  $N$  approach infinity, we recover Parseval's identity:

$$\|u\|_2^2 = 2\pi \sum_{n=-\infty}^{\infty} |\hat{u}_n|^2.$$

This now yields the primary error estimate.

**Theorem 13.3.** *For  $u \in H^p([0, 2\pi])$ , there exists a positive constant  $C$ , independent of  $N$ , such that*

$$\|u - \mathcal{P}_N u\|_2 \leq CN^{-q} \|u^{(q)}\|_2,$$

provided  $0 \leq q \leq p$ .

**Proof.** Using Parseval's identity we recover

$$\|u - \mathcal{P}_N u\|_2^2 = 2\pi \sum_{|n|>N} |\hat{u}_n|^2.$$

Furthermore we have that

$$\sum_{|n|>N} |\hat{u}_n|^2 = \sum_{|n|>N} \frac{1}{n^{2q}} n^{2q} |\hat{u}_n|^2 \leq N^{-2q} \sum_{|n|>N} n^{2q} |\hat{u}_n|^2 \leq N^{-2q} \|u^{(q)}\|_2^2.$$

This establishes the result.  $\square$

If we assume that  $u$  is analytic, then [55]

$$\|u^{(q)}\|_2 \leq C q! \|u\|_2.$$

This allows us to refine the previous results as

$$\|u - \mathcal{P}_N u\|_2 \leq CN^{-q} \|u^{(q)}\|_2 \sim C \frac{q!}{N^q} \|u\|_2 \sim C \frac{q^q e^{-q}}{N^q} \|u\|_2 \sim C e^{-cN} \|u\|_2,$$

where we assume that  $q \propto N$  and make use of Stirling's formula. This highlights the potential for exponentially fast convergence, and offers the motivation for the label of exponentially accurate schemes, often put on spectral methods.

The truncation error depends solely on how fast the expansion coefficients of  $u$  decay which, in turn, depends on the regularity of  $u$  and its periodicity. In fact, if

$u \in C^1([0, 2\pi])$ , integration by parts implies

$$\begin{aligned} 2\pi \hat{u}_n &= \int_0^{2\pi} u(x) \exp(-inx) dx \\ &= \frac{-1}{in} (u(2\pi) - u(0)) + \frac{1}{in} \int_0^{2\pi} u'(x) \exp(-inx) dx, \end{aligned}$$

provided  $n \neq 0$ . If  $u' \in L^2([0, 2\pi])$ , the integral exists and we recover

$$|\hat{u}_n| \propto \frac{1}{n}.$$

If, in addition,  $u$  is periodic the boundary term vanishes and we have

$$|\hat{u}_n| \propto \frac{1}{n^2}.$$

Repeating this line of argument leads to the following theorem.

**Theorem 13.4.** *If  $u \in L^2([0, 2\pi])$  is at least  $m$ -times continuously differentiable in  $[0, 2\pi]$ , i.e.,  $u \in C^m([0, 2\pi])$ , and if*

$$\forall j \in [0, m-2] : u^{(j)}(0) = u^{(j)}(2\pi),$$

*then the Fourier expansion coefficients  $\hat{u}_n$  ( $n \neq 0$ ) decay as*

$$|\hat{u}_n| \propto \frac{1}{n^m}.$$

**Example 13.5.** The function

$$u(x) = 2 \sin\left(\frac{x}{2}\right) \tag{13.6}$$

is infinitely differentiable but its first derivative is not periodic. The expansion coefficients are given as

$$\hat{u}_n = \frac{4}{\pi} \frac{1}{1 - 4n^2}.$$

In agreement with Theorem 13.4, we observe quadratic decay in  $n$ .

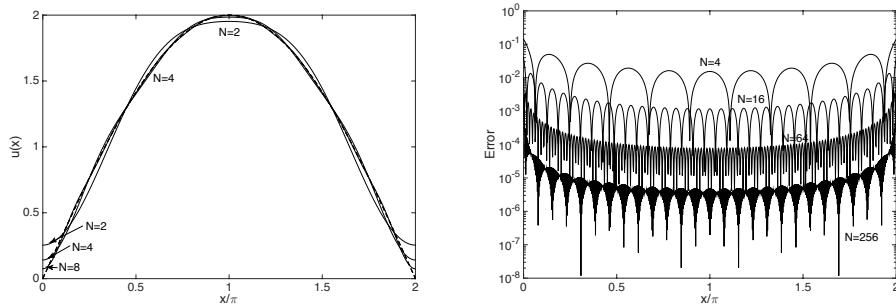
Figure 13.1 shows the truncated Fourier approximation and the pointwise error for increasing values of  $N$ . We observe a nonuniform convergence with quadratic convergence throughout the domain except near the end points where it is only linear. This is associated with the Gibbs phenomenon, discussed in section 8.3. ■

Given the Fourier expansion of  $u$ , we need to recover the expansion coefficients  $\hat{u}_n^{(q)}$  such that

$$\frac{d^q}{dx^q} u(x) = \sum_{n=-\infty}^{\infty} \hat{u}_n^{(q)} \exp(inx).$$

Provided  $u^{(q)} \in C^1([0, 2\pi])$ , orthogonality of the Fourier basis yields

$$\hat{u}_n^{(q)} = (in)^q \hat{u}_n. \tag{13.7}$$



**Figure 13.1.** We show the truncated Fourier approximation of (13.6) for  $N = 2, 4, 8$  (left). The dashed line marks the function being approximated. We also show the pointwise error of the approximation for increasing resolution,  $N = 4, 16, 64, 256$  (right).

The computation of the  $q$ th order differentiation is illustrated in Fourierdx.m.

**Script 13.1. Fourierdx.m: Evaluation of the  $q$ th order Fourier derivative by the Fourier transform.**

---

```
function [dudx] = Fourierdx(u,q);
% function [dudx] = Fourierdx(u,q);
% Purpose: Compute q'th derivative of u using a Fourier spectral approach
ii = sqrt(-1); N = length(u);
uhat = fft(u); nvec = (ii * [0:(N-1)/2 -(N-1)/2:-1]).^q;
dudx = real(ifft(nvec.*uhat));
return
```

---

If  $u \in C^m([0, 2\pi])$  and periodic we have

$$\hat{u}_n \propto \frac{1}{n^{m+1}} \Rightarrow \hat{u}_n^{(q)} \propto \frac{1}{n^{m+1-q}},$$

i.e., for  $q > m$  the expansion fails to converge in agreement with Theorem 13.3.

We now have the more general result [6].

**Theorem 13.6.** Let  $u \in W^p([0, 2\pi])$ . Then for any real  $q$  such that  $0 \leq q \leq p$  there exists a positive constant  $C$ , independent of  $N$ , such that

$$\|u - \mathcal{P}_N u\|_{W,q} \leq C N^{-(p-q)} \|u\|_{W,p}.$$

**Proof.** Using Parseval's identity and orthogonality of the basis functions, we recover

$$\|u - \mathcal{P}_N u\|_{W,q}^2 = \sum_{|n|>N} (1 + |n|)^{2q} |\hat{u}_n|^2.$$

Since  $|n| + 1 \geq N$ , we obtain

$$(1 + |n|)^{2q} = \frac{(1 + |n|)^{2p}}{(1 + |n|)^{2(p-q)}} \leq \frac{(1 + |n|)^{2p}}{N^{2(p-q)}},$$

for any  $q \leq p$ . This yields

$$\|u - \mathcal{P}_N u\|_{W,q}^2 \leq C \sum_{|n|>N} \frac{(1+|n|)^{2p}}{N^{2(p-q)}} |\hat{u}_n|^2 \leq C \frac{\|u\|_{W,p}^2}{N^{2(p-q)}},$$

which establishes the result.  $\square$

An understanding of the pointwise error is obtained in the following result.

**Theorem 13.7.** *For any  $q > 0$ ,  $u \in C^q([0, 2\pi])$ , and periodic up to order  $q$ , there exists a positive constant  $C$ , independent of  $N$ , such that*

$$|u - \mathcal{P}_N u| \leq C N^{-(q-1/2)} \|u^{(q)}\|_2.$$

*Proof.* The triangle inequality and  $|\exp(inx)| \leq 1$  implies

$$|u - \mathcal{P}_N u| = \left| \sum_{|n|>N} \hat{u}_n \exp(inx) \right| \leq \sum_{|n|>N} |\hat{u}_n|.$$

The result follows from the Cauchy–Schwarz inequality:

$$\begin{aligned} \sum_{|n|>N} |\hat{u}_n| &= \sum_{|n|>N} \frac{1}{n^q} n^q |\hat{u}_n| \\ &\leq \left( \sum_{|n|>N} \frac{1}{n^{2q}} \right)^{1/2} \left( \sum_{|n|>N} n^{2q} |\hat{u}_n|^2 \right)^{1/2} \leq N^{-(q-1/2)} \|u^{(q)}\|_2. \end{aligned} \quad \square$$

Finally, we observe that

$$\mathcal{P}_N \frac{du}{dx} = \frac{d}{dx} \mathcal{P}_N u,$$

i.e., truncation and differentiation commute for the continuous Fourier series.

### 13.1.2 ■ The discrete Fourier expansion

The continuous Fourier series require the computation of the expansion coefficients defined through the inner product (13.4). In most situations, this is neither practical, nor possible. As we have already discussed at length in section 12.1 for the Legendre expansion used in the discontinuous Galerkin method, this problem is best overcome by approximating the integrals by quadrature formulas.

Let us introduce a grid

$$x_j = \frac{2\pi}{2N+1} j, \quad j \in [0, \dots, 2N], \quad (13.8)$$

and define the discrete Fourier expansion

$$\mathcal{I}_N u(x) = \sum_{n=-N}^N \tilde{u}_n \exp(inx), \quad \tilde{u}_n = \frac{1}{2N+1} \sum_{j=0}^{2N} u(x_j) \exp(-inx_j), \quad (13.9)$$

in which the discrete expansion coefficients  $\hat{u}_n$  are recovered by approximating (13.4) with the trapezoidal rule. Let us consider

$$\mathcal{I}_N u(x) = \sum_{n=-N}^N \left( \frac{1}{2N+1} \sum_{j=0}^{2N} u(x_j) \exp(-inx_j) \right) \exp(inx).$$

If we exchange the order of the finite summations, we recover

$$\mathcal{I}_N u(x) = \sum_{j=0}^{2N} u(x_j) b_j(x), \quad (13.10)$$

where

$$b_j(x) = \sum_{n=-N}^N \frac{1}{2N+1} \exp\left[in(x-x_j)\right] = \frac{1}{2N+1} \frac{\sin\left(\frac{2N+1}{2}(x-x_j)\right)}{\sin\left(\frac{x-x_j}{2}\right)}. \quad (13.11)$$

It is easily verified that  $b_j(x)$  is a Lagrange polynomial, i.e., the trigonometric polynomial,  $\mathcal{I}_N u$ , interpolates the function,  $u(x)$ , at the quadrature nodes of the trapezoidal formula.

Traditionally, methods with an even number of grid points have been preferred for reasons of computational efficiency and the early availability of fast summation schemes, such as the fast Fourier transform, for specific grids. However, such methods are now generally available for both even and odd number of grid points, provided only that the total number of grid points has a prime factorization based on small primes. We refer to [34] for a discussion of the minor differences between the two representations.

Hence, the discrete Fourier approximation can be defined in one of two equivalent ways. One employs the discrete expansion coefficients, while the other explores the use of the interpolating Lagrange polynomials.

**Example 13.8.** Consider again the function

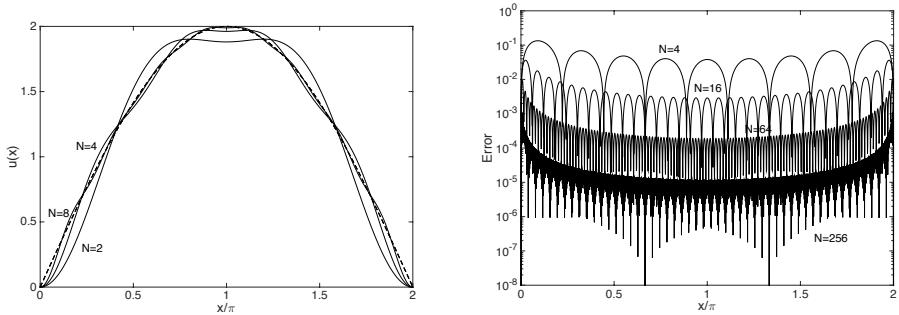
$$u(x) = 2 \sin\left(\frac{x}{2}\right).$$

Recall that this function is infinitely differentiable but nonperiodic in the first derivative. Figure 13.2 shows the truncated discrete Fourier approximation and the pointwise error for increasing  $N$ . From a comparison with Fig. 13.1, it is clear that the qualitative behavior is the same, although there are minor differences in the details of the error, i.e., the interpolatory nature of the discrete Fourier series ensures that the error vanishes at the grid points. ■

When we express the approximation as a series, the computation of derivatives is identical to the one based on the continuous expansion (13.7).

If we instead consider the representation (13.10), an approximation of the derivative at the collocation points  $x_l$  is obtained by differentiating the interpolation directly as

$$\frac{d}{dx} \mathcal{I}_N u(x) \Big|_{x_l} = \sum_{j=0}^{2N} u(x_j) \frac{d}{dx} b_j(x) \Big|_{x_l} = \sum_{j=0}^{2N} D_{lj} u(x_j).$$



**Figure 13.2.** We show the truncated discrete Fourier approximation of (13.6) for  $N = 2, 4, 8$  (left). The dashed line marks the function being approximated. We also show the pointwise error of the approximation for increasing resolution for  $N = 4, 16, 64, 256$  (right).

If we take  $x_l$  as the points used for interpolation, the entries of  $D$  are given as

$$D_{ij} = \begin{cases} \frac{(-1)^{i+j}}{2} \left[ \sin\left(\frac{\pi}{2N+1}(i-j)\right) \right]^{-1}, & i \neq j, \\ 0, & i = j. \end{cases}$$

This operator has a number of properties, i.e., it is a skew-symmetric, circulant, Toeplitz matrix [34]. Furthermore, one easily establishes that

$$D^{(q)} = \mathcal{I}_N \frac{d^q}{dx^q} \mathcal{I}_N = (D)^q,$$

for all values of  $q$ . The definition of this operator is illustrated in FourierD.m.

**Script 13.2. FourierD.m: Differentiation matrix for the  $q$ th order Fourier derivative.**

---

```
function [D] = FourierD(N,q);
% function [D] = FourierD(N);
% Purpose: Initialize q'th order Fourier differentiation matrix
column = [0 (-1).^(1:2*N)./(2*sin(pi/(2*N+1)*(1:2*N)))];
D = toeplitz(column, column([1 (2*N+1):-1:2])).^q;
return
```

---

Finally, let us consider error estimates for the discrete expansion, (13.8)–(13.9). Rather than deriving the estimates of the approximation error directly, we rely on the results for the continuous representation and estimate the difference between the two different expansions, recognized as the aliasing error.

For this, we first need a direct relation between the two sets of expansion coefficients [24, 47, 55]

**Lemma 13.9.** Consider  $u \in W^p([0, 2\pi])$  for  $p > 1/2$ . For  $|n| \leq N$ , we have

$$\tilde{u}_n = \hat{u}_n + \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \hat{u}_{n+2Nm}.$$

**Proof.** By substituting the continuous Fourier expansion into the discrete expansion we have

$$\tilde{u}_n = \frac{1}{2N+1} \sum_{j=0}^{2N} \sum_{l=-\infty}^{\infty} \hat{u}_l \exp(i(l-n)x_j).$$

To interchange the two summations, we need to ensure that

$$\sum_{l=-\infty}^{\infty} |\hat{u}_l| < \infty. \quad (13.12)$$

By the Cauchy-Schwarz inequality, it follows

$$\begin{aligned} \sum_{l=-\infty}^{\infty} |\hat{u}_l| &= \sum_{l=-\infty}^{\infty} (1+|l|)^p \frac{|\hat{u}_l|}{(1+|l|)^p} \\ &\leq \left( \sum_{l=-\infty}^{\infty} (1+|l|)^{2p} |\hat{u}_l|^2 \right)^{1/2} \left( \sum_{l=-\infty}^{\infty} (1+|l|)^{-2p} \right)^{1/2}. \end{aligned}$$

Since  $u \in W^p([0, 2\pi])$  the first part is bounded. Furthermore, the second term converges provided  $p > 1/2$ .

Interchanging the order of summation and using orthogonality of the exponential function yields the result.  $\square$

Let us now consider the behavior of the approximation in the  $L^2([0, 2\pi])$ -space [43].

**Theorem 13.10.** *For any  $u \in W^p([0, 2\pi])$  with  $p > 1/2$ , there exists a positive constant  $C$ , independent of  $N$ , such that*

$$\|u - \mathcal{I}_N u\|_2 \leq CN^{-q} \|u^{(q)}\|_2,$$

provided  $0 \leq q \leq p$ .

**Proof.** We expand  $u$  in the continuous Fourier series and use Parseval's identity to obtain

$$\|u - \mathcal{I}_N u\|_2^2 = \sum_{|n| \leq N} |\hat{u}_n - \tilde{u}_n|^2 + \sum_{|n| > N} |\hat{u}_n|^2.$$

The second term is bounded by Theorem 13.3, while the first term measures the aliasing error.

Using Theorem 13.9, we recover

$$\sum_{|n| \leq N} |\hat{u}_n - \tilde{u}_n|^2 = \sum_{|n| \leq N} \left| \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \hat{u}_{n+2Nm} \right|^2.$$

To estimate this, we note that

$$\left| \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \hat{u}_{n+2Nm} \right|^2 = \left| \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} |n+2Nm|^q \hat{u}_{n+2Nm} \frac{1}{|n+2Nm|^q} \right|^2,$$

which, by the Cauchy–Schwarz inequality, yields

$$\left| \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \hat{u}_{n+2Nm} \right|^2 \leq \left( \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} |n+2Nm|^{2q} |\hat{u}_{n+2Nm}|^2 \right) \left( \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \frac{1}{|n+2Nm|^{2q}} \right).$$

Since  $|n| \leq N$ , the second term is bounded as

$$\sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \frac{1}{|n+2Nm|^{2q}} \leq \frac{2}{N^{2q}} \sum_{m=1}^{\infty} \frac{1}{(2m-1)^{2q}} = C_1 N^{-2q},$$

and we have

$$\begin{aligned} & \sum_{|n| \leq N} \left| \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} \hat{u}_{n+2Nm} \right|^2 \\ & \leq \sum_{|n| \leq N} C_1 N^{-2q} \sum_{\substack{m=-\infty \\ m \neq 0}}^{m=\infty} |n+2mN|^{2q} |\hat{u}_{n+2Nm}|^2 \leq C_2 N^{-2q} \|u^{(q)}\|_2^2. \end{aligned}$$

The total error is bounded as

$$\|u - \mathcal{I}_N u\|_2^2 \leq C N^{-2q} \|u^{(q)}\|_2^2 + C_2 N^{-2q} \|u^{(q)}\|_2^2,$$

establishing the theorem.  $\square$

We observe that if  $u$  has just half a derivative, e.g.,  $u \in C^0([0, 2\pi])$ , the approximation error of the continuous expansion and the discrete expansion are of the same order as the aliasing error.

A similar result can be obtained in the Sobolev spaces [43, 34].

**Theorem 13.11.** *Let  $u \in W^p([0, 2\pi])$  for  $p > 1/2$ . Then, for any real  $q$  such that  $0 \leq q \leq p$ , there exists a positive constant  $C$ , independent of  $N$ , such that*

$$\|u - \mathcal{I}_N u\|_{W,q} \leq C N^{-(p-q)} \|u\|_{W,p}.$$

The projection operator has the special property that it commutes with the linear derivative. However, due to the aliasing, this does not carry over to the discrete expansion. We can estimate the error as

$$\left\| \mathcal{I}_N \frac{du}{dx} - \frac{d}{dx} \mathcal{I}_N \right\| \leq \left\| \frac{du}{dx} - \frac{d}{dx} \mathcal{I}_N \right\| + \left\| \frac{du}{dx} - \mathcal{I}_N \frac{du}{dx} \right\|.$$

For  $u \in W^p([0, 2\pi])$ , Theorem 13.11 yields

$$\left\| \mathcal{I}_N \frac{du}{dx} - \frac{d}{dx} \mathcal{I}_N \right\| \leq N^{-(p-1)} \|u\|_{W,p}.$$

## 13.2 • Fourier spectral methods

Let us now consider the conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \quad x \in [0, L],$$

subject to periodic boundary conditions and an appropriate initial condition. We assume that the solution is expressed as

$$u(x, t) \simeq u_b(x, t) = \sum_{n=-N}^N \tilde{u}_n(t) \exp\left(in2\pi\frac{x}{L}\right),$$

where  $\tilde{u}_n$  represents the continuous or the discrete expansion coefficients, and express the residual:

$$R_b(x, t) = \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x}.$$

By specifying how this is to vanish, different families of methods, with subtle differences between them, arise.

### 13.2.1 • Fourier–Galerkin methods

In the Galerkin approach, we require that the residual is orthogonal to the space spanned by the basis functions,  $\text{span}\{e^{inx}\}_{n=-N}^N$ . If we, for simplicity, assume  $L = 2\pi$ , this yields the scheme

$$\frac{d\tilde{u}_n}{dt} = -\frac{1}{2\pi} \int_0^{2\pi} \frac{\partial f(u_b)}{\partial x} \exp(-inx) dx, \quad \forall n = -N, \dots, N$$

which we can also express as

$$\frac{\partial u_b}{\partial t} + \mathcal{P}_N\left(\frac{\partial f(u_b)}{\partial x}\right) = 0,$$

subject to the initial condition

$$u_b(x, 0) = \mathcal{P}_N u(x, 0).$$

**Example 13.12.** We consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0,$$

subject to the initial condition  $u(x, 0)$ . The Galerkin scheme becomes

$$\frac{\partial u_b}{\partial t} + \sum_{n=-N}^N (in)\tilde{u}_n(t) \exp(inx) = 0,$$

which can be evolved by advancing the  $2N + 1$  decoupled equations:

$$\frac{d\tilde{u}_n(t)}{dt} + in\tilde{u}_n(t) = 0 \quad \forall n = -N, \dots, N.$$

In this particular case, these equations can be solved analytically, and we recover the exact solution:

$$u_b(x, t) = \sum_{n=-N}^N \tilde{u}_n(0) \exp(in(x-t)) = u_b(x-t, 0),$$

where  $u_b(x, 0) = \mathcal{P}_N u(x, 0)$ . Hence, if the initial condition can be expressed exactly in the Fourier basis, the Fourier-Galerkin method solves the linear wave problem exactly. ■

We recall the property that

$$\mathcal{P}_N \left( \frac{\partial f(u_b)}{\partial x} \right) = \frac{\partial \mathcal{P}_N f(u_b)}{\partial x}.$$

This is what allows the exact solution of the linear wave equation. This carries over to all linear, constant-coefficient problems. If we instead consider a nonlinear problem, the situation is more complicated.

**Example 13.13.** Consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to the initial condition  $u(x, 0)$ . The Galerkin scheme becomes

$$\frac{\partial u_b}{\partial t} + \frac{\partial \mathcal{P}_N(u_b)^2}{\partial x} = 0.$$

To recover a scheme for the  $2N + 1$  unknowns, we need to evaluate

$$\mathcal{P}_N(u_b)^2 = \sum_{n=-N}^N \tilde{a}_n(t) \exp(inx), \quad \tilde{a}_n(t) = \frac{1}{2\pi} \int_0^{2\pi} (u_b)^2 \exp(-inx) dx.$$

Since the nonlinearity is of polynomial type, we could apply a quadrature rule of sufficiently high order to evaluate the integral exactly. However, in this particular case, we can evaluate the term directly since

$$(u_b)^2 = \sum_{n=-N}^N \sum_{m=-N}^N \tilde{u}_n \tilde{u}_m \exp(i(n+m)x) = \sum_{n=-N}^N \tilde{a}_n \exp(inx),$$

where

$$\tilde{a}_n = \sum_{m=-N+n}^{N+n} \tilde{u}_n \tilde{u}_{m-n}$$

is the result of a convolution. We note that the cost of evaluating this term is  $(2N)^2$ , i.e., it quickly becomes prohibitive. ■

While it remains possible to develop a Galerkin scheme for Burgers' equation and other types of equations with polynomial nonlinearities, one can easily identify problems where this is no longer possible, e.g., if the nonlinear flux is a rational function or even something as simple as  $f(u) = \exp(u)$ .

The stability of the Fourier–Galerkin method is closely related to the wellposedness of the continuous problem. Consider [43, 33]

$$\frac{\partial u}{\partial t} = \mathcal{L}u, \quad (13.13)$$

where  $u$  is assumed periodic and a suitable initial condition is given.

**Definition 13.14.** *The operator  $\mathcal{L}$  is semibounded if there exists a constant  $\alpha \geq 0$  such that*

$$(u, (\mathcal{L} + \mathcal{L}^*)u) \leq \alpha \|u\|_2^2,$$

for all  $u$  in a suitable Hilbert space. Here  $\mathcal{L}^*$  is the adjoint of  $\mathcal{L}$  with respect to the inner product

$$(\mathcal{L}u, v) = (u, \mathcal{L}^*v).$$

By considering

$$\frac{d}{dt} \|u\|_2^2 = (\mathcal{L}u, u) + (u, \mathcal{L}u) = (u, \mathcal{L}^*u) + (u, \mathcal{L}u) \leq \alpha \|u\|_2^2,$$

we realize that if the operator is semibounded, the problem is well posed.

**Example 13.15.** Consider the operator

$$\mathcal{L} = a(x) \frac{\partial}{\partial x},$$

where  $a$  is continuous, real, and periodic. The adjoint operator is obtained as

$$(\mathcal{L}u, v) = \int_0^{2\pi} a(x) \frac{\partial u}{\partial x} v \, dx = - \int_0^{2\pi} u \frac{\partial}{\partial x} (a(x)v) \, dx = \left( u, \left[ -a(x) \frac{\partial}{\partial x} - a'(x) \right] v \right)$$

such that

$$\mathcal{L}^* = -a(x) \frac{\partial}{\partial x} - a'(x).$$

Now consider

$$\mathcal{L} + \mathcal{L}^* = -a'(x) \leq \alpha$$

or, alternatively,

$$|a'(x)| \leq A.$$

This ensures semiboundedness and, thus, wellposedness of the problem. ■

We can now state the general result.

**Theorem 13.16.** *If (13.13) is semibounded, then the Fourier–Galerkin method is stable.*

*Proof.* As a first step, observe that

$$(u, \mathcal{P}_N v) = (\mathcal{P}_N u, \mathcal{P}_N v) + ((I - \mathcal{P}_N)u, \mathcal{P}_N v) = (\mathcal{P}_N u, \mathcal{P}_N v),$$

where the last equality follows from Galerkin orthogonality. Similarly, we obtain that

$$(\mathcal{P}_N u, v) = (\mathcal{P}_N u, \mathcal{P}_N v),$$

which implies that  $\mathcal{P}_N = \mathcal{P}_N^*$ . We now consider the Fourier–Galerkin scheme

$$\frac{\partial u_b}{\partial t} = \mathcal{P}_N \mathcal{L} u_b = \mathcal{P}_N \mathcal{L} \mathcal{P}_N u_b = \mathcal{L}_N u_b,$$

since  $\mathcal{P}_N u_b = u_b$ . Consequently

$$\mathcal{L}_N + \mathcal{L}_N^* = \mathcal{P}_N \mathcal{L} \mathcal{P}_N + \mathcal{P}_N \mathcal{L}^* \mathcal{P}_N = \mathcal{P}_N (\mathcal{L} + \mathcal{L}^*) \mathcal{P}_N \leq 2\alpha \mathcal{P}_N,$$

from which stability follows, provided that  $\alpha$  exists. This is ensured since the operator is semibounded.  $\square$

Hence, in order to understand stability of Fourier–Galerkin methods it suffices to understand the question of wellposedness of the continuous problem. Nevertheless, as already discussed at some length, the construction of Fourier–Galerkin methods is complex and, for many nonlinear problems, not possible without the introduction of additional sources of errors. In such cases, the above result does not apply.

### 13.2.2 • Fourier collocation methods

In order to overcome the complexities associated with the evaluation of the inner products in the formulation of the Fourier–Galerkin scheme, we can modify the statement on how the residual must vanish. Let us introduce  $2N+1$  distinct collocation points  $y_j$  and require that the residual vanishes in a pointwise sense:

$$R_b(y_j, t) = \left( \frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x} \right) \Big|_{y_j} = 0.$$

This yields  $2N+1$  equations for the  $2N+1$  unknowns. Although there are no immediate restrictions on how the collocation points  $y_j$  are chosen, the stability of the scheme is, to some extent, impacted by this choice [34]. Let us consider the most natural choice

$$x_j = y_j = \frac{2\pi}{2N+1} j, \quad j = 0, \dots, 2N,$$

i.e., the collocation points coincide with the interpolation points on which we base the discrete expansion. If we assume that the solution is expressed as

$$u(x, t) \simeq u_b(x, t) = \sum_{n=-N}^N \tilde{u}_n(t) \exp\left(i n 2\pi \frac{x}{L}\right) = \sum_{j=0}^{2N} u_b(x_j, t) h_j(x),$$

where  $h_j(x)$  is the Lagrange polynomial defined by  $x_j$ , defined in (13.11), we recover the collocation scheme

$$\frac{\partial u_b}{\partial t} \Big|_{x_j} + \mathcal{I}_N \frac{\partial \mathcal{I}_N f(u_b)}{\partial x} = 0.$$

**Example 13.17.** Consider the linear wave equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0,$$

subject to the initial condition  $u_0(x)$ . The collocation scheme is given as

$$\frac{\partial u_h}{\partial t} \Big|_{x_j} + \mathcal{I}_N \frac{\partial u_h}{\partial x} = 0,$$

which we can likewise express as

$$\frac{d}{dt} u(t) + \mathbf{D}u(t) = 0,$$

where we define  $u(t) = [u_h(x_0, t), \dots, u_h(x_{2N}, t)]^T$  and the initial condition is given as  $u_h(0) = u_0(x)$ . In contrast to the Fourier–Galerkin scheme in Ex. 13.12, this does not allow an exact solution of the wave problem due to aliasing errors, discussed in subsection 13.1.2. ■

While the Fourier–Galerkin scheme allows for an exact solution of linear problems, the formulation for nonlinear problems is more complex, as illustrated in Ex. 13.13. However, for the collocation formulation, there is no essential difference between the linear and the nonlinear case.

**Example 13.18.** Consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0,$$

subject to the initial condition  $u_0(x)$ . The collocation scheme is given by

$$\frac{\partial u_h}{\partial t} \Big|_{x_j} + \mathcal{I}_N \frac{\partial \mathcal{I}_N(u_h)^2}{\partial x} = 0,$$

which we can express as

$$\frac{d}{dt} u(t) + \mathbf{D}u^2(t) = 0.$$

Here  $u^2$  refers to the interpolation of the pointwise evaluation of the nonlinearity. While there is no fundamental difference between the formulation of collocation schemes for linear and the nonlinear problems, the interpolation of the nonlinear term may introduce aliasing errors which, as we discussed in subsection 12.2.2, can drive a weak instability. ■

While the issue of stability for the Fourier–Galerkin method is closely related to the wellposedness of the continuous problem, this result does not carry over to the Fourier collocation method. In particular, we can no longer rely on the property of semiboundedness to ensure stability.

There are essentially two different ways, both centered on the use of energy methods, to establish stability of collocation methods. We consider the discrete inner product

and the associated energy norm:

$$[f_b, g_b]_b = \frac{2\pi}{2N+1} \sum_{j=0}^{2N} f_b(x_j) g_b(x_j), \quad \|f_b\|_{b,2}^2 = [f_b, f_b]_b.$$

By the accuracy of the quadrature formula we have

$$(f_b, g_b) = [f_b, g_b]_b, \quad \|f_b\|_2 = \|f_b\|_{b,2}.$$

To establish stability we can either explore the accuracy of the quadrature rule or, alternatively, the properties of the differentiation matrix, particularly its skew-symmetry.

We illustrate these techniques through an example [15, 43, 56, 34].

**Example 13.19.** Let us consider the problem

$$\begin{aligned} \frac{\partial u}{\partial t} + a(x) \frac{\partial u}{\partial x} &= 0, \\ u(x, 0) &= g(x), \end{aligned} \tag{13.14}$$

where the solution is assumed periodic,  $a$  is real and periodic, and  $|a'(x)|$  is bounded to ensure wellposedness. We investigate the question of stability in two different ways.

Let us first assume the method is based on the nodes

$$x_j = \frac{2\pi}{2N+1} j, \quad j \in [0, \dots, 2N],$$

with the associated grid vector

$$\mathbf{u}_b = (u_b(x_0, t), \dots, u_b(x_{2N}, t))^T,$$

and define the diagonal matrix  $\mathbf{A}$  as

$$\mathbf{A}_{jj} = a(x_j).$$

The Fourier collocation scheme for the variable coefficient hyperbolic problem is

$$\frac{d\mathbf{u}_b(t)}{dt} + \mathbf{A} \mathbf{D} \mathbf{u}_b(t) = 0. \tag{13.15}$$

The direct solution of this yields

$$\mathbf{u}_b(t) = \exp[-\mathbf{A} \mathbf{D} t] \mathbf{u}_b(0),$$

and stability is guaranteed provided

$$\|\exp[-\mathbf{A} \mathbf{D} t]\|_2^2 = \exp[-\mathbf{A} \mathbf{D} t] \exp[(-\mathbf{A} \mathbf{D})^T t] \leq K(t).$$

If we first consider the constant coefficient case with  $a(x_j) = a$ , we have

$$\exp[-a \mathbf{D} t] \exp[(-a \mathbf{D})^T t] = \exp[-a(\mathbf{D} + \mathbf{D}^T)t] = 1,$$

since  $\mathbf{D}$  is skew-symmetric and commutes with itself.

If we now consider the case of  $a(x) > 0$ , then  $AD$  no longer commutes with  $-DA$  and the above approach fails. However, since  $a(x) > 0$  we can factorize  $A$  as

$$A = A^{1/2} A^{1/2}$$

and express the matrix exponential as

$$A^{-1/2} \exp[-ADt] A^{1/2} = \exp[-A^{1/2} DA^{1/2} t].$$

Now observe that

$$(A^{1/2} DA^{1/2})^T = A^{1/2} D^T A^{1/2} = -A^{1/2} DA^{1/2},$$

i.e.,  $A^{1/2} DA^{1/2}$  is skew-symmetric. This implies that

$$\begin{aligned} \|\exp[-ADt]\|_2 &= \left\| A^{1/2} \exp[-A^{1/2} DA^{1/2} t] A^{-1/2} \right\|_2 \\ &\leq \left\| A^{1/2} \right\|_2 \left\| \exp[-A^{1/2} DA^{1/2} t] \right\|_2 \left\| A^{-1/2} \right\|_2 \\ &\leq \left\| A^{1/2} \right\|_2 \left\| A^{-1/2} \right\|_2 \leq \frac{\max_x \sqrt{a(x)}}{\min_x \sqrt{a(x)}}, \end{aligned}$$

which establishes stability, provided  $a$  is positive and bounded. When  $a(x) < 0$ , a similar approach is possible, provided  $A$  is factored as

$$A = -|A|^{1/2} |A|^{1/2}.$$

Alternatively, we can proceed by realizing that, if  $a$  is uniformly bounded away from zero, then  $A$  is nonsingular. Therefore, if we multiply from the left by  $u^T A^{-1}$  we recover

$$u^T A^{-1} \frac{du}{dt} = \frac{1}{2} \frac{d}{dt} u^T A^{-1} u = u^T D u = 0,$$

as  $D$  is skew-symmetric and  $A$  is diagonal. This establishes stability in the weighted norm  $u^T A^{-1} u$ . Since this norm is uniformly equivalent to the energy norm, stability follows.

Similar results can be established by using the quadrature rule since

$$\begin{aligned} u_b^T D u_b &= \sum_{j=0}^{2N} u_b(x_j) \frac{du_b}{dx}(x_j) = \frac{2N+1}{2\pi} \int_0^{2\pi} u_b \frac{du_b}{dx} dx \\ &= \frac{2N+1}{4\pi} \int_0^{2\pi} \frac{du_b^2}{dx} dx = 0, \end{aligned}$$

where we use periodicity. This establishes stability in the weighted norm.

When  $a$  changes sign inside the computational domain, the question of stability is more complicated. A stable scheme can be recovered by considering the skew-symmetric form

$$\frac{\partial u}{\partial t} + \frac{1}{2} a(x) \frac{\partial u}{\partial x} + \frac{1}{2} \frac{\partial a(x) u}{\partial x} - \frac{1}{2} a_x(x) u = 0,$$

with the corresponding numerical scheme

$$\frac{\partial u_b}{\partial t} \Big|_{x_j} + \frac{1}{2} a(x_j) \frac{\partial u_b}{\partial x} \Big|_{x_j} + \frac{1}{2} \frac{\partial \mathcal{I}_b[a(x)u_b]}{\partial x} \Big|_{x_j} - \frac{1}{2} a'(x_j) u_b = 0.$$

If we multiply by  $u_b(x_j)$  and sum over all points  $x_j$ , we obtain

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \sum_{j=0}^{2N} u_b^2(x_j, t) &= -\frac{1}{2} \sum_{j=0}^{2N} a(x_j) u_b(x_j, t) \frac{\partial u_b(x, t)}{\partial x} \Big|_{x_j} \\ &\quad - \frac{1}{2} \sum_{j=0}^{2N} u_b(x_j, t) \frac{\partial \mathcal{I}_N[a(x)u_b(x, t)]}{\partial x} \Big|_{x_j} + \frac{1}{2} \sum_{j=0}^{2N} a'(x_j) u_b^2(x_j, t). \end{aligned}$$

The quadrature rule is exact for the second term

$$\begin{aligned} \frac{1}{2} \sum_{j=0}^{2N} u_b(x_j) \frac{\partial \mathcal{I}_N[a(x)u_b(x)]}{\partial x} \Big|_{x_j} &= \frac{2N+1}{4\pi} \int_0^{2\pi} u_b(x) \frac{\partial \mathcal{I}_N[a(x)u_b(x)]}{\partial x} dx \\ &= -\frac{2N+1}{4\pi} \int_0^{2\pi} \mathcal{I}_N[a(x)u_b(x)] \frac{\partial u_N(x)}{\partial x} dx \\ &= -\frac{1}{2} \sum_{j=0}^{2N} a(x_j) u_b(x_j) \frac{\partial u_b(x)}{\partial x} \Big|_{x_j}, \end{aligned}$$

since  $a(x)$  and  $u_b(x, t)$  are periodic.

Hence, we recover

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_{b,2}^2 = \frac{1}{2} \sum_{j=0}^{2N} a_x(x_j) u_b^2(x_j, t) \leq \frac{1}{2} \max_x |a_x(x)| \sum_{j=0}^{2N} u_b^2(x_j, t),$$

which establishes stability, provided the problem is wellposed.

While the skew-symmetric formulation allows for the formulation of a stable scheme, this formulation may be of less interest as the scheme is twice as expensive as the one based on the original problem (13.14). The question of stability of the direct Fourier collocation method to this latter problem is, however, delicate and the scheme is weakly unstable, driven by aliasing [56]. ■

### 13.3 • Nonlinear problems

While the properties of spectral methods for the linear conservation law are well understood, the situation for the nonlinear case is more complex, both in terms of accuracy and stability. Indeed, as we discussed in section 8.3, the appearance of discontinuous solutions gives rise to the Gibbs phenomenon. Not only does this destroy the pointwise accuracy but it also has the potential to impact stability and convergence, due to aliasing errors.

In what follows, we discuss these issues and techniques to address them. We begin with formulations that ensure stability and convergence. In the next section, we return to the question of accuracy in the presence of numerically generated oscillations.

### 13.3.1 • Skew-symmetric form

In Ex. 13.19 we discussed the conservation law

$$\frac{\partial \mathbf{u}}{\partial t} + a(x) \frac{\partial \mathbf{u}}{\partial x} = 0,$$

and observed that the stability of the Fourier collocation scheme cases is delicate when  $a(x)$  changes sign. However, by rewriting the spatial operator in a skew-symmetric form, we observed that stability could be guaranteed, even for the general case, albeit at the expense of doubling the computational cost of the scheme.

Since stability is a primary concern, it is worth exploring such an approach in more detail to understand whether it offers a general path towards stable schemes. Let us therefore consider the system of conservation laws

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} = 0,$$

where  $\mathbf{u}$  is the  $m$ -vector of conserved variables. We assume that the problem is endowed with a convex entropy  $\eta(\mathbf{u})$  and recall from Theorem 3.11 that  $\mathbf{v}^T = \nabla_{\mathbf{u}} \eta(\mathbf{u})$  symmetrizes the system, i.e., we can write it in these new variables as

$$A_0 \frac{\partial \mathbf{v}}{\partial t} + A_1 \frac{\partial \mathbf{v}}{\partial x} = 0,$$

with

$$A_0 = \frac{\partial \mathbf{u}}{\partial \mathbf{v}} > 0, \quad A_1 = \frac{\partial \mathbf{f}}{\partial \mathbf{v}}, \quad (13.16)$$

where  $A_0, A_1$  are both symmetric and the positivity of  $A_0$  follows from the convexity of the entropy.

Let us now further assume that the entropy and the flux are homogeneous functions of order  $\alpha_0$  and  $\alpha_1$ , respectively, i.e., we assume that  $\eta(t\mathbf{u}) = t^{\alpha_0} \eta(\mathbf{u})$ , and likewise for  $\mathbf{f}$ . Since  $\eta$  is homogeneous of order  $\alpha_0$ ,  $\mathbf{v}^T = \nabla_{\mathbf{u}} \eta$  is homogeneous of order  $\alpha_0 - 1$ . Consequently,  $\mathbf{u}(\mathbf{v})$  is homogeneous of order  $1/(\alpha_0 - 1)$ .

Now recall a result, known as Euler's theorem for homogeneous functions, namely

$$\alpha f(x) = x \cdot \nabla_x f(x), \quad (13.17)$$

for a homogeneous function  $f(x)$  of order  $\alpha$ . Using (13.16) allows us to write

$$\frac{1}{\alpha_0 - 1} \mathbf{u} = A_0 \mathbf{v}, \quad \frac{\alpha_1}{\alpha_0 - 1} \mathbf{f} = A_1 \mathbf{v}. \quad (13.18)$$

Returning to the conservation law, we have

$$\frac{\partial \mathbf{u}}{\partial t} = \frac{\alpha_0 - 1}{\alpha_0} \frac{\partial \mathbf{u}}{\partial t} + \frac{1}{\alpha_0} \frac{\partial \mathbf{u}}{\partial t} = \frac{\alpha_0 - 1}{\alpha_0} \left( A_0 \frac{\partial \mathbf{v}}{\partial t} + \frac{\partial A_0 \mathbf{v}}{\partial t} \right),$$

and, similarly,

$$\frac{\partial \mathbf{f}}{\partial x} = \frac{\alpha_0 - 1}{\alpha_0 + \alpha_1 - 1} \frac{\partial \mathbf{f}}{\partial x} + \frac{\alpha_1}{\alpha_0 + \alpha_1 - 1} \frac{\partial \mathbf{f}}{\partial x} = \frac{\alpha_0 - 1}{\alpha_0 + \alpha_1 - 1} \left( A_1 \frac{\partial \mathbf{v}}{\partial x} + \frac{\partial A_1 \mathbf{v}}{\partial x} \right),$$

which establishes the following fundamental result [54].

**Theorem 13.20.** *The conservation law*

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} = 0,$$

with a homogeneous flux of order  $\alpha_1$  and a homogeneous convex entropy of order  $\alpha_0$ , can be written in skew-symmetric form

$$\mathbf{A}_0 \frac{\partial \mathbf{v}}{\partial t} + \frac{\partial \mathbf{A}_0 \mathbf{v}}{\partial t} + \frac{\alpha_0}{\alpha_0 + \alpha_1 - 1} \left( \mathbf{A}_1 \frac{\partial \mathbf{v}}{\partial x} + \frac{\partial \mathbf{A}_1 \mathbf{v}}{\partial x} \right) = 0,$$

in the entropy variable  $\mathbf{v}$ .

This is a strong result, confirming that any conservation law endowed with an entropy can be written in skew-symmetric form. Since all symmetrizable systems have a convex entropy, the result extends to this case. Although we only discussed the result for the one-dimensional system, the result generalizes to the multidimensional case. Furthermore, the assumption that the flux is homogeneous can be overcome [54].

Considering the conservation law on skew-symmetric form

$$\mathbf{A}_0 \frac{\partial \mathbf{v}}{\partial t} + \frac{\partial \mathbf{A}_0 \mathbf{v}}{\partial t} + \frac{\alpha_0}{\alpha_0 + \alpha_1 - 1} \left( \mathbf{A}_1 \frac{\partial \mathbf{v}}{\partial x} + \frac{\partial \mathbf{A}_1 \mathbf{v}}{\partial x} \right) = 0,$$

we multiply by  $\mathbf{v}^T$  from the left to obtain

$$\frac{\partial}{\partial t} \mathbf{v}^T \mathbf{A}_0 \mathbf{v} + \frac{\alpha_0}{\alpha_0 + \alpha_1 - 1} \frac{\partial}{\partial x} \mathbf{v}^T \mathbf{A}_1 \mathbf{v} = 0.$$

Recalling (13.18), we have

$$\mathbf{v}^T \mathbf{A}_0 \mathbf{v} = \frac{1}{(\alpha_0 - 1)^2} \mathbf{u}^T \mathbf{A}_0^{-1} \mathbf{A}_0 \mathbf{A}_0^{-1} \mathbf{u} = \frac{1}{(\alpha_0 - 1)^2} \mathbf{u}^T \nabla_{\mathbf{u} \mathbf{u}} \eta(\mathbf{u}) \mathbf{u},$$

since

$$\mathbf{A}_0^{-1} = \frac{\partial \mathbf{v}}{\partial \mathbf{u}} = \nabla_{\mathbf{u} \mathbf{u}} \eta(\mathbf{u}).$$

Applying (13.17) twice, we recover

$$\mathbf{u}^T \nabla_{\mathbf{u} \mathbf{u}} \eta(\mathbf{u}) \mathbf{u} = \alpha_0(\alpha_0 - 1) \eta(\mathbf{u}).$$

As the entropy is convex, it follows that either  $\alpha_0 > 1$  or  $0 < \alpha_0 < 1$  is required to guarantee that the entropy is single signed. Now define the entropy flux  $\psi(\mathbf{u})$  through the relation

$$\nabla_{\mathbf{u}} \eta(\mathbf{u}) \nabla_{\mathbf{u}} \mathbf{f} = (\nabla_{\mathbf{u}} \psi)^T,$$

and observe that by the definition it must be homogeneous of order  $\alpha_0 + \alpha_1 - 1$ . Hence, using (13.18) we recover

$$\mathbf{v}^T \mathbf{A}_1 \mathbf{v} = \frac{1}{\alpha_0 - 1} \mathbf{v}^T \frac{\partial \mathbf{f}}{\partial \mathbf{v}} \frac{\partial \mathbf{v}}{\partial \mathbf{u}} \mathbf{u} = \frac{1}{\alpha_0 - 1} (\nabla_{\mathbf{u}} \psi)^T \mathbf{u} = \frac{\alpha_0 + \alpha_1 - 1}{\alpha_0 - 1} \psi(\mathbf{u})$$

and, therefore,

$$\frac{\alpha_0}{\alpha_0 + \alpha_1 - 1} \mathbf{v}^T \mathbf{A}_1 \mathbf{v} = \frac{\alpha_0}{\alpha_0 - 1} \psi(\mathbf{u}).$$

When we combine the previous results, we recover

$$\frac{\partial \eta(\mathbf{u})}{\partial t} + \frac{\partial \psi(\mathbf{u})}{\partial x} = 0,$$

where  $(\eta, \psi)$  is the entropy pair. In other words, the skew-symmetric form conserves entropy.

As we discussed in section 3.3 and subsection 8.2.2, entropy conservation may be desirable for smooth solutions. However, once a shock forms, entropy dissipation is required to guarantee recovery of the entropy solution. Before we return to this, let us first realize that the results hold also for the semidiscrete case.

**Example 13.21.** Let us consider Burgers' equation

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{u}^2}{\partial x} = 0,$$

subject to periodic boundary conditions and an appropriate initial condition.

Clearly, Burgers' equation has a homogeneous flux of order 2, i.e.,  $\alpha_1 = 2$ , and we consider the convex entropy  $\eta(\mathbf{u}) = \mathbf{u}^2$  with  $\alpha_0 = 2$ . We can now express Burgers' equation in skew-symmetric form as

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{2}{3} \left( \frac{\partial \mathbf{u}^2}{\partial x} + \mathbf{u} \frac{\partial \mathbf{u}}{\partial x} \right) = 0.$$

Let us assume that we employ a Fourier collocation method, expressed as

$$\frac{d\mathbf{u}}{dt} + \frac{2}{3} (\mathbf{D}\mathbf{u}^2 + \mathbf{u}\mathbf{D}\mathbf{u}) = 0,$$

where  $\mathbf{u}$  is the  $2N + 1$  vector of unknowns and  $\mathbf{D}$  is the differentiation matrix. We rewrite this as

$$\frac{d\mathbf{u}}{dt} + \frac{2}{3} (\mathbf{D}\mathbf{U} + \mathbf{U}\mathbf{D})\mathbf{u} = \frac{d\mathbf{u}}{dt} + \frac{2}{3} \mathbf{D}_u \mathbf{u} = 0,$$

where

$$\mathbf{D}_u = \mathbf{D}\mathbf{U} + \mathbf{U}\mathbf{D}, \quad \mathbf{U} = \text{diag}(\mathbf{u}).$$

Since  $\mathbf{D} = -\mathbf{D}^T$ , it follows that  $\mathbf{D}_u = -\mathbf{D}_u^T$ ,

$$\mathbf{u}^T \frac{d\mathbf{u}}{dt} + \left( \frac{d\mathbf{u}}{dt} \right)^T \mathbf{u} = \mathbf{u}^T \mathbf{D}_u \mathbf{u} + \mathbf{u}^T \mathbf{D}_u^T \mathbf{u} = 0,$$

and the scheme

$$\frac{d\mathbf{u}}{dt} + \frac{2}{3} \mathbf{D}_u \mathbf{u} = 0$$

conserves energy. In this particular case, the energy and the entropy are the same. It is noteworthy that the scheme is stable, even in the case when aliasing is present. ■

### 13.3.2 ■ Vanishing viscosity

From the previous section, it is clear that care has to be exercised to ensure the computation of the correct entropy solution. Let us consider the Fourier–Galerkin approximation to the periodic Burgers’ equation:

$$\frac{\partial u_b}{\partial t} + \mathcal{P}_N \frac{\partial u_b^2}{\partial x} = 0.$$

We rewrite this as

$$\frac{\partial u_b}{\partial t} + \frac{\partial u_b^2}{\partial x} = (I - \mathcal{P}_N) \frac{\partial u_b^2}{\partial x}, \quad (13.19)$$

multiply with  $u_b$ , and integrate over  $x$  to recover

$$\frac{1}{2} \frac{d}{dt} \|u_b\|_2^2 = - \int_0^{2\pi} u_b \frac{\partial u_b^2}{\partial x} = \frac{1}{3} u_b^3 \Big|_0^{2\pi} = 0,$$

by using periodicity and the observation that the right-hand side of (13.19) vanishes by Galerkin orthogonality. This implies that

$$\|u_b(t)\|_2 = \|u_b(0)\|_2 \leq \|u(0)\|_2.$$

Hence, the scheme conserves energy and entropy, both of which violate the need for entropy dissipation after shock formation. With such a scheme, one cannot expect to recover the weak entropy solution.

To overcome this problem, let us consider the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0,$$

subject to periodic boundary conditions. As discussed in Theorem 2.6 its unique entropy solution is the vanishing viscosity solution to

$$\frac{\partial u_\varepsilon}{\partial t} + \frac{\partial f(u_\varepsilon)}{\partial x} = \varepsilon \frac{\partial^2 u_\varepsilon}{\partial x^2},$$

in the limit  $\lim_{\varepsilon \rightarrow 0} u_\varepsilon = u$ .

To mimic this, consider the modified Fourier–Galerkin scheme

$$\frac{\partial u_b}{\partial t} + \mathcal{P}_N \frac{\partial u_b^2}{\partial x} = \varepsilon_b \frac{\partial}{\partial x} \left( \mathcal{Q} * \frac{\partial u_b}{\partial x} \right), \quad (13.20)$$

where  $*$  denotes convolution and define the operator  $\mathcal{Q}(x)$  as

$$\mathcal{Q}(x) = \sum_{n=-N}^N \hat{Q}_n \exp(inx), \quad \hat{Q}_n = \begin{cases} 0, & |n| \leq m(N), \\ \hat{q}_n, & m(N) \leq |n| \leq N, \end{cases} \quad (13.21)$$

where  $\hat{q}_n$  are functions that we shall prescribe shortly. Thus, we can write the dissipation operator as

$$\varepsilon_b \frac{\partial}{\partial x} \left( \mathcal{Q} * \frac{\partial u_b}{\partial x} \right) = -\varepsilon_b \sum_{m(N) \leq |n| \leq N} n^2 \hat{q}_n \tilde{u}_n \exp(inx),$$

so that the connection to the vanishing viscosity solution is established as

$$\frac{\partial u_b}{\partial t} + \frac{\partial f(u_b)}{\partial x} = \varepsilon_b \frac{\partial^2 u_b}{\partial x^2} + \frac{\partial}{\partial x} (I - \mathcal{P}_N) f(u_b) - \varepsilon_b \frac{\partial}{\partial x} \left( \mathcal{R} * \frac{\partial u_b}{\partial x} \right),$$

where  $\mathcal{R} = \mathcal{I} - \mathcal{Q}$ . Expressed in Fourier space as

$$\mathcal{R}(x) = \sum_{n=-N}^N \hat{R}_n \exp(inx), \quad \hat{R}_n = \begin{cases} 1, & |n| \leq m(N), \\ 1 - \hat{q}_n, & m(N) \leq |n| \leq N, \end{cases}$$

it is clear that  $\mathcal{R}$  plays the role of a filter, as discussed in subsection 13.4.1.

The scheme in (13.20) depends on the details of  $\hat{q}_n$  and the two parameters  $\varepsilon_b$  and  $m(N)$ . Let us follow the original work [57], where

$$\hat{q}_n = 1, \quad m(N) \leq |n| \leq N$$

are shown to allow the following result.

**Theorem 13.22.** *Consider the scheme (13.20) with parameters  $\varepsilon_b$  and  $m(N)$  such that*

$$\varepsilon_b \propto N^{-2\beta}, \quad m(N) \propto N^\beta,$$

*for some  $0 < \beta \leq \frac{1}{4}$ . If  $u_b$  is uniformly bounded, then it converges boundedly, almost everywhere, to the unique entropy solution of the conservation law.*

**Proof.** The proof is technical and relies on establishing a number of bounds on the solution which, in combination with results on compensated compactness, allow for establishing convergence. We refer to [57] for the details.  $\square$

As discussed in [57], the bounds to select  $\varepsilon_b$  and  $m(N)$  are sufficient to guarantee convergence, but may not be necessary as shown by computational results. In [58, 46], it is demonstrated that an alternative choice

$$\varepsilon_b \propto N^{-1}, \quad m(N) \propto N^\beta, \quad 0 < \beta < \frac{1}{2},$$

likewise yields good results.

While Theorem 13.22 establishes convergence of the solution to the unique entropy solution, the accuracy of the computational results is less clear. To realize the impact of the vanishing viscosity term, consider [57]

$$\left\| \varepsilon_b \frac{\partial}{\partial x} \left( \mathcal{Q} * \frac{\partial u_b}{\partial x} \right) \right\|_2 \leq C m^{-s} \|u\|_{s+2} \leq C N^{-\beta s} \|u\|_{s+2},$$

for  $u \in H^{s+2}([0, 2\pi])$ . Hence, the vanishing viscosity term does not impact the overall convergence rate, and allows spectral convergence for sufficiently smooth solutions.

The extension of these results to the Fourier collocation method is completed in [46] and some additional results are found in [58].

Whereas the vanishing viscosity method ensures stability and convergence, an improved vanishing superviscosity method has been introduced. To improve accuracy

in this case, one considers

$$\frac{\partial u_b}{\partial t} + \mathcal{P}_N \frac{\partial f(u_b)}{\partial x} = \varepsilon_b (-1)^{s+1} \frac{\partial^s}{\partial x^s} \left( \mathcal{Q} * \frac{\partial^s u_b}{\partial x^s} \right). \quad (13.22)$$

In [59], the following result is established.

**Theorem 13.23.** *Consider the scheme (13.22) with parameters  $\varepsilon_b$  and  $m(N)$  such that*

$$\varepsilon_b \propto \frac{C}{N^{2s-1}}, \quad m(N) \propto N^\theta, \quad 0 < \theta \leq \frac{2s-1}{2s},$$

for  $Q$  in (13.21) defined through

$$1 - \left( \frac{m(N)}{|n|} \right)^{\frac{2s-1}{\theta}} \leq \hat{q}_n \leq 1, \quad m(N) \leq |n| \leq N.$$

If  $u_b$  is uniformly bounded, then it converges boundedly, almost everywhere, to the unique entropy solution of the conservation law.

The main advantage of the superviscosity approach is a decrease in the amount of dissipation needed. However, this comes at the expense of re-introducing artificial oscillations, as the solution can no longer be guaranteed to be monotone.

Before discussing the performance of the vanishing viscosity approach, it is worth it to briefly discuss its efficient implementation. If we were to directly implement

$$\varepsilon_b (-1)^{s+1} \frac{\partial^s}{\partial x^s} \left( \mathcal{Q} * \frac{\partial^s u_b}{\partial x^s} \right) = -\varepsilon_b \sum_{m(N) \leq |n| \leq N} n^{2s} \hat{q}_n \tilde{u}_n \exp(inx),$$

it would impact the maximum allowable time step if an explicit method were used. As an alternative, consider

$$\frac{\partial u_b}{\partial t} = \varepsilon_b (-1)^{s+1} \frac{\partial^s}{\partial x^s} \left( \mathcal{Q} * \frac{\partial^s u_b}{\partial x^s} \right),$$

and express the solution in Fourier space as

$$u_b(x, t) = \sum_{n=-N}^N \tilde{u}_n(x, t) \exp(inx) = \sum_{n=-N}^N \exp(-\varepsilon_b n^{2s} \hat{q}_n t) \tilde{u}_n(x, 0) \exp(inx).$$

Taken over a time step, we recover

$$u_b(x, k) = \sum_{n=-N}^N \exp(-\varepsilon_b n^{2s} \hat{q}_n k) \tilde{u}_n(x, 0) \exp(inx).$$

We can generalize this as

$$u_b(x, t+k) = \sum_{n=-N}^N \sigma \left( \frac{|n|}{N} \right) \tilde{u}_n(t) \exp(inx),$$

where we define the filter function as

$$\sigma(\xi) = \begin{cases} 1, & \xi \leq m/N, \\ \exp(-\varepsilon_b k N^{2s} \xi^{2s} \hat{q}_\xi), & m/N \leq \xi \leq 1, \end{cases}$$

and

$$\hat{q}_\xi = 1 - \left( \frac{m}{N\xi} \right)^{\frac{2s-1}{\theta}}.$$

Hence, we can apply the vanishing viscosity as a filter, thereby eliminating the cost of a reduced time step as  $s$  increases. For the hyperviscosity where  $s > 1$ , this is essential to make the method practical.

The implementation of the vanishing viscosity as a filter is illustrated in *FourierVanishHypVisc.m*.

**Script 13.3. *FourierVanishHypVisc.m*: Routine to apply vanishing hyperviscosity to a Fourier spectral method.**

---

```
function [Fu] = FourierVanishHypVisc(u,s,k,N,L);
% function [Fu] = FourierVanishHypVisc(u,s,k,N,L);
% Purpose: Apply vanishing viscosity as a filter

% Set parameters in vanishing viscosity model
theta = 0.9*(2*s-1)/(2*s); mN = floor((L/(2*pi)*N)^theta);
C=0.5*(2*pi/L)^(2*s);

% Define vanishing viscosity by a filter function
qhat = zeros(N+1,1); qhat((mN+1):(N+1)) = 1 - (mN./[mN:N]).^(2*s-1)/theta;
sigma = exp(-C*N*k*(([0:N])/N).^(2*s).*qhat);
nvec = [sigma; flipud(sigma(2:end))];
Fu = real(ifft(nvec.*fft(u)));
return
```

---

We illustrate the behavior of the vanishing viscosity on the solution of a nonlinear conservation law.

**Example 13.24.** Consider Burgers' equation

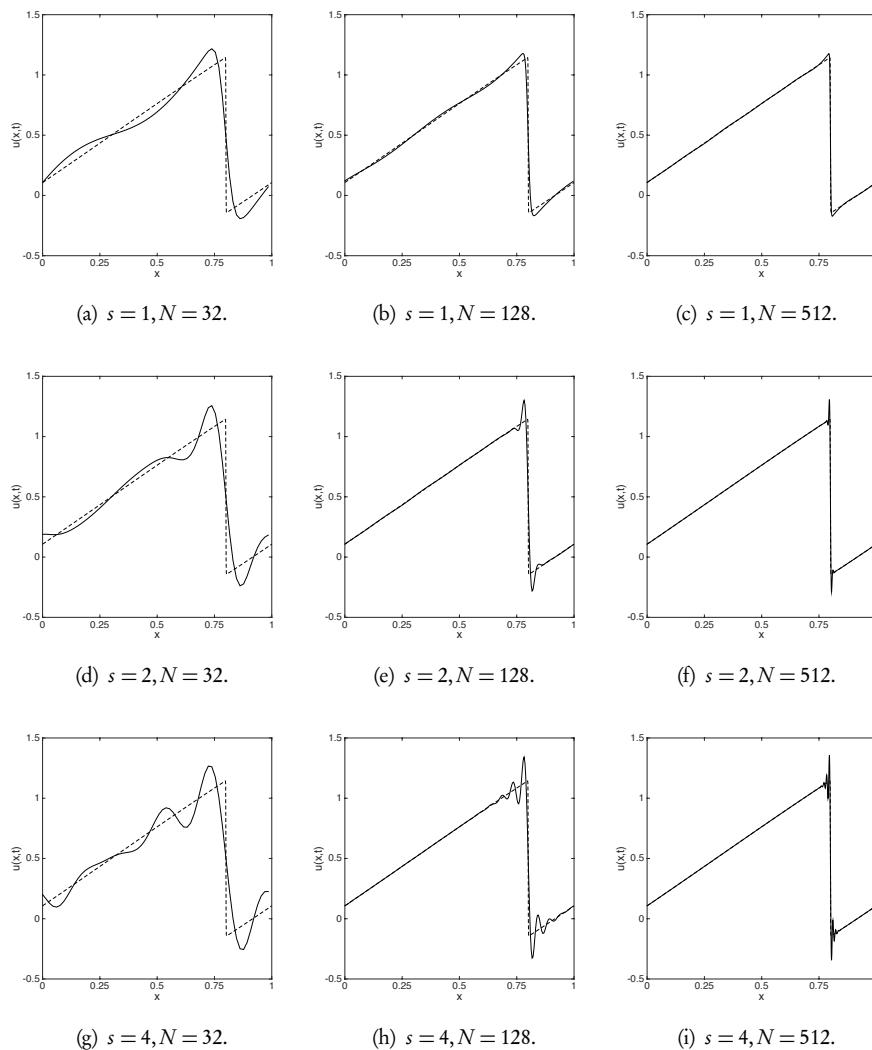
$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

subject to periodic boundary conditions and the initial condition

$$u(x, 0) = \sin(2\pi x) + 0.5.$$

This initial condition gives rise to steeping and the formation of a shock that propagates to the right while slowly dissipating. The problem is solved using a Fourier collocation method with vanishing hyperviscosity to ensure stability and accuracy. The results for different values of  $N$  and different values of the order of the hyperviscosity are shown in Fig. 13.3. They confirm visual convergence of the scheme to the entropy solution and the elimination of Gibbs oscillations in the case of  $s = 1$ . For higher values of  $s$ , we note a substantial improvement in the overall accuracy of the solution, but also that Gibbs oscillations reappear in the neighborhood of the discontinuity. ■

The extension of vanishing viscosity techniques to systems is discussed in [57] and can be done in a componentwise manner, although there is no theoretical support for such a procedure. Extensions to multidimensional scalar conservation laws are discussed in [9] and the development of vanishing viscosity techniques for nonperiodic problems was initiated in [45, 41, 32].



**Figure 13.3.** Solution of Burgers' equation with a Fourier collocation method, stabilized with a vanishing hyperviscosity term of order  $2s$ . The solution is shown at  $T = 0.3$  and the dashed line is a reference solution obtained with a highly resolved, essentially nonoscillatory scheme.

## 13.4 • Postprocessing

In section 8.3 and again in subsection 12.2.1 we found that a solution may still exhibit high-order accuracy even when contaminated by Gibbs oscillations. In fact, the discussion in section 8.3 extends immediately to the case of Fourier spectral methods. Furthermore, as illustrated in Ex. 13.24, it is clear that by increasing the order of the hyperviscosity, the overall accuracy substantially improves, albeit at the expense of local Gibbs oscillations.

Therefore, it is of interest to pursue techniques that enable recovery of the spectrally accurate solution that underlies the computed solution. A number of such postprocessing techniques have been developed, and in the following we focus on three of these and provide some insight into their efficiency.

### 13.4.1 • Filtering

Filters were introduced in subsection 8.3.3 to reduce the impact of the Gibbs oscillation and we observed, among other things, that filtering has the potential to improve the accuracy away from the points of discontinuity. Although this was only an experimental observation, additional observations, more directly related to the interplay between the smoothness of the function and the properties of a filter, were made in subsection 12.2.2.

In what follows, we revisit this discussion in the context of Fourier spectral methods. As we shall see, this allows for the development of a firm approximation theory for the filtered expansion.

Since we are concerned with postprocessing techniques for computational results, let us assume that we know the first  $2N+1$  expansion coefficients of a piecewise analytic function,  $u$ , defined on  $[0, 2\pi]$  with a point of discontinuity at  $x = \xi$ . Our objective is to recover the value of the function at any point in the interval  $[0, 2\pi]$ , with the highest possible order of approximation. We recall that, without any post-processing, we can generally expect no better than first order pointwise accuracy, and no convergence at the point of discontinuity due to the Gibbs phenomenon.

Let us consider the modified approximation

$$u_b^\sigma(x) = \sum_{n=-N}^N \sigma\left(\frac{|n|}{N}\right) \tilde{u}_n \exp(inx),$$

where  $\sigma$  is the filter function. The objective is to design  $\sigma$  such that the filtered approximation converges faster than the original expansion:

$$u_b(x) = \sum_{n=-N}^N \tilde{u}_n \exp(inx).$$

As already discussed in subsection 8.3.3, we require the filter function to have the following properties [61].

**Definition 13.25.** For any  $p$  ( $p > 1$ ), the filter is defined by the function  $\sigma \in C^p : R^+ \rightarrow [0, 1]$  such that

$$\begin{cases} \sigma^{(k)}(0) = \delta_{k0}, & k = 0, \dots, p-1, \\ \sigma(\eta) = 0, & \eta \geq 1, \\ \sigma^{(k)}(\pi) = 0, & k = 1, \dots, p-1. \end{cases}$$

Following this definition, we express the modified approximation as

$$u_b^\sigma(x) = \sum_{n=-\infty}^{\infty} \sigma(\eta) \tilde{u}_n \exp(inx) = \frac{1}{2\pi} \int_0^{2\pi} S(x-y) u(y) dy, \quad (13.23)$$

where  $\eta = |n|/N$  and we have the filter function

$$S(z) = \sum_{n=-\infty}^{\infty} \sigma(\eta) \exp(inz), \quad (13.24)$$

recovered by inserting the continuous expansion coefficients and rearranging the terms. Note that in (13.23) we replaced the truncated series with the actual function  $u$ , thus eliminating the impact of the truncation error. In this way, the following discussion remains simple without impacting the validity of the results.

Following [61], let us introduce a family of filter functions.

**Definition 13.26.** *The family of periodic filter functions  $S_l(z)$  is defined from  $S$  (13.24) as*

$$\begin{aligned} S_0(z) &= S(z), & S'_l(z) &= S_{l-1}(z), \\ \int_0^{2\pi} S_l(z) dz &= 0, & l &\geq 1. \end{aligned}$$

This definition leads to several equivalent representations of the corresponding filter family  $S_l$ , stated in the following lemma.

**Lemma 13.27.** *Let  $\sigma(\eta)$  represent a filter of order  $p$ , with the associated filter function*

$$S(z) = S_0(z) = \sum_{n=-\infty}^{\infty} \sigma(\eta) \exp(inz).$$

*Then, the filter family  $S_l$  ( $1 \leq l \leq p$ ) admits the following equivalent representations:*

(a)

$$S_l(z) = \frac{1}{N^l} \sum_{n=-\infty}^{\infty} G_l(\eta) i^l \exp(inz), \quad G_l(\eta) = \frac{\sigma(\eta) - 1}{\eta^l},$$

(b)

$$S_l(z) = \frac{1}{N^{l-1}} \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} i^l \exp(iN(z + 2\pi m)\eta) G_l(\eta) dz,$$

(c)

$$S_1(z) = z - \pi + \sum_{\substack{n=-\infty \\ n \neq 0}}^{n=\infty} \sigma(\eta) (in)^{-1} \exp(inx),$$

$$S_l(z) = B_l(z) + \sum_{\substack{n=-\infty \\ n \neq 0}}^{n=\infty} \sigma(\eta) (in)^{-l} \exp(inx), \quad l \leq 2,$$

where  $B_l$  is the Bernoulli polynomial of order  $l$ .

The importance of the filter family and its integrals is expressed in the following result [61].

**Theorem 13.28.** *Let  $u$  be a piecewise  $C^p([0, 2\pi])$  function with a single point of discontinuity at  $\xi$ . Then*

$$u_b^\sigma(x) - u(x) = \frac{1}{2\pi} \sum_{l=0}^{p-1} S_{l+1}(c) (u^{(l)}(\xi^+) - u^{(l)}(\xi^-)) + \frac{1}{2\pi} \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy,$$

where  $c = x - \xi$  for  $x > \xi$  and  $c = 2\pi + x - \xi$  for  $x < \xi$ .

**Proof.** In the first place, as a consequence of the definition of the filter family, we have that

$$S_l(2\pi) - S_l(0) = 2\pi \delta_{l1} \quad (13.25)$$

for  $l \geq 1$ . For  $l = 1$ , this follows from c) for  $S_l(z)$  in Lemma 13.27, while for  $l > 1$  it is a consequence of the integral condition on  $S_l$ .

Let us now consider the case  $x > \xi$ . If we integrate (13.23) by parts  $p$  times, excluding the point of discontinuity, we recover

$$\begin{aligned} 2\pi u_b^\sigma(x) &= \int_0^{\xi^-} S(x-y) u(y) dy + \int_{\xi^+}^{2\pi} S(x-y) u(y) dy \\ &= \sum_{l=0}^{p-1} (S_{l+1}(2\pi) - S_{l+1}(0)) u^{(l)}(x) \\ &\quad + \sum_{l=0}^{p-1} S_{l+1}(x-\xi) (u^{(l)}(\xi^+) - u^{(l)}(\xi^-)) + \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy \\ &= 2\pi u(x) + \sum_{l=0}^{p-1} S_{l+1}(x-\xi) (u^{(l)}(\xi^+) - u^{(l)}(\xi^-)) + \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy, \end{aligned}$$

where the last reduction follows from (13.25). Likewise, for  $x < \xi$  we obtain

$$\begin{aligned} 2\pi u_b^\sigma(x) &= \int_0^{\xi^-} S(x-y) u(y) dy + \int_{\xi^+}^{2\pi} S(x-y) u(y) dy \\ &= \sum_{l=0}^{p-1} (S_{l+1}(2\pi) - S_{l+1}(0)) u^{(l)}(x) \\ &\quad + \sum_{l=0}^{p-1} S_{l+1}(2\pi + x - \xi) (u^{(l)}(\xi^+) - u^{(l)}(\xi^-)) + \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy \\ &= 2\pi u(x) + \sum_{l=0}^{p-1} S_{l+1}(2\pi + x - \xi) (u^{(l)}(\xi^+) - u^{(l)}(\xi^-)) \\ &\quad + \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy, \end{aligned}$$

which establishes the result.  $\square$

This offers a precise estimate of the pointwise difference between  $u$  and its truncated and filtered representation  $u_h^\sigma$ . To complete the analysis, we need to estimate the two expressions in Theorem 13.28.

Let us first consider the last term in Theorem 13.28. The estimate is based on classic results. Indeed, when  $u \in C^{p-1}([0, 2\pi])$  the first term of Theorem 13.28 vanishes, and the last term is the error term associated with smooth functions, as stated in the following result.

**Lemma 13.29.** *Let  $S_l$  be defined as in (13.24). Then*

$$\frac{1}{2\pi} \left| \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy \right| \leq C \frac{\sqrt{N}}{N^p} \left( \int_0^{2\pi} |u^{(p)}|^2 dx \right)^{1/2},$$

where the constant  $C$  is independent of  $u$  and  $N$ .

**Proof.** Since  $S_p$  is periodic, the Cauchy–Schwarz inequality yields

$$\frac{1}{2\pi} \left| \int_0^{2\pi} S_p(x-y) u^{(p)}(y) dy \right| \leq \frac{1}{2\pi} \left( \int_0^{2\pi} S_p^2(x) dx \right)^{1/2} \left( \int_0^{2\pi} |u^{(p)}(x)|^2 dx \right)^{1/2}.$$

We can estimate the first term if we express  $S_p$  using (a) of Lemma 13.27, to obtain

$$\begin{aligned} \frac{1}{2\pi} \int_0^{2\pi} S_p^2(x) dx &= \frac{1}{2\pi} \int_0^{2\pi} \sum_{n=-\infty}^{\infty} \left( \frac{1}{N^p} \frac{\sigma(\eta)-1}{\eta^p} i^p \exp(inx) \right)^2 \\ &= \sum_{n=-\infty}^{\infty} \frac{1}{N^{2p}} \left( \frac{\sigma(\eta)-1}{\eta^p} \right)^2 \\ &= \frac{1}{N^{2p-1}} \sum_{n=-N}^N \frac{2}{2N} \left( \frac{\sigma(\eta)-1}{\eta^p} \right)^2 + \sum_{|n|>N} \frac{1}{n^{2p}} \\ &\leq \frac{1}{N^{2p-1}} \int_{-1}^1 \left( \frac{\sigma(\eta)-1}{\eta^p} \right)^2 d\eta + \frac{1}{N^{2p-1}} \leq C \frac{N}{N^{2p}}. \end{aligned}$$

Here we used orthogonality of the exponential function and bound the Riemann sum by its integral.  $\square$

Second, to estimate the first term in Theorem 13.28 we use Lemma 13.27, as these suffice to guarantee boundedness.

**Lemma 13.30.** *Let  $S_l$  be defined as in (13.24). Then*

$$|S_l(x)| \leq C \frac{1}{N^{p-1}} \left( \frac{1}{|x|^{p-l}} + \frac{1}{|2\pi-x|^{p-l}} \right) \int_{-\infty}^{\infty} |G_l^{(p-l)}(\eta)| d\eta,$$

where  $G_l$  is defined in Lemma 13.27.

**Proof.** Recall the representation of  $S_l$  in b) of Lemma 13.27. Since  $G_l$  is  $p-1$  times differentiable and  $\sigma$  is defined so that  $\sigma^{(l)}(0) = \sigma^{(l)}(1) = 0$ , integration by parts  $p-l$  times ( $l \leq 1$ ) is

$$|S_l(x)| = \frac{1}{N^{l-1}} \left| \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\exp[iN(x+2\pi m)\eta]}{N^{p-l}(x+2\pi m)^{p-l}} G_l^{(p-l)}(\eta) d\eta \right|.$$

Since  $x \in [0, 2\pi]$ , the dominating terms are those corresponding to  $m = 0$  and  $m = -1$ . By the triangle inequality, and taking these two contributions outside the integral, the result follows.  $\square$

Finally, we are now in the position to state the main theorem [61].

**Theorem 13.31.** *Let  $u$  be a piecewise  $C^p([0, 2\pi])$ -function with only one point of discontinuity at  $\xi$  and let  $\sigma$  be a filter as defined in Definition 13.25. We denote the distance between a point  $x \in [0, 2\pi]$  and the discontinuity as  $d(x) = |x - \xi|$ .*

*The pointwise difference between  $u$  and its truncated and filtered representation  $u_b^\sigma$  at all  $x$  with the exception of  $\xi$  can be bounded*

$$|u(x) - u_b^\sigma(x)| \leq C_1 \frac{1}{N^{p-1}} \frac{1}{d(x)^{p-1}} K(u) + C_2 \frac{\sqrt{N}}{N^p} \left( \int_0^{2\pi} |u^{(p)}|^2 dx \right)^{1/2},$$

where

$$K(u) = \sum_{l=0}^{p-1} d(x)^l \left| u^{(l)}(\xi^+) - u^{(l)}(\xi^-) \right| \int_{-\infty}^{\infty} |G_l^{(p-l)}(\eta)| d\eta.$$

**Proof.** The second part of the estimate follows directly from Lemma 13.29. Similarly, Lemma 13.30 implies that the first part of the expression in Theorem 13.28 is bounded.

Since  $c = x - \xi$  for  $x > \xi$  and  $c = 2\pi + x - \xi$  for  $x < \xi$  in Theorem 13.28 we have

$$\frac{1}{|c|^{p-l}} + \frac{1}{|2\pi - c|^{p-l}} \leq \frac{2}{d(x)^{p-l}}.$$

Combined with the estimate in Lemma 13.30, this yields the result.  $\square$

The theorem proves that filtering is an effective post-processing tool away from the discontinuity while we cannot expect improvements in the rate of convergence at the point of discontinuity. This is in full agreement with the results in Ex. 12.22, although those results were obtained for a polynomial basis. Nevertheless, the qualitative behavior remains the same. We also observe that if the order of the filter  $p$  is low, it limits the maximum rate of convergence, which is again in full agreement with Theorem 13.31. The extension of the results for the Fourier series to the nonperiodic polynomial case is discussed in [37], although a complete analysis remains open.

The practical implementation of the filter is straightforward, as illustrated in FourierFm.

**Script 13.4. FourierF.m: Routine to apply a filter to a Fourier expansion method.**


---

```

function [Fu] = FourierF(u,p,alpha,N0);
% function [Fu] = FourierF(u, alpha, N0);
% Purpose: Apply modal exponential filter in a Fourier spectral approach;
N = length(u); N2 = (N-1)/2;

% Define filter function
sigma = ones(N2+1,1);
sigma((N0+1):(N2+1)) = exp(-alpha *(([N0:N2]'-N0)/(N2-N0)).^(2*p));
nvec = [sigma; flipud(sigma(2:end))];
Fu = real(ifft(nvec.*fft(u)));
return

```

---

In this case we consider the widely used exponential filter function defined as

$$\sigma\left(\frac{|n|}{N}\right) = \begin{cases} 1, & |n| \leq N_0, \\ \exp\left(-\alpha\left(\frac{|n|-N_0}{N-N_0}\right)^{2p}\right), & N_0 \leq |n| \leq N, \\ 0, & |n| > N. \end{cases}$$

The order  $2p$ , the strength  $\alpha$ , and the filter cutoff  $N_0$  are used to adjust the strength of the filter.

To provide a basis for comparison between different postprocessing schemes, it is illustrative to consider an example.

**Example 13.32.** The periodic function

$$u(x) = \begin{cases} 2x, & 0 \leq x \leq \frac{1}{2}, \\ 2(x-1), & \frac{1}{2} < x \leq 1 \end{cases} \quad (13.26)$$

has a discontinuity at  $x = \frac{1}{2}$ , and its Fourier coefficients can be computed exactly as

$$\tilde{u}_n = \frac{i(-1)^n}{\pi n}.$$

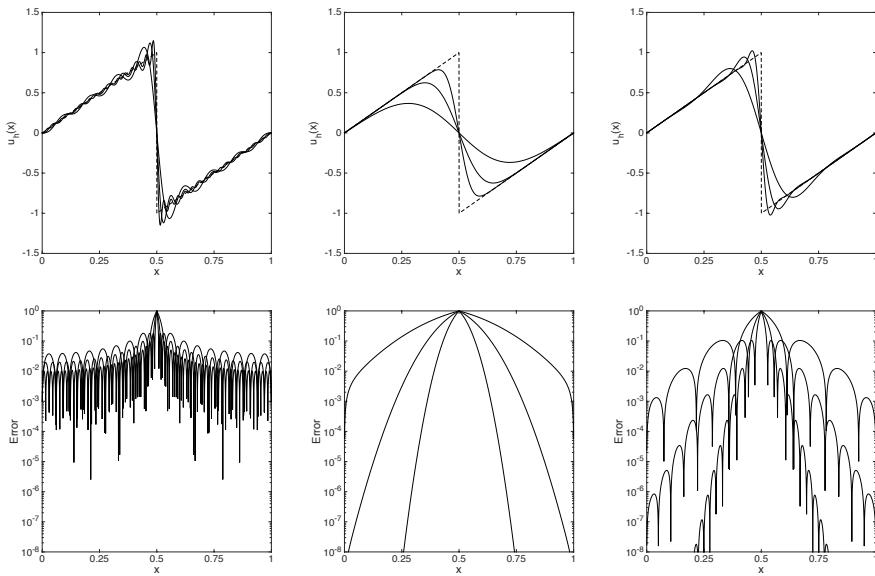
The function, its truncated Fourier representation, and the impact of the exponential filter on the overall accuracy of the truncated approximation is illustrated in Fig. 13.4. In agreement with the results of Theorem 13.31, we observe a substantial improvement in the accuracy of the filtered approximation away from the discontinuity but no noticeable improvement in the neighborhood of  $x = \frac{1}{2}$ . On the other hand, by increasing the order of the filter, the quality of the solution close to the discontinuity improves dramatically. ■

### 13.4.2 • Fourier–Padé reconstruction

An alternative to the use of filters can be pursued by expressing the Fourier series as [12]

$$u_b(x) = \sum_{n=-N}^N \tilde{u}_n \exp(inx) = \sum_{n=0}^N' \tilde{u}_n z^n + \sum_{n=0}^N \tilde{u}_{-n} z^{-n} = u_b^+(z) + u_b^-(z),$$

where  $z = \exp(ix)$  and  $\sum'$  indicates that a factor of  $1/2$  has been added for index  $n = 0$ . We recognize this as a Laurent series and  $u_b^+(z)$  and  $u_b^-(z)$  can be viewed as Taylor series around  $z = 0$  and  $z = \infty$ , respectively. In case  $u_b(x)$  is real,  $\tilde{u}_n = \overline{\tilde{u}_{-n}}$ .



**Figure 13.4.** Illustration of the impact of filtering on the Fourier approximation of a discontinuous function. In the top row, we show the solution for  $N = 8, 16, 32$ , while the bottom row shows the corresponding pointwise error. Higher order, i.e., higher values of  $N$ , yields improved accuracy in all figures. In the first column, no filtering is applied, while the middle and last column uses a second order ( $p = 1$ ) and a fourth order ( $p = 2$ ) filter, respectively.

While Taylor series have known problems of convergence for nonanalytic functions, the convergence of rational functions, i.e., Padé-forms, is superior [5]. Indeed, as shown in [50], the Gibbs phenomenon for certain Padé-approximants to a discontinuous function is 0.4% of the size of the jump, as compared to the 9% for the classic Fourier series.

Therefore, it is natural to consider approximations of the form

$$u_b^+(z) = \frac{p_M^+(z)}{q_L^+(z)} + \mathcal{O}(z^{N+1}), \quad u_b^-(z) = \frac{p_M^-(z)}{q_L^-(z)} + \mathcal{O}(z^{N+1}),$$

where  $p_M^\pm(z)$  and  $q_L^\pm(z)$  are polynomials of order  $M$  and  $L$ , respectively. The computation of this Padé approximation, also known as a Fourier–Padé approximation as it is based on the Fourier series, can be pursued in different ways. Here we focus on the linear Fourier–Padé-form by considering

$$p_M^+(z) + q_L^+(z)u_b^+(z) = \mathcal{O}(z^{N+1}), \quad p_M^-(z) + q_L^-(z)u_b^-(z) = \mathcal{O}(z^{N+1}). \quad (13.27)$$

We first observe that we have  $N + 1$  known coefficients, and  $M + L + 2$  unknown coefficients to specify  $p_M$  and  $q_L$ . However, we can scale the two polynomials arbitrarily, hence leaving one coefficient undetermined. We therefore require  $M + L = N$  to ensure uniqueness of the Padé-form up to a scaling.

To recover  $q_L$ , observe that  $q_L u_b = 0$  for all polynomial orders of  $M + 1$  through  $N$ . Since  $u_b$  is given, this yields a linear Toeplitz system of order  $L \times (L + 1)$ , which

ensures a nonempty null space. By fixing one coefficient, typically  $q_0 = 1$ , the system becomes square. Once  $q_L$  is found,  $p_M$  is recovered directly from (13.27). Since we assume  $u_b$  is real, we recover that  $p_M^+(z) = \overline{p_M^-(z^{-1})}$  and, likewise, for  $q_L^\pm$ . Hence, only one of the two polynomials are computed.

Once all polynomials are recovered, the Fourier–Padé approximation is obtained as

$$u_b(x) = \frac{p_M^+(e^{ix})}{q_L^+(e^{ix})} + \frac{p_M^-(e^{-ix})}{q_L^-(e^{-ix})}.$$

In `FourierPade.m`, we illustrate the implementation of the Fourier–Padé reconstruction of a general periodic function  $u_b$ , given at the  $2N + 1$  grid points  $x_j$ .

**Script 13.5. *FourierPade.m*: Routine to recover a Fourier–Padé reconstruction to a Fourier representation of a function.**

---

```

function [Pu] = FourierPade(x,u,N0,Nc,M,L);
% function [Pu] = FourierPade(x,u,N,M,L);
% Purpose: Apply Fourier–Padé postprocessing to periodic function with
% p_M(x)/q_L(x) = u_Nc(x) + x^(Nc+1). u_N is assumed real
% Approach follows Driscoll and Fornberg (2001).
% NOTE: M+L=Nc is assumed

% Extract Nc Fourier coefficients and compute denominator
uh = fft(u)/(2*N0+1); uhat = uh(1:Nc+1); N=Nc;

Cq = uhat(M+2:N+1); Rq = [ uhat(M+2:-1:max(1,M-L+2)) ; zeros(L-M-1,1) ];
Z = null(toeplitz(Cq,Rq));
qp = Z(:,end); qp = qp/qp(min(find(qp))); qm = conj(qp);

% Compute numerator
Cq = uhat(1:M+1); Cq(1) = Cq(1)/2;
Rq = zeros(1,M+1); Rq(1) = Cq(1);
A = toeplitz(Cq,Rq);
pp = A*qp(1:M+1); pm = conj(pp);

% Evaluate Pade–Fourier approximation
Nx = length(x); xl = 2*pi/Nx*[0:Nx-1]';
ii = sqrt(-1.0); xp = exp(ii*xl); xm = exp(-ii*xl);

Pu = polyval(flipud(pp),xp) ./ polyval(flipud(qp),xp) ...
    + polyval(flipud(pm),xm) ./ polyval(flipud(qm),xm);
return

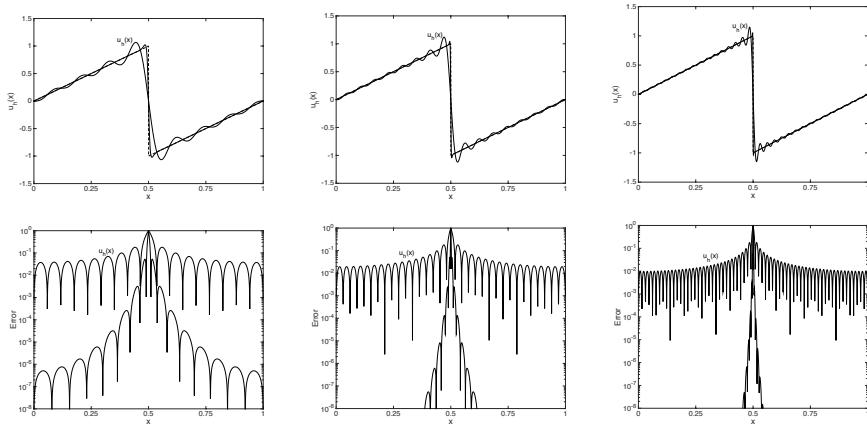
```

---

To illustrate the performance of the Fourier–Padé reconstruction, let us revisit the problem discussed in Ex. 13.32.

**Example 13.33.** We seek a Fourier–Padé reconstruction of the periodic discontinuous function, discussed in Ex. 13.32. For simplicity, we use  $M = L = N/2$  in the Padé-form, although this is not required.

The function, its truncated Fourier representation, and the Fourier–Padé reconstructed function are illustrated in Fig. 13.5. We also show the pointwise errors. When compared to the results of Fig. 13.4, the Fourier–Padé reconstruction is vastly superior in terms of pointwise accuracy and reduction of the overshoot in the neighborhood of the discontinuity, in agreement with the analysis in [50]. However, the Gibbs phenomenon persists and the Fourier–Padé reconstruction does not improve the convergence at the point of discontinuity. ■



**Figure 13.5.** Illustration of the impact of a Fourier–Padé reconstruction on the Fourier approximation of a discontinuous function. In the top row, we show the solution, the truncated Fourier approximation  $u_b$  and the Fourier–Padé processed approximation. The bottom row shows the corresponding pointwise error. In the first column  $N = 8$ , in the middle column  $N = 16$ , and in the last column  $N = 32$ . The exact function is marked by dashed lines.

In [12], by considering a nonperiodic function,  $f(x) = x$ , it is observed that the convergence of the Fourier–Padé approximation in the neighborhood of a discontinuity is dominated by a logarithmic term. This leads to the introduction of the singular Fourier–Padé approximation, defined as

$$u_b(z)^\pm = \frac{p_M^\pm(z)}{q_L^\pm(z)} + \log\left(1 - \frac{z}{\xi^\pm}\right) \frac{r_R^\pm(z)}{q_L^\pm(z)}, \quad (13.28)$$

where  $\xi^\pm = e^{\pm i x_0}$  represents the location  $x_0$  of the discontinuity and  $r_R^\pm$  are polynomials of order  $R$ . If more than one discontinuity is present, additional terms can be added and discontinuities in the  $s$ th derivative can be addressed through terms of the type  $r_R(z) = (z + 1)^s \log(z + 1)$ .

If we keep the denominator  $q_L$  constant for both Padé-forms in (13.28), the polynomials,  $p_M$ ,  $q_L$ , and  $r_R$ , can be recovered by considering the linear Padé problem as outlined above. The reconstruction using the singular Fourier–Padé-form is outlined in *SingularFourierPade.m*, which enables a reconstruction with an arbitrary number of points of discontinuity.

**Script 13.6.** *SingularFourierPade.m*: Routine to apply the singular Fourier–Padé reconstruction to a Fourier representation of a function.

```

function [Pu] = SingularFourierPade(x,u,z,N0,Nc);
% function [Pu] = SingularFourierPade(x,u,N);
% Purpose: Apply Singular Fourier–Padé postprocessing to periodic function
% p_M(x)/q_L(x) + r_R(x)/Q_L(x) = u_Nc(x) + x^(Nc+1). u_N is assumed real
% Approach follows Driscoll and Fornberg (2001) and
% padelog.m (Driscoll, MathWorks File Exchange, 2006)

% Extract Nc Fourier coefficients and compute denominator
uh = fft(u)/(2*N0+1); uhat = uh(1:Nc+1); N=Nc;

```

```

% Determine parameters and location of discontinuities in [0,2pi]
Nx = length(x); xl = 2*pi/Nx*[0:Nx-1]';
ii = sqrt(-1.0); xp = exp(ii*xl); xm = exp(-ii*xl);

xmin = min(x); xmax = max(x); xi = (z-xmin)/(xmax-xmin)*2*pi;
xip = exp(ii*xi); xim = exp(-ii*xi); mm = length(z);

% Setting orders of polynomials — these can be adjusted
L = ceil((N-mm)/(mm+1.5))-1; s = floor((N-mm-L)/(mm+1))+1;
R = s*ones(1,mm); M = N-mm-L-sum(R);

% Taylor coeffs of log terms
k = (1:N)';
for s = 1:mm
    lp{s} = [0;-1./(k.*xip(s).^k)];
end

% The polynomials Q and R{:} are found from the highest-order coeffs
row = [uhat(M+2:-1:max(1,M-L+2)); zeros(L-M-1,1)];
Cq = toeplitz(uhat(M+2:N+1),row);
Lp = cell(1,mm);
for s = 1:mm
    row = [lp{s}(M+2:-1:max(1,M-R(s)+2)); zeros(R(s)-M-1,1)];
    Lp{s} = toeplitz(lp{s}(M+2:N+1),row);
end

% Find a vector v satisfying [Cq -Lp{1} ... -Lp{m}]*v = 0
Z = null(cat(2,-Cq,Lp{:}));
qr = Z(:,end); qr = qr/qr(min(find(qr)));

% Pull out polynomials
qp = qr(1:L+1);
idx = L+1; rp = cell(1,mm); rm = cell(1,mm);
for s = 1:mm
    rp{s} = qr(idx+(1:R(s)+1)); rm{s} = conj(rp{s});
    idx = idx + R(s)+1;
end

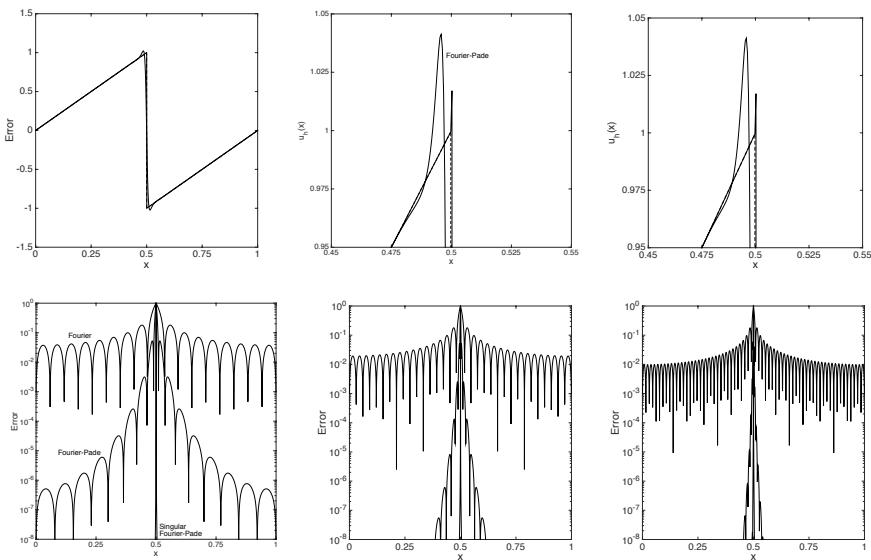
% Remaining polynomial is found using low-order terms
Cq = uhat(1:M+1); Cq(1) = Cq(1)/2; Rq = zeros(1,M+1); Rq(1) = Cq(1);
A = toeplitz(Cq,Rq);
pp = A*qp(1:M+1);
for s = 1:mm
    Lp = toeplitz(lp{s}(1:M+1),[lp{s}(1) zeros(1,R(s))]);
    pp = pp - Lp*rp{s};
end
pm = conj(pp); qm = conj(qp);

% Evaluate singular Pade-Fourier form
qpp = polyval(flipud(qp),xp); qmp = polyval(flipud(qm),xm);
Pu = polyval(flipud(pp),xp)./qpp + polyval(flipud(pm),xm)./qmp;

for s=1:mm
    rpp = polyval(flipud(rp{s}),xp); rpm = polyval(flipud(rm{s}),xm);
    Pu = Pu + rpp.*log(1-xp/xip(s))./qpp + rpm.*log(1-xm/xim(s))./qmp;
end
return

```

To illustrate the impact of the logarithmic term in the reconstruction, let us return to the example discussed previously.



**Figure 13.6.** Illustration of the impact of a singular Fourier-Padé reconstruction on the truncated Fourier approximation of a discontinuous function. In the top row, we show the solution, the Fourier-Padé approximation, and the singular Fourier-Padé processed approximation. The bottom row shows the corresponding pointwise error, including those of the truncated Fourier series. In the left column  $N = 8$ , in the middle column  $N = 16$ , and in the right column  $N = 32$ . The exact function is marked by dashed lines.

**Example 13.34.** We consider the periodic discontinuous function discussed in Ex. 13.32 and use a singular Fourier-Padé reconstruction. No effort has been made to optimize  $M$ ,  $L$ , and  $R$ , and they are all approximately  $N/3$ .

The function, its Fourier-Padé reconstructed approximation, and the reconstruction obtained with the singular Fourier-Padé approach are illustrated in Fig. 13.6. We also show the pointwise errors. When compared to the results in Fig. 13.5, the singular Fourier-Padé reconstruction shows improved pointwise accuracy, and reduced overshoots in the neighborhood of the discontinuity. However, the singular Fourier-Padé reconstruction does not improve the convergence at the point of discontinuity. ■

More advanced applications of the Fourier-Padé reconstruction are demonstrated in [49].

While the Gibbs phenomenon is still present in the Fourier-Padé reconstruction, a direct comparison with the filtered approximation, discussed in subsection 13.4.1, suggests that it is generally superior. This could suggest that a Fourier-Padé reconstruction be applied during the computational stage rather than as a postprocessing technique. Unfortunately, the nonlinearity of the Padé-form makes this approach difficult, due to a lack of robustness in the construction of the Padé-form. In particular, one cannot guarantee that a particular choice of  $(M, L)$  ensures the existence of the Padé-form and, furthermore, one needs a Padé-form such that  $q_L \neq 0$  in the domain of interest. This helps explain why only limited rigorous analysis of the accuracy of the Fourier-Padé reconstruction is available.

These challenges are further enhanced with the introduction of the singular Fourier–Padé-form where one also needs accurate information about the location of the discontinuity. While techniques to address this issue do exist [19, 20], the robustness of the construction of the Padé-form remains a challenge, associated with the nature of the Padé-form. When used as a postprocessing tool, the orders of the polynomials can be adjusted to yield the best result, although limited guidelines for these choices are available [5].

The extension of Padé reconstruction techniques to the polynomial case was initiated in [13, 40, 36] and a detailed analysis of the reconstruction of the Gibbs phenomenon for polynomial expansions is provided in [35].

### 13.4.3 • Overcoming the Gibbs phenomenon

The two postprocessing techniques discussed so far have the potential to substantially improve the accuracy of the global approximation, even when it is contaminated by the Gibbs phenomenon. However, even the singular Fourier–Padé reconstruction does not improve the order of convergence at the discontinuity. To recover a technique that overcomes the Gibbs phenomenon altogether, a more complex approach is required.

Let us consider a function  $u \in L^2([0, L])$  with the Fourier expansion

$$u(x) \simeq u_b(x) = \sum_{n=-N}^N \hat{u}_n \exp\left(in2\pi\frac{x}{L}\right),$$

$$\hat{u}_n = \frac{1}{L} \int_0^L u(x) \exp\left(-in2\pi\frac{x}{L}\right) dx = \frac{1}{L}(u, \phi_n),$$

where  $\phi_n(x) = \exp(in2\pi\frac{x}{L})$  represents the Fourier basis. Here we have used the continuous Fourier coefficients, but the discrete coefficients can likewise be considered.

Let us also assume that  $u$  is piecewise analytic such that there exists an interval  $x \in [a, b] \subset [0, L]$  in which  $u$  is analytic. Following [29], we assume that we can re-expand  $u_b$  as

$$u_b^\lambda(x) = \sum_{m=0}^M (u_b, \psi_m^{(\lambda)})_\lambda \psi_m^{(\lambda)}(r(x)),$$

where  $\psi_m^{(\lambda)}$  is a family of basis functions, orthogonal under the inner product  $(\cdot, \cdot)_\lambda$ , which we shall specify shortly. We also define the local variable

$$x(r) = \frac{1}{2}((b-a)r + (b+a)) = \varepsilon r + \delta, \quad r \in [-1, 1],$$

with  $\varepsilon = \frac{b-a}{2}$  and  $\delta = \frac{b+a}{2}$ . The goal of the subsequent discussion is to establish that  $u_b^\lambda(x)$  converges exponentially fast to  $u(x)$  for  $x \in [a, b]$ , provided the basis  $\psi_m^{(\lambda)}(r(x))$  is chosen to be Gibbs-complementary to  $\phi_n(x)$ . The conditions are stated in the following definition [29].

**Definition 13.35.** *The basis  $\psi_m^{(\lambda)}(r(x))$  is said to be Gibbs-complementary to the basis  $\phi_n(x)$  if the following conditions hold*

1. *The basis  $\psi_m^{(\lambda)}(r(x))$  is orthonormal with respect to the inner product  $(\cdot, \cdot)_\lambda$ , i.e.,*

$$(\psi_i^{(\lambda)}, \psi_j^{(\lambda)})_\lambda = \delta_{ij}.$$

2. The expansion of a function  $u(x(r))$ , which is analytic in  $r \in [-1, 1]$ , in the basis  $\psi_m^{(\lambda)}(r(x))$  converges exponentially fast in  $\lambda$ , i.e., there exists  $C, c \geq 0$  such that

$$\max_{r \in [-1, 1]} \left| u(x(r)) - \sum_{m=0}^{\lambda} (u, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r) \right| \leq C \exp(-c \lambda).$$

3. There exists a number  $\beta < 1$  such that when  $\lambda = \beta N$ , then

$$|(\phi_n, \psi_m^{(\lambda)})_{\lambda}| \max_{r \in [-1, 1]} |\psi_m^{(\lambda)}(\xi)| \leq \left( \frac{\alpha N}{n} \right)^{\lambda},$$

where  $n > N$ ,  $m < \lambda$ , and  $\alpha < 1$ . This condition is referred to as the Gibbs condition.

We observe that the first two conditions guarantee that the complementary basis can accurately approximate an analytic function. On the other hand, the Gibbs condition ensures accuracy of the reprojected function by requiring that the high modes in  $\phi_n$  can be accurately expressed by the low modes in the complementary basis  $\psi_m^{(\lambda)}$ , defined on the interval  $[a, b]$ .

That the identification of a Gibbs-complementary basis suffices to recover an exponentially accurate representation of  $u$  on  $[a, b]$  is expressed in the following result [29].

**Theorem 13.36.** Let  $u \in L^2([0, L])$  be analytic on  $x \in [a, b] \subset [0, L]$  and suppose that we have a basis  $\phi_n(x)$  which is orthonormal in the inner product  $(\cdot, \cdot)$ . Let us also assume that we have a Gibbs-complementary basis,  $\psi_m^{(\lambda)}(r)$ , which is orthonormal in the inner product  $(\cdot, \cdot)_{\lambda}$ . Furthermore, suppose that

$$(u(x) - u_b(x), \psi_m^{(\lambda)}(r))_{\lambda} = \sum_{n=m+1}^{\infty} (u, \phi_n)(\psi_m^{(\lambda)}, \phi_n)_{\lambda}.$$

Then

$$\max_{a \leq x \leq b} \left| u(x) - \sum_{m=0}^{\lambda} (u_b, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right| \leq \exp(-cN), \quad c > 0.$$

**Proof.** Since  $u$  is analytic on  $[a, b]$ , condition 2 of Definition 13.35 guarantees that there exists  $c > 0$  such that

$$\max_{a \leq x \leq b} \left| u(x) - \sum_{m=0}^{\lambda} (u, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right| \leq \exp(-cN),$$

where  $\lambda = \beta N$ . However, since we only have access to  $u_b$  rather than  $u$ , we need to estimate

$$\begin{aligned} \max_{a \leq x \leq b} \left| \sum_{m=0}^{\lambda} (u - u_b, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right| &\leq \max_{a \leq x \leq b} \sum_{m=0}^{\lambda} \sum_{n=N+1}^{\infty} |(u, \phi_n)(\psi_m^{(\lambda)}, \phi_n)_{\lambda}| \psi_m^{(\lambda)}(r(x)) \\ &\leq C \sum_{m=0}^{\lambda} \sum_{n=N+1}^{\infty} \left( \frac{\alpha N}{n} \right)^{\lambda} \leq \exp(-cN). \end{aligned}$$

The last reduction follows from the Gibbs condition. Finally, by the triangle inequality, we have

$$\begin{aligned} & \left| u(x) - \sum_{m=0}^{\lambda} (u_b, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right| \\ & \leq \left| u(x) - \sum_{m=0}^{\lambda} (u, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right| + \left| \sum_{m=0}^{\lambda} (u - u_b, \psi_m^{(\lambda)})_{\lambda} \psi_m^{(\lambda)}(r(x)) \right|, \end{aligned}$$

and the result follows.  $\square$

The theorem establishes that when a Gibbs-complementary basis exists, a slowly converging series in  $\phi_n(x)$  can be expressed as a rapidly converging series in  $\psi_m^{(\lambda)}(x)$  for all segments where  $u$  is analytic. In other words, even if the original approximation  $u_b$  is an inaccurate approximation to  $u$ , it contains all necessary information to recover a piecewise accurate approximation.

The identification of a Gibbs-complementary basis depends on the choice of the original basis  $\phi_n(x)$ . Let us focus on the Fourier basis,  $\phi_n(x) = \exp(inx)$ , and rely on the following result [30, 28].

**Theorem 13.37.** *Let  $\phi_n(x)$  be the Fourier basis. Then a Gibbs-complementary basis is given by*

$$\psi_m(r) = \frac{1}{\sqrt{\gamma_m^{(\lambda)}}} \tilde{C}_m^{(\lambda)}(r), \quad r \in [-1, 1],$$

where  $\tilde{C}_m^{(\lambda)}$  is the Gegenbauer polynomial of order  $m$  and type  $\lambda$ , and

$$\gamma_m^{(\lambda)} = (\tilde{C}_m^{(\lambda)}, \tilde{C}_m^{(\lambda)})_{\lambda} = \frac{\pi 2^{1-2\lambda} \Gamma(m+2\lambda)}{m!(m+\lambda)\Gamma^2(\lambda)}.$$

Here  $\Gamma(x)$  is the Gamma function.

Before proceeding, we briefly discuss the Gegenbauer polynomials  $\tilde{C}_m^{(\lambda)}$ . These are defined as the polynomial solution to the singular Sturm–Liouville problem

$$\frac{d}{dr} (1-r^2)^{\lambda+\frac{1}{2}} \frac{d\tilde{C}_m^{(\lambda)}(r)}{dr} + m(m+2\lambda)(1-r^2)^{\lambda-\frac{1}{2}} \tilde{C}_m^{(\lambda)}(r) = 0.$$

This ensures that the expansion of an analytic function, defined on  $[-1, 1]$ , is exponentially convergent [34]. We consider the orthonormal Gegenbauer polynomials  $C_m^{(\lambda)}$ :

$$C_m^{(\lambda)}(r) = \frac{1}{\sqrt{\gamma_m^{(\lambda)}}} \tilde{C}_m^{(\lambda)}(r).$$

These can be evaluated using the three-term recurrence formula

$$r C_m^{(\lambda)}(r) = \alpha_m C_{m-1}^{(\lambda)}(r) + \alpha_{m+1} C_{m+1}^{(\lambda)}(r),$$

with the coefficient

$$\alpha_m = \frac{1}{2} \sqrt{\frac{m(m+2\lambda-1)}{(m+\lambda)(m+\lambda-1)}}$$

and the initial values

$$C_0^{(\lambda)}(r) = \sqrt{\frac{\Gamma(\lambda+1)}{\sqrt{\pi}\Gamma(\lambda+\frac{1}{2})}}, \quad C_1^{(\lambda)}(r) = \sqrt{\frac{2\Gamma(\lambda+2)}{\sqrt{\pi}\Gamma(\lambda+\frac{1}{2})}}r.$$

It is worth noting that for  $\lambda = \frac{1}{2}$  we recover the Legendre polynomials, discussed at length in subsection 12.1.1. The evaluation of the normalized Gegenbauer polynomials is illustrated in `GegenbauerP.m`.

**Script 13.7. *GegenbauerP.m*: Routine to evaluate Gegenbauer polynomial of order  $N$  at point  $r \in [-1, 1]$ .**

---

```

function [C] = GegenbauerP(r, lambda, N);
% function [C] = GegenbauerP(r, lambda, N)
% Purpose: Evaluate Gegenbauer polynomial of type lambda > -1/2
%           points r for order N and returns P[1:length(r)]
% Note     : They are normalized to be orthonormal.

% Turn points into row if needed.
xp = r; dims = size(xp);
if (dims(2)==1) xp = xp'; end;

CL = zeros(N+1,length(xp));
% Initial values C_0(x) and C_1(x)
gamma0 = sqrt(pi)*gamma(lambda+0.5)/gamma(lambda+1);
CL(1,:) = 1.0/sqrt(gamma0);
if (N==0) C=CL'; return; end;
gamma1 = (lambda+1/2)^2/(2*lambda+2)*gamma0;
CL(2,:) = (2*lambda+1)*xp/(2*sqrt(gamma1));
if (N==1) C=CL(N+1,:); return; end;

% Repeat value in recurrence.
aold = 2/(2*lambda+1)*sqrt((lambda+1/2)^2/(2*lambda+2));

% Forward recurrence using the symmetry of the recurrence.
for i=1:N-1
    h1 = 2*i+2*lambda-1;
    anew = 2/(h1+2)*sqrt( (i+1)*(i+2*lambda)*(i+lambda+1/2)^2/(h1+1)/(h1+3));
    CL(i+2,:) = 1/anew*( -aold*CL(i,:) + xp.*CL(i+1,:));
    aold = anew;
end;
C = CL(N+1,:)';
return

```

---

With this in place, we need to show that the Gegenbauer polynomial forms a Gibbs complementary basis to the Fourier basis, i.e., we need to establish the three conditions in Definition 13.35.

The orthonormality of the Gegenbauer basis follows directly from its definition as an eigensolution to the Sturm–Liouville problem and the normalization, hence

establishing the first condition. The second condition requires us to analyze the quantity

$$\max_{r \in [-1,1]} \left| u(x(r)) - \sum_{m=0}^M \hat{u}_m^\lambda C_m^{(\lambda)}(r) \right|,$$

where

$$\hat{u}_m^\lambda = \int_{-1}^1 u(r) C_m^{(\lambda)}(r) (1-r^2)^{\lambda-1/2} dr$$

are the Gegenbauer expansion coefficients. While this might at first appear as a standard approximation question, we observe that Definition 13.35 requires spectral convergence in  $\lambda$  as opposed to convergence in  $M$  for fixed  $\lambda$ , as is traditionally considered.

When  $u$  is analytic, there exists  $\rho \geq 1$  such

$$\max_{r \in [-1,1]} \left| \frac{d^l u}{d r^l} \right| \leq C(\rho) \frac{l!}{\rho^l}.$$

Recalling the properties of the Gegenbauer polynomials, one proves that [30, 34]

$$\max_{r \in [-1,1]} \left| u(x(r)) - \sum_{m=0}^M \hat{u}_m^\lambda C_m^{(\lambda)}(r) \right| \leq K \frac{C(\rho) \Gamma(\lambda + \frac{1}{2}) \Gamma(M+1+2\lambda)}{M \sqrt{\lambda} (2\rho)^M \Gamma(2\lambda) \Gamma(M+\lambda)}.$$

Using Stirling's formula, the estimate

$$\max_{r \in [-1,1]} \left| u(x(r)) - \sum_{m=0}^M \hat{u}_m^\lambda C_m^{(\lambda)}(r) \right| \leq A q^M$$

follows. Since  $\rho \geq 1$ ,

$$q = \frac{(1+2\gamma)^{1+2\gamma}}{\rho^{2+2\gamma} \gamma^\gamma (1+\gamma)^{1+\gamma}} \leq 1.$$

Here we have assumed that  $\lambda = \gamma M$ , i.e., the type of the Gegenbauer polynomial must increase as  $M$  increases, to guarantee that the second condition of Definition 13.35 is fulfilled. It is noteworthy that the discussion is so far independent of the fact that we use a Fourier basis. In other words, we have established the validity of the first two condition for any basis  $\phi_n(x)$ .

Finally, to establish the Gibbs condition, we must prove that

$$\left| \int_{-1}^1 \exp(i n \pi r) C_m^{(\lambda)}(r) (1-r^2)^{\lambda-1/2} dr \right| \leq \left( \frac{\alpha N}{n} \right)^\alpha,$$

where  $n > N$ ,  $m \leq \lambda = \beta N$ , and  $x \in [0, L]$ . This is done by recognizing that [2]

$$\int_{-1}^1 \exp(i n \pi r) C_m^{(\lambda)}(r) (1-r^2)^{\lambda-1/2} dr = \sqrt{\gamma_m^\lambda} \Gamma(\lambda) \left( \frac{2}{n \pi} \right)^\lambda i^m (m+\lambda) J_{m+\lambda}(n \pi),$$

where  $J_\nu$  is the Bessel function of the first kind. Using Stirling's formula, one obtains that  $\beta \geq 2\pi/27$  is sufficient to ensure that the Gibbs condition is fulfilled. Thus, it is

possible to recover a piecewise spectrally accurate solution from the truncated Fourier expansion of a piecewise analytic solution, assuming only that we know the location of discontinuities. In other words, it is possible to completely overcome the Gibbs phenomenon.

To evaluate the reconstructed solution, we need to evaluate

$$u_b^\lambda(x(r)) = \sum_{m=0}^M \hat{u}_m^\lambda C_m^{(\lambda)}(r), \quad x \in [a, b],$$

where

$$\hat{u}_m^\lambda = \hat{u}_0 \delta_{0m} + \Gamma(\lambda) i^m (m + \lambda) \sqrt{\gamma_m^\lambda} \sum_{\substack{n=-N \\ n \neq 0}}^N e^{in\delta} \left( \frac{2}{n\varepsilon} \right)^\lambda J_{m+\alpha}(n\varepsilon),$$

with

$$\varepsilon = \frac{b-a}{2}, \quad \delta = \frac{a+b}{2}.$$

The implementation of the Gegenbauer reconstruction is illustrated in `GegenbauerRecon.m`.

**Script 13.8. *GegenbauerRecon.m: Routine to reconstruct a piecewise analytic function, expressed by its truncated Fourier expansion, using a Gegenbauer reconstruction.***

---

```

function [uRecon] = GegenbauerRecon(x,u,xp,xeval,N,Nc);
% function [uRecon] = GegenbauerRecon(x,u,xp,xeval,N,Nc);
% Purpose: Gegenbauer reconstruction of u with points of discontinuity in
% vector xp. Recobstructed at xeval xp(1)/xp(end) = left/right point of
% domain
% Only Nc first terms are used.
Nseg = length(xp)-1; ii=sqrt(-1); Llen = xp(end)-xp(1);
xleval = length(xeval); uRecon = zeros(xleval,1);
uhat = fft(u)/(2*N+1);
if (Nc>N)
    uhat(Nc+2:2*N+1-Nc)=0;
end

% Set parameters for the Gegenbauer construction
lambda = floor(sqrt(Nc)); M = floor(sqrt(Nc));

% Postprocess each segment
for ns=1:Nseg
    a = xp(ns); b = xp(ns+1);
    epsh=(b-a)/(2*pi); deltah = (a+b)/(2*pi);
    id = find((xeval>=a)&(xeval<=b));
    xl = xeval(id); xlen = length(id); r = -1 + 2*(xl-a)/(b-a);

    Cmat = zeros(xlen,M+1); b = zeros(M+1,1);
    for m=0:M
        Cmat(:,m+1) = GegenbauerP(r,lambda,m);
        gammaG = pi*2^(1-2*lambda)*gamma(m+2*lambda) / ...
            (gamma(m+1)*(m+lambda)*gamma(lambda)^2);
        for n=1:N
            b(m+1) = b(m+1)+uhat(n+1)*(2/(n*epsh*pi))^lambda * ...

```

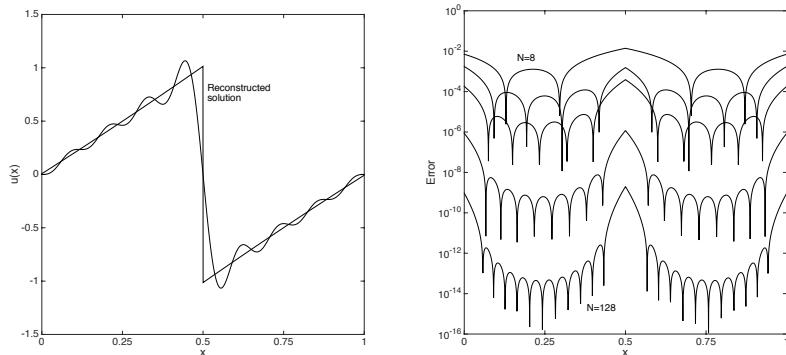
```

besselj (m+lambda , n*epsh*pi )*exp ( ii *pi*n*delthah ) ;
b (m+1) = b (m+1)+uhat (2*N+2-n)*(2/(-n*epsh*pi ))^lambda *...
besselj (m+lambda , -n*epsh*pi )*exp (- ii *pi*n*delthah ) ;
end
b (m+1) = sqrt (gammaG )*gamma (lambda )*ii ^m*(m+lambda )*b (m+1) ;
end
gammaG0 = pi*2^(1-2*lambda )*gamma (2*lambda )/(lambda *gamma (lambda )^2) ;
b (1) = b (1)+uhat (1)*sqrt (gammaG0 ) ;
uRecon (id) = Cmat*b ;
end
return

```

**Example 13.38.** We consider the discontinuous, periodic function, discussed in Ex. 13.32, and apply a Gegenbauer reconstruction to the truncated expansion.

Since the exact location of the discontinuity is known, we proceed by considering the reconstruction in two parts, separated at  $x = 0.5$ . The results are illustrate in Fig. 13.7 and confirm the exponential convergence of the reconstructed solution, including at the point of discontinuity. In this case we use  $M = \lambda = \sqrt{N}$ , but other choices are possible. ■



**Figure 13.7.** Gegenbauer reconstruction of truncated Fourier expansion to eliminate the Gibbs phenomenon. On the left we show the truncated Fourier representation of the test function for  $N = 8$  as well as the reconstructed function. On the right we show the pointwise accuracy for  $N = 8, 16, 32, 64, 128$  with  $M = \lambda = \sqrt{N}$ .

The development of Gegenbauer-based reconstruction techniques that overcome the Gibbs phenomenon on global expansions was initiated in [30], and subsequently extended to orthogonal polynomials and discrete expansions in a series of papers [27, 25, 26]. A review is given in [28], and the general setting discussed above was introduced in [29]. Issues related to the proper choice of  $M$  and  $\lambda$  are discussed at length in [38, 21, 18, 39] and an alternative formulation is considered in [52]. A comprehensive review of these and other techniques for the accurate recovery is available in [60], and applications of the Gegenbauer reconstruction techniques can be found in [17, 1, 48, 23]. The use of Gegenbauer reconstructions in WENO methods is discussed in [31].

Returning to the results of Fig. 13.7, we observe that whereas the reconstructed solution clearly recovers high accuracy, the error is dominated by contributions from

the ends of the segments of analyticity. While one could suspect that this is caused by the Gibbs phenomenon around  $x = 0.5$ , this would not explain similar behavior near  $x = 0$ , where no Gibbs phenomenon is present.

As realized in [4], this behavior is caused by a Runge-like phenomenon, in the limit when both  $\lambda$  and  $M$  increase, as required to ensure spectral convergence for increasing values of  $N$ . This observation suggests that improved accuracy could be obtained through the use of a Padé-form [44].

To this end, we seek a Padé expansion

$$u_b^\lambda(x(r)) = \frac{p_M(r)}{q_L(r)} + \mathcal{O}(r^{N+1})$$

of the locally reconstructed solution  $u_b^\lambda(x(r))$ . To keep things simple we assume that  $N = M + L$ , although this is not required. If we assume that

$$p_M(r) = \sum_{m=0}^M \hat{p}_m C_m^\lambda(r), \quad q_L(r) = \sum_{l=0}^L \hat{q}_l C_l^\lambda(r),$$

and require

$$(p_M(r) - q_L(r)u_b^\lambda(x(r)), C_k^{(\lambda)})_\lambda = 0 \quad \forall k = 0 \dots N,$$

we recover the conditions

$$k = M + 1 \dots N : \sum_{l=0}^L \hat{q}_l H_{lk} = 0,$$

$$k = 0 \dots M : \sum_{l=0}^L \hat{q}_l H_{lk} = \hat{p}_k.$$

The entries of  $H$  are defined as

$$H_{lk} = \int_{-1}^1 u_b^\lambda(x(r)) C_l^\lambda(r) C_k^\lambda(r) (1 - r^2)^{\lambda - 1/2} dr,$$

and can be evaluated to adequate accuracy by a Gaussian quadrature as illustrated in GegenbauerGQ.m using the standard algorithm of [22].

---

**Script 13.9. GegenbauerGQ.m: Routine to compute Gaussian quadrature for the Gegenbauer polynomial of type  $\lambda$ .**

---

```
function [x,w] = GegenbauerGQ(lambda,N);
% function [x,w] = GegenbauerGQ(lambda,N);
% Purpose: Compute the N'th order Gauss quadrature points, x,
%           and weights, w, associated with the Gegenbauer
%           polynomial, of type lambda > -1/2
if (N==0) x(1)=0; w(1) = 2; return; end;

% Form symmetric matrix from recurrence.
J = zeros(N+1);
h1 = 2*(0:N)+2*lambda-1;
J = diag(2./(h1(1:N)+2).*sqrt((1:N).*((1:N)+2*lambda-1).*...)
```

---

```

((1:N)+lambda-0.5).*((1:N)+lambda-0.5)./(h1(1:N)+1)./(h1(1:N)+3)),1);

if (2*lambda-1<10*eps) J(1,1)=0.0;end;
if (abs(lambda)<10*eps) J(1,2) = lambda+0.5;end;
J = J + J';

%Compute quadrature by eigenvalue solve
[V,D] = eig(J); x = diag(D);
w = (V(1,:')).^2*2^(2*lambda)*gamma(lambda+1/2)^2/gamma(2*lambda+1);
return;

```

---

We can now recover the Padé-form and evaluate it as illustrated in *GegenbauerPade.m*.

**Script 13.10. *GegenbauerPade.m*: Routine to post-process a Gegenbauer expansion to improve accuracy.**

---

```

function [uPade] = GegenbauerPade(b,r,lambda,N,M,L);
% function [uPade] = GegenbauerPade(b,x,N,M,L);
% Purpose: Express Gegenbauer expansion as Pade form to suppress
% Runge phenomenon in diagonal limit
NQ = 2*N; [xG,wG] = GegenbauerGQ(lambda,NQ);

% Evaluate Gegenbauer polynomials at nodes to order 2*N
Cmat = zeros(NQ+1,N+1);
for m=0:N
    Cmat(:,m+1) = GegenbauerP(xG,lambda,m);
end

% Evaluate function at quadrature points
u = Cmat*b;

% Set up coefficient matrix
H = Cmat'*diag(u.*wG)*Cmat;

% Compute coefficient for q
Hq = H(M+2:N+1,1:L+1);
Z = null(Hq); q = Z(:,end); q = q/min(q);

% Compute coefficients for p
p = H(1:M+1,1:L+1)*q;

% Evaluate Pade form at r
rlen = length(r); Cmat = zeros(rlen,N+1);
for m=0:N
    Cmat(:,m+1) = GegenbauerP(r,lambda,m);
end

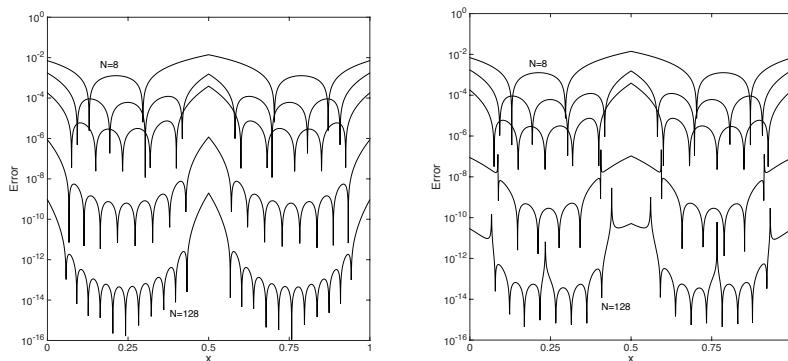
pG = Cmat(:,1:M+1)*p; qG = Cmat(:,1:L+1)*q;
uPade = pG./qG;
return

```

---

In Fig. 13.8, we compare the results of Fig. 13.7 with those obtained by post-processing the Gegenbauer expansion, illustrating the potential to recover enhanced accuracy close to the ends of the segments of analyticity. The enhancement is most pronounced for higher values of  $M$  and  $\lambda$ , as one would expect.

We used  $M = N/2 - 1$  and  $L = N/2 + 1$ , and no attempt was made to optimize these values. As discussed at length in [44], there is flexibility in these choices, and the resulting accuracy may vary substantially.



**Figure 13.8.** Gegenbauer reconstruction of a truncated Fourier expansion to eliminate the Gibbs phenomenon. On the left we show the pointwise error of the Gegenbauer reconstructed function and on the right we show the result of the Padé reconstruction of the Gegenbauer expansion. Results are for  $N = 8, 16, 32, 64, 128$ , with  $M = \lambda = \sqrt{N}$ .

## 13.5 • Spectral methods in action

In the following, we explore the performance of Fourier spectral methods for conservation laws through a number of examples. Before proceeding, let us briefly consider the choice of the time step, based on CFL-like considerations. For the standard finite difference scheme, we recall that

$$|\alpha| \frac{k}{h} \leq 1$$

is a natural condition for a local method. Here  $\alpha$  is the maximum phase velocity. For the Fourier method, we realize that

$$|\alpha| \lambda_{\max} k \leq 1 \Rightarrow |\alpha| \frac{k}{h} \leq \pi^{-1},$$

where  $\lambda_{\max}$  is the maximum eigenvalue of the Fourier differentiation matrix. Hence, there is a penalty of  $\pi^{-1}$  in the time step, as compared to a local finite difference method. Nevertheless, to achieve a comparable accuracy, a Fourier spectral method typically allows a much larger value of  $h$  than in a finite difference methods, and the factor of  $\pi^{-1}$  has limited consequences.

### 13.5.1 • Burgers' equation

Let us first consider Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \quad x \in [0, 1],$$

subject to periodic boundary conditions and the initial condition  $u(x, 0) = \sin(2\pi x)$ .

In `BurgersSpecDriver1D.m` we show the driver that defines the initial conditions and set parameters for the problems.

---

**Script 13.11. *BurgersSpecDriver1D.m*: Driver routine for a spectral Fourier collocation method to solve Burgers' equation.**

---

```
% Driver script for solving the 1D Burgers equations using a Fourier
% spectral collocation method
clear all

% Set problem parameters
FinalTime = 0.3; N = 256; CFL = 1.0;
xmin = 0.0; xmax = 1.0;

% Define domain and initial conditions
L = xmax-xmin;
x = xmin + L/(2*N+1)*[0:2*N]';
fic = @(x,t) sin(2*pi*x);

% Compute initial conditions
u = fic(x,0);

% Solve Problem
[u] = BurgersSpec1D(x,u,N,L,CFL,FinalTime);
```

---

This routine calls BurgersSpec1D.m, which performs the temporal integration using a third order SSPERK scheme and introduces vanishing viscosity through a filter, if needed.

---

**Script 13.12. *BurgersSpec1D.m*: Temporal integration of a spectral Fourier collocation method to solve Burgers' equation using an SSPERK scheme of order three. Vanishing hyperviscosity is added.**

---

```
function [u] = BurgersSpec1D(x,u,N,L,CFL,FinalTime);
% function [u] = BurgersSpec1D(x,u,N,L,CFL,FinalTime);
% Purpose : Integrate 1D Burgers equation until FinalTime using a Fourier
% spectral collocation method and 3rd order SSP-RK method
time = 0; tstep = 0; h = L/(2*N+1);

% Parameter for hyper viscosity
p = 16;

% integrate scheme
while (time<FinalTime)
    % Decide on timestep
    maxvel = max(2*abs(u)); k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end

    % Update solution
    rhsu = BurgersSpecrhs1D(x,u,L); u1 = u + k*rhsu;
    [u1] = FourierVanishHypVisc(u1,p,k,N,L);
    rhsu = BurgersSpecrhs1D(x,u1,L); u2 = (3*u + u1 + k*rhsu)/4;
    [u2] = FourierVanishHypVisc(u2,p,k,N,L);
    rhsu = BurgersSpecrhs1D(x,u2,L); u = (u + 2*u2 + 2*k*rhsu)/3;
    [u] = FourierVanishHypVisc(u,p,k,N,L);
    time = time+k; tstep = tstep+1;
end
return
```

---

The spatial approximation of the Burgers' equation is performed in BurgersSpecrhs1D.m. Both a direct approximation and the skew-symmetric form are considered.

**Script 13.13. BurgersSpecrhs1D.m: Evaluation of the spatial approximation of Burgers' equation using a spectral Fourier collocation method. Both a direct and a skew-symmetric form are available.**

---

```
function [du] = BurgersSpecrhs1D(x,u,L)
% function [du] = BurgersSpecrhs1D(x,u,L)
% Purpose : Evaluate the RHS of Burgers equations using spectral Fourier
% collocation method

% Compute residual - direct form
du = - 2*pi/L*Fourierdx(u.^2,1);

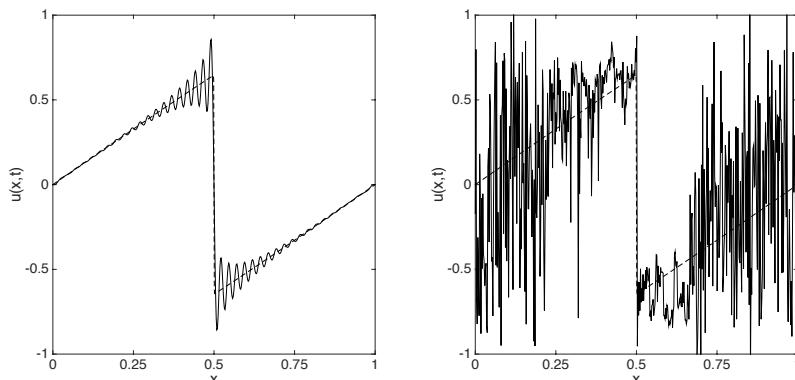
% Compute residual - skew symmetric form
% du = - 2*pi/L*2/3*(u.*Fourierdx(u,1) + Fourierdx(u.^2,1));
return
```

---

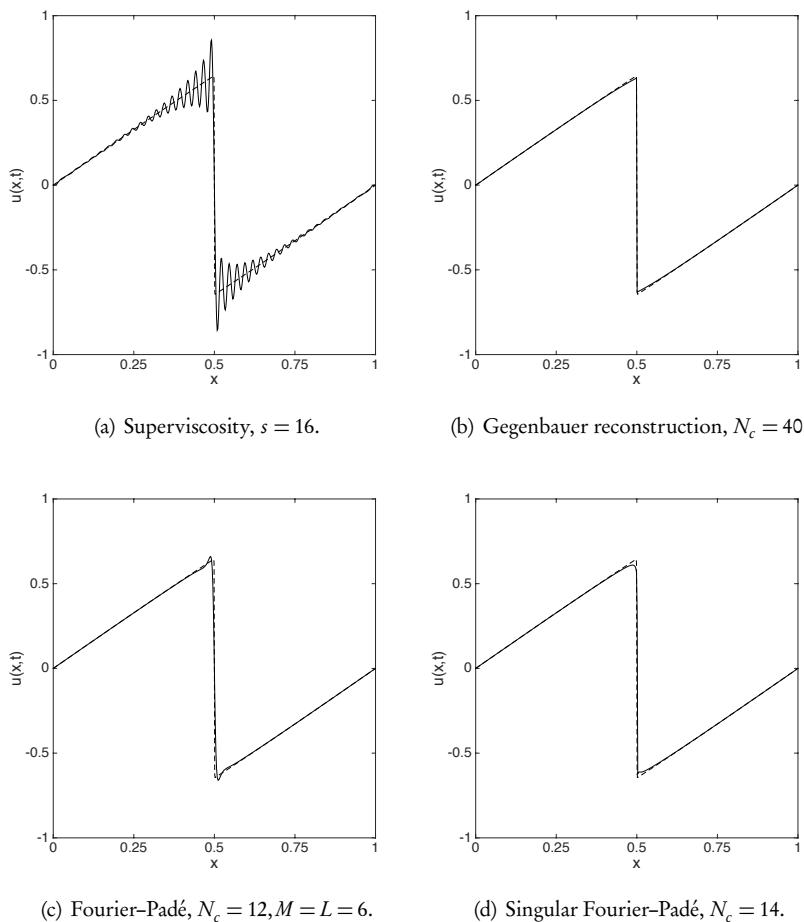
We begin by comparing the results obtained with  $N = 256$  and a stabilized direct approach to those computed with a stable skew-symmetric scheme. The results are shown in Fig. 13.9. For the stabilized scheme, we clearly identify the Gibbs oscillations in the neighborhood of the shock at  $x = 0.5$  but also note the very sharp representation of the shock. In contrast to this case in which additional dissipation is needed to ensure stability, the skew-symmetric form remain stable even without dissipation. However, the quality of the solution is very poor, emphasizing that an energy/entropy conservative scheme is not suitable for problems with shocks.

We investigate the different post-processing techniques discussed in section 13.4 by comparing the result of their application to this example, illustrated in Fig. 13.10. In general, it is important not to use all expansion coefficients, as the high modes are likely to be contaminated. We typically use only  $|n| \leq N_c < N$  modes, where the choice of  $N_c$  depends on the post-processing technique.

As expected, we observe that the Gegenbauer post-processing yields an excellent solution and eliminates the Gibbs oscillations entirely. We also find that both of the



**Figure 13.9.** Burgers' equation solved using a Fourier collocation method with  $N = 256$ . The solution (solid) is shown at  $T = 0.3$  and compared with a high-resolution solution (dashed), obtained with an ENO scheme. On the left we show the solution obtained with the direct formulation of the scheme, stabilized with a vanishing hyperviscosity ( $s = 16$ ) term. On the right we show the result obtained using the skew-symmetric form without any additional dissipation.



**Figure 13.10.** Comparison of different postprocessing techniques for Burgers' equation solved using a Fourier collocation method with  $N = 256$  at  $T = 0.3$  and compared with high-resolution solution (dashed) obtained with an ENO scheme.

Padé-based post-processing techniques yield excellent results as compared to the original solution. For the Padé-based methods, the choice of parameters is important. For a different problem, different choices may be needed to recover an accurate post-processed solution.

While the results in Fig. 13.10 show a visible improvement in the accuracy of the solution, little is known about the formal accuracy of the solution. For Burgers' equation, [53] presents a detailed study which suggests spectral accuracy after Gegenbauer post-processing. Qualitative comparisons between postprocessed spectral methods and ENO schemes [11] suggest excellent accuracy also for complex cases. However, [14] suggests that once a shock has passed, only first order accuracy is available. This latter study, however, focuses on pointwise accuracy, and the accuracy of the moments remains unknown.

### 13.5.2 • Maxwell's equations

Let us now discuss the solution of Maxwell's equations using a spectral Fourier collocation scheme. We recall that the one-dimensional Maxwell's equations

$$\varepsilon(x) \frac{\partial E}{\partial t} = \frac{\partial H}{\partial x}, \quad \frac{\partial H}{\partial t} = \frac{\partial E}{\partial x}, \quad x \in [-2, 2],$$

assume the solution is periodic and define the material coefficient as

$$\varepsilon(x) = \begin{cases} \varepsilon, & |x| \leq 1, \\ 1, & 1 < |x| \leq 2. \end{cases}$$

This is a periodic extension of the nonperiodic test case discussed in subsection 1.4.1 and the exact solution is given as

$$\begin{aligned} E(x, t) &= [A_k e^{i n_k \omega z} - B_k e^{-i n_k \omega z}] e^{-i \omega t}, \\ H(x, t) &= n_k [A_k e^{i n_k \omega z} + B_k e^{-i n_k \omega z}] e^{-i \omega t}, \end{aligned} \quad (13.29)$$

where  $k = 1, 2, 3$  refer to the three regions, and  $n_1 = n_3 = 1, n_2 = n = \sqrt{\varepsilon}$  is the index of refraction. The coefficients are given as

$$A_1 = B_3 = \frac{n \cos(n\omega)}{\cos \omega}, \quad A_2 = B_2 = e^{i2\omega}, \quad A_3 = B_1 = A_1 e^{-i4\omega},$$

where the frequency  $\omega$  is found as the solution to

$$n \tan \omega = -\tan(n\omega).$$

Finally, we note that the solution is continuous, but has a discontinuous derivative at the material interfaces  $|x| = 1$ .

In `MaxwellSpecDriver1D.m` we show the driver which defines the initial conditions and sets the parameters for the problems. The routine to compute the exact solution is not shown.

**Script 13.14. `MaxwellSpecDriver1D.m`: Driver routine for a spectral Fourier collocation method to solve Maxwell's equations with periodic boundary conditions.**

---

```
% Driver script for solving the 1D Maxwell's equations using a spectral
% Fourier collocation scheme
clear all

% Set problem parameters
FinalTime = pi/2; N = 64; CFL = 0.1;
xmin = -2.0; xmax = 2.0;

% Define domain and initial conditions
L = xmax - xmin; x = xmin + L/(2*N+1)*[0:2*N]';
epl = 1.0; mul = 1.0; epr = 2.25; mur = 1.0;

[Ef, Hf, ep, mu] = CavityExactLong(x, epl, epr, mul, mur, 0);

% Solve Problem
q = [Ef Hf];
[q] = MaxwellSpec1D(x, q, ep, mu, N, L, CFL, FinalTime);
```

---

This routine calls `MaxwellSpec1D.m`, which performs the temporal integration using a third order SSPERK scheme.

**Script 13.15. *MaxwellSpec1D.m: Temporal integration of a spectral Fourier collocation method to solve Maxwell's equations using a third order SSPERK scheme.***

---

```
function [q] = MaxwellSpec1D(x, q, ep, mu, N, L, CFL, FinalTime)
% function [q] = MaxwellSpec1D(x, q, ep, mu, N, L, CFL, FinalTime)
% Purpose : Integrate 1D Maxwell's equation until FinalTime using an
% spectral Fourier collocation scheme and 3rd order SSP-RK method.
time = 0; tstep = 0; h = L/(2*N+1);

% Set timestep
cvel = 1./sqrt(ep.*mu); k = CFL*h/max(cvel);

% integrate scheme
while (time < FinalTime)
    if (time+k > FinalTime) k = FinalTime-time; end

    % Update solution
    rhsq = MaxwellSpecrhs1D(x, q, ep, mu, L);
    q1 = q + k*rhsq;
    rhsq = MaxwellSpecrhs1D(x, q1, ep, mu, L);
    q2 = (3*q + q1 + k*rhsq)/4;
    rhsq = MaxwellSpecrhs1D(x, q2, ep, mu, L);
    q = (q + 2*q2 + 2*k*rhsq)/3;
    time = time+k;
end
return
```

---

The spatial approximation of Maxwell's equations is performed in `MaxwellSpecrhs1D.m`.

**Script 13.16. *MaxwellSpecrhs1D.m: Evaluation of spatial approximation of Maxwell's equations using a spectral Fourier collocation scheme.***

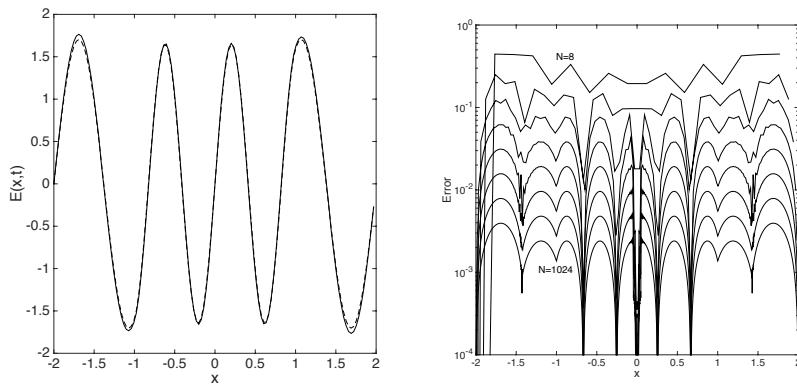
---

```
function [dq] = MaxwellSpecrhs1D(x, q, ep, mu, L);
% function [dq] = MaxwellSpecrhs1D(x, q, ep, mu, L)
% Purpose: Evaluate right hand side for Maxwell's equation using spectral
% Fourier collocation method

N = length(x); dq = zeros(N, 2);
dq(:, 1) = 2*pi/L*Fourierdx(q(:, 2), 1) ./ ep;
dq(:, 2) = 2*pi/L*Fourierdx(q(:, 1), 1) ./ mu;
return
```

---

We solve Maxwell's equations with  $\epsilon = 2.25$  in the central material, using a spectral Fourier collocation method. Figure 13.11 shows the results obtained at  $T = \pi/2$ , along with the exact solution. We note that we only recover first order pointwise accuracy due to the lack of smoothness. In agreement with the discussion in [14], the limited regularity of the solution propagates, and eventually dominates the global error. As this is a consequence of propagation, the application of postprocessing will not quantitatively improve on this issue. However, it is possible that the more advanced techniques for recovering accurate moments, discussed in subsection 12.2.1, can still be successfully applied, although this remains unexplored.



**Figure 13.11.** Maxwell's equations solved using a Fourier collocation method with  $N = 64$  (left). The solution (solid) is shown at  $T = \pi/2$  and compared with the exact solution (dashed). On the right we show the pointwise error for  $N = 2^p$ ,  $p = 3 - 10$ , suggesting first order accuracy.

### 13.5.3 • Euler equations

As a final example, let us consider the solution of the Euler equations using a spectral Fourier collocation method. The one-dimensional Euler equations are

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f} = 0,$$

where the conserved variables  $\mathbf{q}$  and the flux  $\mathbf{f}$  are defined as

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (E + p)u \end{bmatrix}.$$

The equations are closed by the ideal gas law

$$p = (\gamma - 1) \left( E - \frac{1}{2} \rho u^2 \right), \quad c = \sqrt{\frac{\gamma p}{\rho}},$$

where  $c$  represents the local sound speed and  $\gamma$  is a fluid-dependent constant which we take to be  $\gamma = 7/5$ .

We assume all variables to be periodic, and modify Sod's problem, defined in subsection 1.4.1, yielding the initial condition

$$\rho(x,0) = \begin{cases} 1.000 & |x| > 0.5, \\ 0.125 & |x| \leq 0.5, \end{cases} \quad \rho u(x,0) = 0, \quad E(x,0) = \frac{1}{\gamma - 1} \begin{cases} 1.0, & |x| > 0.5, \\ 0.1, & |x| \leq 0.5, \end{cases}$$

with the computational domain being  $x \in [-1, 1]$ .

In EulerSpecDriver1D.m, we show the driver which defines the initial condition and parameters for the problem.

**Script 13.17. EulerSpecDriver1D.m: Driver routine for a spectral Fourier collocation method to solve the Euler equations with periodic boundary conditions.**

---

```
% Driver script for solving the 1D Euler equations using a spectral Fourier
% collocation method
clear all

% Set problem parameters
FinalTime = 0.2; N = 512; CFL = 1.0; gamma=1.4;
xmin = -1.0; xmax = 1.0;

% Define domain and initial conditions
L = xmax-xmin; x = xmin + L/(2*N+1)*[0:2:N]';

% Define domain, materials and initial conditions
r = zeros(2*N+1,1); ru = zeros(2*N+1,1); E = zeros(2*N+1,1);

% Initialize for Sod's problem
r = (abs(x)>0.5) + (abs(x)<=0.5)*0.125;
E = ((abs(x)>0.5) + (abs(x)<=0.5)*0.1)/(gamma-1);

% Solve Problem
q = zeros(2*N+1,3); q(:,1)=r; q(:,2)=ru; q(:,3)=E;
[q] = EulerSpec1D(x,q,N,L,CFL,gamma,FinalTime);
```

---

This routine calls EulerSpec1D.m, which performs the temporal integration using a third order SSPERK scheme and applies vanishing hyperviscosity for stability.

**Script 13.18. EulerSpec1D.m: Temporal integration of a spectral Fourier collocation method to solve the Euler equations using a third order SSPERK scheme.**

---

```
function [q] = EulerSpec1D(x,q,N,L,CFL,gamma,FinalTime);
% function [q] = EulerSpec1D(x,q,N,L,CFL,gamma,FinalTime);
% Purpose : Integrate 1D Euler equations until FinalTime using a Fourier
% spectral collocation method and 3rd order SSP-RK method
time = 0; tstep = 0; h = L/(2*N+1);

% Parameter for hyper viscosity
p = 8;

% integrate scheme
while (time<FinalTime)
    % Decide on timestep
    pres = (gamma-1)*(q(:,3) - 0.5*q(:,2).^2./q(:,1));
    c = sqrt(gamma*pres./q(:,1)); maxvel = max(max(c+abs(q(:,2)./q(:,1)))); 
    k = CFL*h/maxvel;
    if (time+k>FinalTime) k = FinalTime-time; end

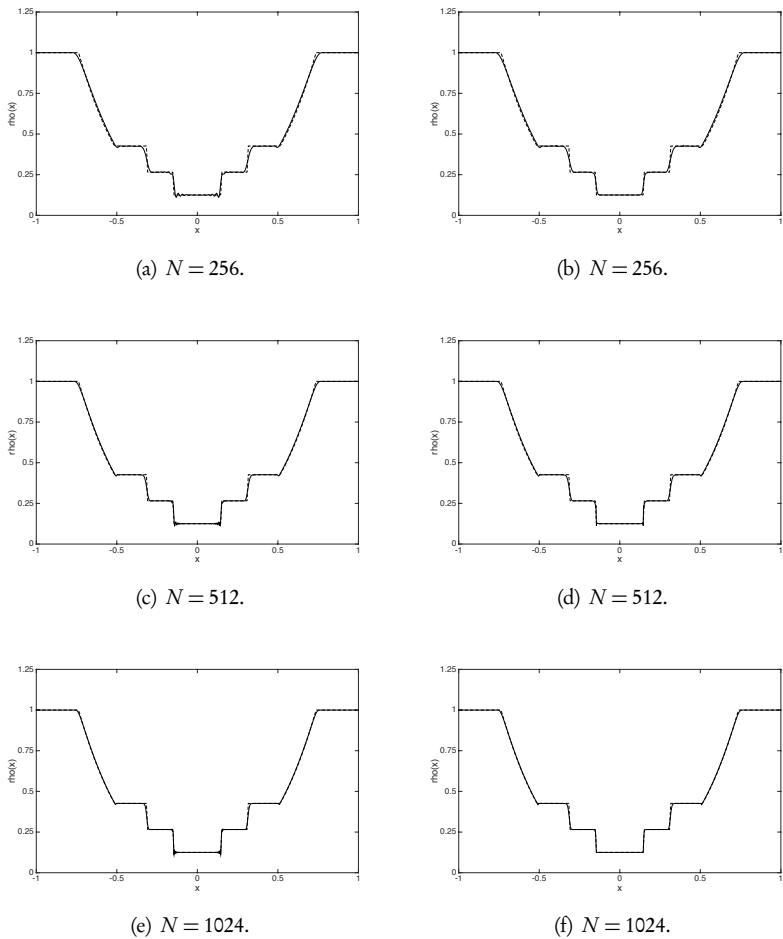
    % Update solution
    rhsq = EulerSpecrhs1D(x,q,L,gamma);
    q1 = q + k*rhsq;
    [q1(:,1)] = FourierVanishHypVisc(q1(:,1),p,k,N,L);
    [q1(:,2)] = FourierVanishHypVisc(q1(:,2),p,k,N,L);
    [q1(:,3)] = FourierVanishHypVisc(q1(:,3),p,k,N,L);
    rhsq = EulerSpecrhs1D(x,q1,L,gamma);
```

---

```

q2 = (3*q + q1 + k*rhsq)/4;
[q2(:,1)] = FourierVanishHypVisc(q2(:,1),p,k,N,L);
[q2(:,2)] = FourierVanishHypVisc(q2(:,2),p,k,N,L);
[q2(:,3)] = FourierVanishHypVisc(q2(:,3),p,k,N,L);
rhsu = EulerSpecrhs1D(x,q2,L,gamma);
q = (q + 2*q2 + 2*k*rhsq)/3;
[q(:,1)] = FourierVanishHypVisc(q(:,1),p,k,N,L);
[q(:,2)] = FourierVanishHypVisc(q(:,2),p,k,N,L);
[q(:,3)] = FourierVanishHypVisc(q(:,3),p,k,N,L);
time = time+k; tstep = tstep+1;
end
return

```



**Figure 13.12.** The Euler equations solved using a Fourier collocation method with vanishing hyperviscosity ( $p = 8$ ) and  $N$  modes. The computed density (solid) is shown at  $T = 0.2$  and compared with the exact solution (dashed). In the left column, we show the results of the Fourier collocation solution, while the right column shows the results obtained after postprocessing with a Fourier-Padé approach, using  $N_c = N/4, M = L = N/8$ .

The spatial approximation of the Euler equations is performed in EulerSpecrhs1D.m.

**Script 13.19. EulerSpecrhs1D.m: Evaluation of the spatial approximation of the Euler equations using a spectral Fourier collocation scheme.**

---

```
function [dq] = EulerSpecrhs1D(x,q,L,gamma)
% function [dq] = EulerSpecrhs1D(x,q,L,gamma)
% Purpose : Evaluate the RHS of the Euler equations using spectral Fourier
% collocation method
r = q(:,1); ru = q(:,2); E = q(:,3); p = (gamma-1)*(E-0.5*ru.^2./r);

% Compute right hand side
dq(:,1) = -2*pi/L*Fourierdx(ru,1);
dq(:,2) = -2*pi/L*Fourierdx(ru.^2./r+p,1);
dq(:,3) = -2*pi/L*Fourierdx((E+p).*ru./r,1);
return
```

---

Figure 13.12 shows the computed results at  $T = 0.2$  for different resolutions. We observe excellent resolution of the shock and the rarefaction wave. Furthermore, the smearing of the contact wave is also clearly reduced when increasing the resolution. Although the Fourier–Padé post-processing visibly improves the accuracy and eliminates the Gibbs oscillations in the neighborhood of the shock, it does not reduce smearing of the contact wave, which results from accumulation of errors.

## References

- [1] Rick Archibald, Kewei Chen, Anne Gelb, and Rosemary Renaut. Improving tissue segmentation of human brain MRI through preprocessing by the Gegenbauer reconstruction method. *NeuroImage*, 20(1):489–502, 2003.
- [2] Harry Bateman and A. Erdelyi. *Higher Transcendental Functions*, volumes 1–3 of the Bateman Manuscript Project, McGraw-Hill, 1953.
- [3] John P. Boyd. *Chebyshev and Fourier Spectral Methods*. Courier Corporation, 2001.
- [4] John P. Boyd. Trouble with Gegenbauer reconstruction for defeating Gibbs’ phenomenon: Runge phenomenon in the diagonal limit of Gegenbauer polynomial approximations. *Journal of Computational Physics*, 204(1):253–264, 2005.
- [5] Claude Brezinski. *Padé-Type Approximation and General Orthogonal Polynomials*. Springer, 1980.
- [6] Claudio Canuto, M. Yousuff Hussaini, Alfio Quarteroni, and Thomas A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1988.
- [7] Claudio G. Canuto, M. Yousuff Hussaini, Alfio M. Quarteroni, and Thomas A. Zang. *Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics (Scientific Computation)*. Springer-Verlag, 2007.
- [8] Lennart Carleson. On convergence and growth of partial sums of Fourier series. *Acta Mathematica*, 116(1):135–157, 1966.

- [9] Gui Qiang Chen, Qiang Du, and Eitan Tadmor. Spectral viscosity approximations to multidimensional scalar conservation laws. *Mathematics of Computation*, 61(204):629–643, 1993.
- [10] Michel O. Deville, Paul F. Fischer, and Ernest H. Mund. *High-Order Methods for Incompressible Fluid Flow*, volume 9 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2002.
- [11] Wai Sun Don and Carl B. Quillen. Numerical simulation of shock-cylinder interactions: I. Resolution. *Journal of Computational Physics*, 122(2):244–265, 1995.
- [12] Tobin A. Driscoll and Bengt Fornberg. A Padé-based algorithm for overcoming the Gibbs phenomenon. *Numerical Algorithms*, 26(1):77–92, 2001.
- [13] Laurent Emmel, Sidi Mahmoud Kaber, and Yvon Maday. Padé–Jacobi filtering for spectral approximations of discontinuous solutions. *Numerical Algorithms*, 33(1–4):251–264, 2003.
- [14] Björn Engquist and Björn Sjögreen. The convergence rate of finite difference schemes in the presence of shocks. *SIAM Journal on Numerical Analysis*, 35(6):2464–2485, 1998.
- [15] Bengt Fornberg. On a Fourier method for the integration of hyperbolic equations. *SIAM Journal on Numerical Analysis*, 12(4):509–528, 1975.
- [16] Bengt Fornberg. *A Practical Guide to Pseudospectral Methods*, volume 1 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 1998.
- [17] Anne Gelb and David Gottlieb. The resolution of the Gibbs phenomenon for “spliced” functions in one and two dimensions. *Computers & Mathematics with Applications*, 33(11):35–58, 1997.
- [18] Anne Gelb and Zdzisław Jackiewicz. Determining analyticity for parameter optimization of the Gegenbauer reconstruction method. *SIAM Journal on Scientific Computing*, 27(3):1014–1031, 2005.
- [19] Anne Gelb and Eitan Tadmor. Detection of edges in spectral data. *Applied and Computational Harmonic Analysis*, 7(1):101–135, 1999.
- [20] Anne Gelb and Eitan Tadmor. Detection of edges in spectral data II. Nonlinear enhancement. *SIAM Journal on Numerical Analysis*, 38(4):1389–1408, 2000.
- [21] Anne Gelb and Jared Tanner. Robust reprojection methods for the resolution of the Gibbs phenomenon. *Applied and Computational Harmonic Analysis*, 20(1):3–25, 2006.
- [22] Gene H. Golub and John H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation*, 23(106):221–230, 1969.
- [23] David Gottlieb and Sigal Gottlieb. Spectral methods for compressible reactive flows. *Comptes Rendus Mécanique*, 333(1):3–16, 2005.

- [24] David Gottlieb and Steven A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*, volume 26 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1977.
- [25] David Gottlieb and Chi-Wang Shu. On the Gibbs phenomenon. IV: Recovering exponential accuracy in a subinterval from a Gegenbauer partial sum of a piecewise analytic function. *Mathematics of Computation*, 64(211):1081–1095, 1995.
- [26] David Gottlieb and Chi-Wang Shu. On the Gibbs phenomenon V: Recovering exponential accuracy from collocation point values of a piecewise analytic function. *Numerische Mathematik*, 71(4):511–526, 1995.
- [27] David Gottlieb and Chi-Wang Shu. On the Gibbs phenomenon III: Recovering exponential accuracy in a sub-interval from a spectral partial sum of a piecewise analytic function. *SIAM Journal on Numerical Analysis*, 33(1):280–290, 1996.
- [28] David Gottlieb and Chi-Wang Shu. On the Gibbs phenomenon and its resolution. *SIAM Review*, 39(4):644–668, 1997.
- [29] David Gottlieb and Chi-Wang Shu. A general theory for the resolution of the Gibbs phenomenon. *Atti Dei Convegni Lincei-Accademia Nazionale Dei Lincei*, 147:39–48, 1998.
- [30] David Gottlieb, Chi-Wang Shu, Alex Solomonoff, and Hervé Vandeven. On the Gibbs phenomenon I: Recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function. *Journal of Computational and Applied Mathematics*, 43(1-2):81–98, 1992.
- [31] Sigal Gottlieb, David Gottlieb, and Chi-Wang Shu. Recovering high-order accuracy in WENO computations of steady-state hyperbolic systems. *Journal of Scientific Computing*, 28(2-3):307–318, 2006.
- [32] Ben-yu Guo, He-ping Ma, and Eitan Tadmor. Spectral vanishing viscosity method for nonlinear conservation laws. *SIAM Journal on Numerical Analysis*, 39(4):1254–1268, 2001.
- [33] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. *Time Dependent Problems and Difference Methods*, volume 24 of Pure and Applied Mathematics. John Wiley & Sons, 1995.
- [34] Jan S. Hesthaven, Sigal Gottlieb, and David Gottlieb. *Spectral methods for time-dependent problems*, volume 21. Cambridge University Press, 2007.
- [35] Jan S. Hesthaven and Sidi M. Kaber. *Padé-Jacobi Approximants*. Technical report, HAL-01418450, 2016.
- [36] Jan S. Hesthaven, Sidi Mahmoud Kaber, and Laura Lurati. Padé-Legendre interpolants for Gibbs reconstruction. *Journal of Scientific Computing*, 28(2-3):337–359, 2006.
- [37] Jan S. Hesthaven and Robert Kirby. Filtering in Legendre spectral methods. *Mathematics of Computation*, 77(263):1425–1452, 2008.
- [38] Zdzislaw Jackiewicz. Determination of optimal parameters for the Chebyshev–Gegenbauer reconstruction method. *SIAM Journal on Scientific Computing*, 25(4):1187–1198, 2004.

- [39] Zdzislaw Jackiewicz and R. Park. A strategy for choosing Gegenbauer reconstruction parameters for numerical stability. *Applied Mathematics and Computation*, 212(2):418–434, 2009.
- [40] Sidi Mahmoud Kaber and Yvon Maday. Analysis of some Padé–Chebyshev approximants. *SIAM Journal on Numerical Analysis*, 43(1):437–454, 2005.
- [41] Sidi Mahmoud Ould Kaber. A Legendre pseudospectral viscosity method. *Journal of Computational Physics*, 128(1):165–180, 1996.
- [42] George Karniadakis and Spencer Sherwin. *Spectral/hp Element Methods for Computational Fluid Dynamics*. Oxford University Press, 2013.
- [43] Heinz-Otto Kreiss and Joseph Oliger. Stability of the Fourier method. *SIAM Journal on Numerical Analysis*, 16(3):421–433, 1979.
- [44] Laura B. Lurati. Padé–Gegenbauer suppression of Runge phenomenon in the diagonal limit of Gegenbauer approximations. *Journal of Computational Physics*, 222(1):1–8, 2007.
- [45] Yvon Maday, Sidi M. Ould Kaber, and Eitan Tadmor. Legendre pseudospectral viscosity method for nonlinear conservation laws. *SIAM Journal on Numerical Analysis*, 30(2):321–342, 1993.
- [46] Yvon Maday and Eitan Tadmor. Analysis of the spectral vanishing viscosity method for periodic conservation laws. *SIAM Journal on Numerical Analysis*, 26(4):854–870, 1989.
- [47] Andrew Majda, James McDonough, and Stanley Osher. The Fourier method for nonsmooth initial data. *Mathematics of Computation*, 32(144):1041–1081, 1978.
- [48] Mi-Sun Min, Tae-Woo Lee, Paul F. Fischer, and Stephen K. Gray. Fourier spectral simulations and Gegenbauer reconstructions for electromagnetic waves in the presence of a metal nanoparticle. *Journal of Computational Physics*, 213(2):730–747, 2006.
- [49] Misun Min, Sidi Kaber, and Wai-Sun Don. Fourier–Padé approximations and filtering for spectral simulations of an incompressible Boussinesq convection problem. *Mathematics of computation*, 76(259):1275–1290, 2007.
- [50] G. Németh and G. Páris. The Gibbs phenomenon in generalized Padé approximation. *Journal of Mathematical Physics*, 26(6):1175–1178, 1985.
- [51] Jie Shen, Tao Tang, and Li-Lian Wang. *Spectral Methods: Algorithms, Analysis and Applications*, volume 41. Springer Science+Business Media, 2011.
- [52] Bernie D. Shizgal and Jae-Hun Jung. Towards the resolution of the Gibbs phenomena. *Journal of Computational and Applied Mathematics*, 161(1):41–65, 2003.
- [53] Chi-Wang Shu and Peter S. Wong. A note on the accuracy of spectral method applied to nonlinear conservation laws. *Journal of Scientific Computing*, 10(3):357–369, 1995.
- [54] Eitan Tadmor. Skew-selfadjoint form for systems of conservation laws. *Journal of Mathematical Analysis and Applications*, 103(2):428–442, 1984.

- 
- [55] Eitan Tadmor. The exponential accuracy of Fourier and Chebyshev differencing methods. *SIAM Journal on Numerical Analysis*, 23(1):1–10, 1986.
  - [56] Eitan Tadmor. Stability analysis of finite difference, pseudospectral and Fourier-Galerkin approximations for time-dependent problems. *SIAM Review*, 29(4):525–555, 1987.
  - [57] Eitan Tadmor. Convergence of spectral methods for nonlinear conservation laws. *SIAM Journal on Numerical Analysis*, 26(1):30–44, 1989.
  - [58] Eitan Tadmor. Shock capturing by the spectral viscosity method. *Computer Methods in Applied Mechanics and Engineering*, 80(1–3):197–208, 1990.
  - [59] Eitan Tadmor. Super viscosity and spectral approximations of nonlinear conservation laws. In: *Numerical Methods for Fluid Dynamics IV*, pages 69–82, Clarendon Press, 1993.
  - [60] Eitan Tadmor. Filters, mollifiers and the computation of the Gibbs phenomenon. *Acta Numerica*, 16:305, 2007.
  - [61] Hervé Vandeven. Family of spectral filters for discontinuous problems. *Journal of Scientific Computing*, 6(2):159–192, 1991.

# Index

- $[\cdot]$ , 31, 127
- $\llbracket \cdot \rrbracket$ , 330, 403
- $\{\cdot\}$ , 127
- $\{\{\cdot\}\}$ , 330, 403
- $\langle \cdot, \cdot \rangle$ , 247
- $\Delta^+$ , 89
- $\Delta^-$ , 89
- $\delta u_j$ , 260
- $\overline{\delta u_j}$ , 263
- $\lambda$ , 76, 92
- $a \vee b$ , 78
- $a \wedge b$ , 78
- $A^\pm$ , 126, 249
- $c_{ij}^m$ , 311
- $d_r^{j_0}$ , 312
- $D^+$ , 89
- $D^-$ , 89
- $E_l$ , 89
- $h$ , 89
- $h_x$ , 148
- $h_y$ , 148
- $k$ , 89
- $L^1$ -contraction, 36, 37, 77
- $L^1$ -norm, 36
- $L^2$ -norm, 49, 383
- $L^p$ , 90
- $r_j$ , 179, 240
- $u^\pm$ , 260
- Adjoint problem, 207
- Affine mapping, 386
- Aliasing, 396, 420, 511
- Amplification matrix, 95
- Basis function, 379
  - high-order, 380
  - modal, 386
- monomial, 386
- nodal, 391
- orthonormal, 386
- Beam-Warming scheme, 241, 262
- Burgers' equation, 6, 14, 31, 33, 69, 74
  - characteristics, 7
  - two-dimensional, 149
- Cardinal function, 201
  - Shannon, 201
  - Whittaker, 201
- Cell
  - average, 123, 137, 148
  - boundary, 123
  - centered, 123
  - entropy condition, 126
  - entropy inequality, 184
- Cell entropy condition, 126
  - discontinuous Galerkin method, 398
- Central flux, 384
- Central scheme, 137, 274
  - Burgers' equation, 284
  - cell entropy condition, 280
  - Euler equations, 286
  - first order, 277
  - second order, 278
  - TVD-stability, 278
  - wave equation, 282
- Central WENO (CWENO), 345
- CFL (Courant, Friedrichs, and Lewy) condition, 99, 124, 550
- CFL number, 92
- Characteristic decomposition, 318
- Characteristic reconstruction, 318
- Characteristic waves, 54
- Characteristics, 4, 7, 29
- Collocation points, 517
- Compression, 241

- Conservation, 2  
 Conservation form, 70, 71  
 Conservation law  
     differential form, 2  
     integral form, 1, 2, 32, 123  
     linear system, 49  
     scalar, 29, 69  
     system, 49  
     two dimensions, 469  
 Conserved variable, 2  
 Consistency, 402  
 Contact discontinuity, 57  
 Convergence, 90  
 Convex flux, 30  
 CWENO (central WENO), 345  
 Differentiation matrix, 393  
 Diffusivity coefficient, 427  
 Dirac delta function  
     continuous, 33  
 Discontinuous Galerkin method, 380  
     average -  $\llbracket \cdot \rrbracket$ , 403  
     boundary conditions, 472  
     cell entropy condition, 398  
     central flux, 384  
     convergence rate, 384  
     entropy dissipation, 435  
     global conservation, 397  
     Hermite-WENO limiting, 452  
     hidden accuracy, 409, 416  
     high-order TVD-slope limiting, 443  
     jump -  $\{\cdot\}$ , 403  
     Lax-Friedrichs flux, 398  
     Lax-Wendroff theorem, 397, 424  
     local conservation, 397  
     local discontinuous Galerkin (LDG), 431  
     modal representation, 386  
     moment accuracy, 409  
     multidimensional problems, 469  
     nodal representation, 391  
     nonlinear dissipation, 425  
     nonlinear viscosity, 432  
     phase error analysis, 405  
     second order operator, 427  
     slope limiter, 441  
     storage penalty, 382  
     strong form, 381, 397, 470  
     time step, 436, 473  
     upwind flux, 384  
     weak form, 381, 397, 470  
 Discrete conservation, 71  
 Dispersion analysis  
     resolution, 408  
     upwind flux, 407  
 Dispersion relation, 406  
 Domain of dependence, 51, 99  
 Domain of influence, 52  
 E-flux, 176, 244  
     cell entropy condition, 177  
 ENO (essentially nonoscillatory) scheme,  
     307, 309, 314  
     accuracy, 319, 325  
     entropy flux, 321  
     entropy stability, 321  
     sign property, 320  
 Entropy  
     cell condition, 80, 126, 184  
     condition, 34, 35, 42, 59, 74  
     conservative, 187, 195  
     fix, 131  
     flux, 40, 80  
     functions, 40, 59  
     inequality, 184  
     Kružkov pair, 45  
     Lax condition, 35  
     Lax condition system, 60  
     Oleinik condition, 36, 174  
     pair, 40, 61, 62  
     production, 185  
     stable, 184  
 Equations of elastodynamics, 11  
 Essentially nonoscillatory (ENO) scheme,  
     307, 309, 314  
 Euler equations, 12, 13  
     characteristics, 52  
     entropy pair, 63  
     entropy variables, 63  
     one-dimensional, 16, 52, 63  
     shock-entropy problem, 17  
     Sod's problem, 17  
     two-dimensional, 18, 153  
     vortex solution, 19, 159  
 Explicit SSP multistep methods (SSPEMS),  
     234  
 Explicit SSP Runge–Kutta (SSPERK), 225, 226  
 Exponential accuracy, 170  
 Extrema preserving limiters, 453  
 FCT (flux correction transport), 239

- Filter, 209, 530  
accuracy, 216  
coefficients, 209  
exponential, 211  
function, 209, 421  
matrix, 421  
optimal, 211  
order, 209  
stability, 215
- Finite difference  
boundary conditions, 104  
consistent, 94  
convergence, 93  
Lax–Friedrichs, 90, 92, 99  
Lax–Wendroff, 95, 99, 102  
local Lax–Friedrichs, 102  
Rusanov, 102  
stable, 94
- Finite difference method, 89, 166, 237
- Finite difference scheme  
central, 166  
stencil, 166
- Finite volume  
boundary conditions, 104
- Finite volume method, 124, 125, 237  
two-dimensional, 147, 148
- Flux, 2, 123  
convex, 30  
Jacobian, 52  
numerical, 71
- Flux correction transport (FCT), 239
- Flux limited scheme, 238  
two-dimensional, 288
- Flux limiter, 238, 251  
Chakravarthy–Osher, 242  
Koren, 243  
minmod, 242  
OSPRE, 243  
Superbee, 243  
Sweby, 243  
symmetric, 243  
van Leer, 243
- Flux reconstruction scheme, 464, 465
- Flux splitting  
Lax–Friedrichs, 317  
Roe, 317
- Forward Euler method, 220
- Fourier basis, 505
- Fourier–Galerkin  
stability, 516
- Fourier method  
CFL condition, 550  
collocation stability, 519  
Galerkin stability, 516
- Fourier series, 504  
accuracy, 506, 512  
aliasing error, 511  
continuous, 504  
derivative, 507  
differentiation matrix, 511  
discrete, 97, 509  
Parseval’s identify, 506
- Fourier spectral method, 503
- Fromm scheme, 262
- Frozen problem, 50
- Galerkin scheme, 207, 379
- Gaussian quadrature, 387  
Gauss–Lobatto quadrature, 392
- Gegenbauer  
Padé reconstruction, 548  
polynomial, 543  
quadrature, 548
- Generalized Riemann problem, 261
- Genuinely nonlinear, 57
- Gibbs  
complementary basis, 541  
condition, 542  
phenomenon, 201, 204, 409
- Godunov’s method, 123, 125
- Godunov’s theorem, 82
- Grid, 69, 90  
function, 89  
operator, 89  
stencil, 166, 205
- Grid size  
spatial  $h$ , 69  
temporal  $k$ , 69
- Heat equation, 427
- Heaviside function, 308
- Homogeneous function, 522
- Hyperbolic system  
genuinely nonlinear, 57  
linear, 51  
quasilinear form, 52  
strictly, 51  
strongly, 51  
symmetric, 51

- weakly, 51  
wellposed, 50
- Ideal gas law, 16
- Implicit SSP Runge–Kutta (SSPIRK), 227, 228
- Inverse Lax–Wendroff, 364
- Kernel, 202  
  Dirichlet, 202
- KPP (Kolmogorov–Petrovskii–Piskunov) problem  
  monotone scheme, 151
- Kreiss matrix theorem, 95
- Kronecker delta, 12
- Lagrange interpolation, 201
- Lagrange polynomial, 166, 204, 311, 392  
  remainder, 166
- Lagrangian approach, 266
- Lax–Friedrichs  
  flux, 398, 415  
  flux splitting, 317  
  method, 82, 178  
  scheme, 200
- Lax–Friedrichs scheme  
  central, 138
- Lax–Richtmyer equivalence theorem, 94
- Lax–Wendroff  
  method, 82, 178, 200, 262
- Lax–Wendroff theorem, 72  
  discontinuous Galerkin method, 397
- Legendre polynomial, 387  
  derivative, 388  
  recurrence, 387
- Legendre quadrature  
  Gauss, 323  
  Gauss–Radau, 463
- Linear scheme, 81
- Linear system, 49  
  hyperbolic, 51  
  Riemann problem, 54  
  strictly hyperbolic, 51  
  strongly hyperbolic, 51  
  symmetric hyperbolic, 51  
  weakly hyperbolic, 51
- Mass matrix  
  global, 379  
  local, 382, 387, 393, 396
- Maximum principle, 77
- Maxmod function, 444
- Maxwell’s equations, 10, 15, 56, 60
- Lax–Friedrichs flux, 330
- one-dimensional, 127
- upwind flux, 329
- Method of lines, 219
- Minmod function, 239  
  TVB, 265
- Modal representation, 386
- Modified equation, 92, 167  
  central scheme, 169  
  downwind scheme, 169  
  upwind scheme, 168
- Monotone scheme, 76, 77  
  two-dimensional, 149
- Monotone upstream-centered scheme for conservation laws (MUSCL), 260
- Monotonicity, 74
- Monotonicity-preserving scheme, 81
- MUSCL (monotone upstream-centered scheme for conservation laws), 260
- Navier–Cauchy equation, 12
- Newton divided differences, 315
- Newton form, 315
- Nodal representation, 391
- Nonconservation form, 70
- Nonconvex problem, 18, 151
- Nonlinear diffusivity, 432  
  coefficient decay based, 434  
  entropy-based, 435
- Nonlinear schemes, 165
- Norm  
   $L^1$ , 36  
   $L^2$ , 49, 383  
   $L^p$ , 90  
  broken, 383  
  continuous, 90  
  discrete, 90  
  matrix, 51, 94  
  seminorm, 383  
  Sobolev, 383
- Normal matrix, 96
- Numerical flux, 71, 127  
  alternating local discontinuous Galerkin, 431  
  central, 428  
  consistent, 71  
  E-flux, 176
- Engquist–Osher, 134, 190
- Godunov, 126, 190

- heat equation, 428  
HLL (Harten–Lax–van Leer), 135, 140  
HLLC, 136, 141  
Lax–Friedrichs, 74, 78, 102, 127, 130  
Lax–Wendroff, 74, 79, 102, 191  
Lipschitz continuous, 71  
local Lax–Friedrichs, 102, 190, 470  
monotone, 76  
multidimensional, 148  
Roe, 75, 102, 125, 129, 139  
Rusanov, 74, 102  
Numerical phase speed, 171  
Numerical phase velocity, 406
- Padé approximation  
Fourier, 536  
Gibbs phenomenon, 536  
singular, 538  
Parseval’s identity, 98  
Petrov–Galerkin scheme, 381, 458  
Phase error, 171, 408  
Phase error analysis, 170, 405  
Picard’s  
theorem, 221  
formula, 219  
Piecewise linear method (PLM), 266  
Piecewise parabolic method (PPM), 267  
PLM (piecewise linear method), 266  
Points per wavelength, 171, 172, 406  
Positive schemes, 248  
PPM (piecewise parabolic method), 267  
Primitive, 310
- Quasilinear form, 52
- Rankine–Hugoniot condition, 31, 55, 57  
Rarefaction wave, 34  
Residual, 378  
Riemann invariants, 58  
Riemann problem, 53, 54  
approximate, 128  
small data, 58  
Roe  
average, 136  
condition, 129  
flux, 75, 125  
flux splitting, 317  
linearization, 129  
matrix, 129  
numerical flux, 129, 131
- Runge–Kutta method, 221, 222  
diagonally implicit, 223  
explicit, 223  
implicit, 223  
order conditions, 228  
reducible, 223
- Scalar conservation law, 29  
Semibounded operator, 516  
Shape function  
linear, 379  
Shift operator  $E_l$ , 71  
Shock, 60  
Shock tube problem  
one-dimensional, 17  
two-dimensional, 19, 159  
Shock wave, 35  
Shock-entropy problem, 17  
sinc-function, 201  
Single-valued flux, 397  
Singular Fourier–Padé form, 538  
Skew-symmetric form, 520, 522  
Slope limited scheme  
two-dimensional, 288  
Slope limiter, 264  
discontinuous Galerkin method, 441  
minmod, 264, 443  
moment limiting, 446  
MUSCL, 264, 443  
parameter free, 444  
Superbee, 264  
van Albada, 264  
WENO, 449
- Smoothness estimator, 435  
Smoothness indicator, 432  
Sobolev norm, 504  
Sod’s problem, 17  
Software libraries, 13  
Sonic point, 102, 130  
fix, 102, 126
- Spectral collocation penalty method, 459  
Spectral difference scheme, 461, 468  
Spectral finite volume scheme, 461  
Spectral Galerkin penalty method, 460  
Spectral method  
collocation penalty, 459  
difference scheme, 461  
finite volume, 461  
Galerkin penalty, 460  
staggered grid, 462

- Speed of sound, 17  
 SSP (strong stability preserving) effectivity index, 232  
 SSP scheme, 219, 224  
   effective SSP coefficient, 226  
   explicit SSPERK, 223  
   implicit SSPIRK, 227  
   modified Shu–Osher form, 230  
   multistep method, 232  
   negative coefficients, 232  
   order barriers, 228  
   Shu–Osher form, 223  
 SSP coefficient, 224  
 SSPEMS(s,p) (explicit SSP multistep), 234  
 SSPERK(10,4) (explicit SSP Runge–Kutta), 226  
 SSPERK(2,2), 225  
 SSPERK(3,3), 226  
 SSPERK(s,p), 225  
 SSPIRK(s,2) (implicit SSP Runge–Kutta), 227  
 SSPIRK(s,3), 228  
 SSPIRK(s,p), 227  
 stage order, 229  
 Stability, 402  
   von Neumann, 96  
 Staggered scheme, 277  
 Stencil, 71, 166  
 Stiffness matrix  
   global, 379  
   local, 382, 388, 393, 396, 428  
 Sturm–Liouville problem, 389  
 Symbol of scheme, 98  
 Symmetrizer, 62, 522
- Test function, 379  
 Test problem  
   Burgers' equation, 14  
   linear wave equation, 14  
   Maxwell's equations, 15  
   nonconvex problem, 18  
   one-dimensional Euler equations, 16  
   two-dimensional Euler equations, 18  
 Total variation (TV), 36  
 bounded (TVB), 175, 181  
 continuous, 174  
 diminishing (TVD), 36, 77, 174, 175, 180  
 discrete, 174  
 stable, 174  
 Trace inequality, 404  
 Trapezoidal method, 220  
 Trial function, 379  
   high-order, 380  
 Troubled-cell indicator, 443, 449  
 Truncation error, 93, 402  
 TV (total variation)-stable, 174  
 TVB (total variation bounded)-stability, 265  
 TVD (total variation diminishing)-region, 240  
 Upwind flux, 384  
   Maxwell's equations, 329  
 Vandermonde matrix, 391  
 Vanishing viscosity solution, 32, 38  
 Viscosity coefficient, 427  
 von Neumann stability, 96
- Wave  
   characteristic, 54  
   contact, 57  
   equation, 3, 14  
   N-wave, 39  
   rarefaction, 34  
   shock, 35  
 Weak solution, 32, 33  
 Well balanced scheme, 347  
 Wellposed problem, 49  
 WENO (weighted essentially nonoscillatory) method, 307  
   accuracy, 353  
   central (CWENO), 345  
   critical point, 343  
   embedded, 345  
   inverse Lax–Wendroff, 364  
   mapped weights, 343  
   smoothness indicator, 338, 339  
   stability, 353  
   WENO-Z, 344, 355

Conservation laws are the mathematical expression of the principles of conservation and provide effective and accurate predictive models of our physical world. Although intense research activity during the last decades has led to substantial advances in the development of powerful computational methods for conservation laws, their solution remains a challenge and many questions are left open; thus it is an active and fruitful area of research.

*Numerical Methods for Conservation Laws: From Analysis to Algorithms*

- offers the first comprehensive introduction to modern computational methods and their analysis for hyperbolic conservation laws, building on intense research activities for more than four decades of development;
- discusses classic results on monotone and finite difference/finite volume schemes, but emphasizes the successful development of high-order accurate methods for hyperbolic conservation laws;
- addresses modern concepts of TVD and entropy stability, strongly stable Runge–Kutta schemes, and limiter-based methods before discussing essentially nonoscillatory schemes, discontinuous Galerkin methods, and spectral methods;
- explores algorithmic aspects of these methods, emphasizing one- and two-dimensional problems and the development and analysis of an extensive range of methods;
- includes MATLAB software with which all main methods and computational results in the book can be reproduced; and
- demonstrates the performance of many methods on a set of benchmark problems to allow direct comparisons.

Code and other supplemental material are available online at [www.siam.org/books/cs18](http://www.siam.org/books/cs18).

This book is intended for graduate students in computational mathematics and researchers seeking a comprehensive introduction to modern methods for solving conservation laws. Students and researchers in applied sciences and engineering will benefit from the book's emphasis on algorithmic aspects of complex algorithms. The text also includes extensive references which allows researchers to pursue advanced research and results.



**Jan S. Hesthaven** is Dean of Basic Sciences, Professor of Mathematics, and holds the Chair of Computational Mathematics and Simulation Science at Ecole Polytechnique Fédérale de Lausanne (EPFL) in Switzerland. Prior to joining EPFL in 2013, he was Professor of Applied Mathematics at Brown University. He has worked for more than two decades on the development, analysis, and application of modern computational methods for linear and nonlinear wave problems, with an emphasis on high-order accurate methods. He is an Alfred P. Sloan Fellow (2001), an NSF Career award winner (2002), and a SIAM Fellow (2014).

**siam**

Society for Industrial and Applied Mathematics

3600 Market Street, 6th Floor

Philadelphia, PA 19104-2688 USA

+1-215-382-9800 • Fax: +1-215-386-7999

[siam@siam.org](mailto:siam@siam.org) • [www.siam.org](http://www.siam.org)

ISBN 978-1-611975-09-3



9781611975093

**MATLAB®**  
examples



**CS18**