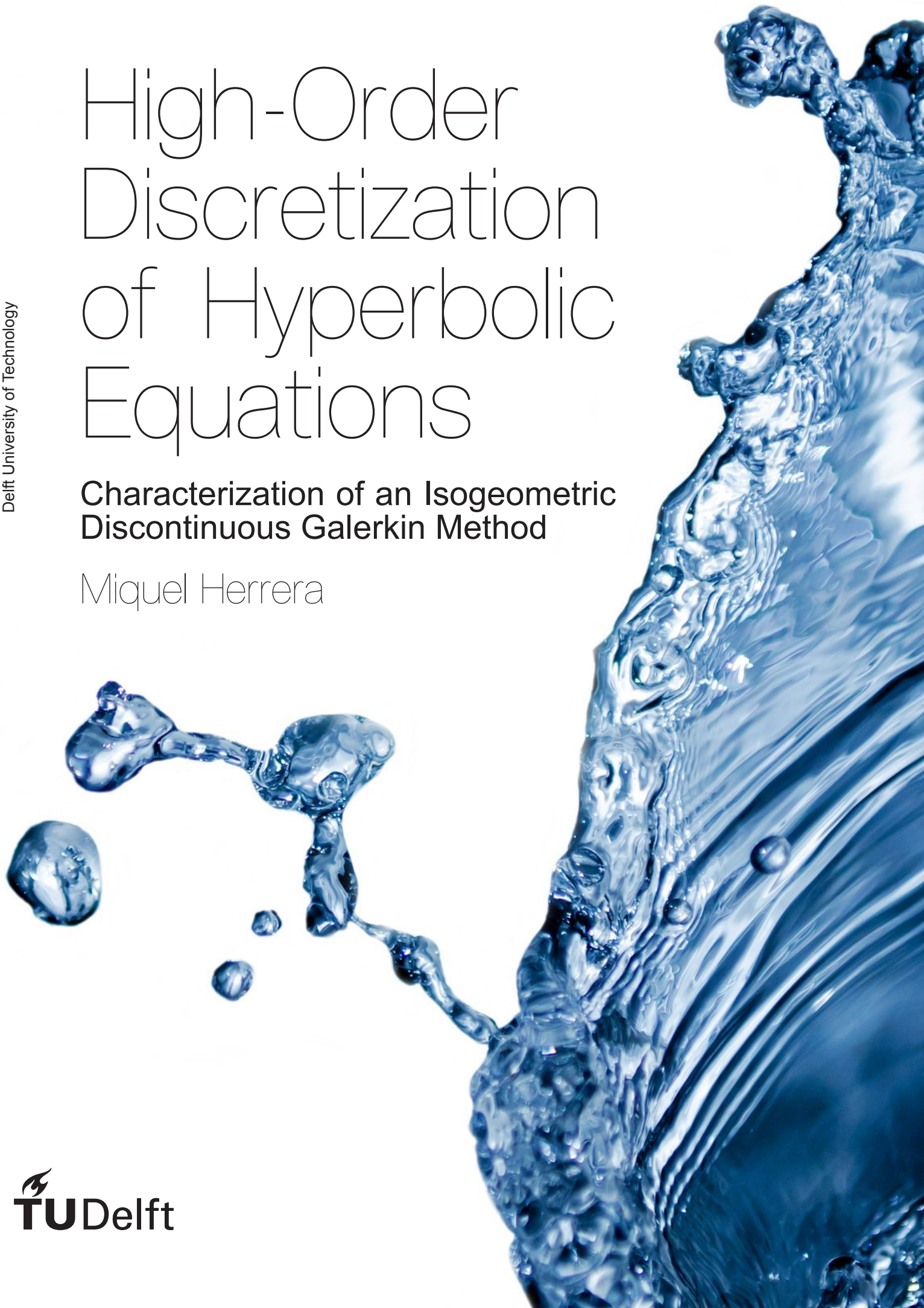
High-Order Discretization of Hyperbolic Equations

Characterization of an Isogeometric Discontinuous Galerkin Method

Miquel Herrera

Delft University of Technology



TU Delft

High-Order Discretization of Hyperbolic Equations

Characterization of an Isogeometric Discontinuous Galerkin Method

by

Miquel Herrera

to obtain the degree of Master of Science in Aerospace Engineering
at the Delft University of Technology,
to be defended publicly on Thursday, 3 June 2021 at 10:00 a.m.

Student number:	4719441	
Project duration:	4 March 2019 – 10 May 2021	
Thesis committee:	Prof. dr. ir. S. Hickel,	TU Delft, LR (supervisor)
	Dr. M. Möller,	TU Delft, EEMCS (supervisor)
	Dr. ir. M. I. Gerritsma,	TU Delft, LR
	Dr. ir. D. Toshniwal,	TU Delft, EEMCS

LR Faculty of Aerospace Engineering
EEMCS Faculty of Electrical Engineering, Mathematics and Computer Science

TU Delft Delft University of Technology

Abstract

Computational fluid dynamics is nowadays one of the pillars of modern aircraft design, just as important as experimental wind tunnel testing. Very ambitious goals in regards to performance, efficiency and sustainability are being asked of the aviation industry, the kind that warrant a virtual exploration of the edges of the flight envelope. High-order has come to be regarded as a necessary ingredient to achieve breakthrough advances in this direction. So much so, in fact, that the major aircraft manufacturers, governmental aerospace research agencies and top universities worldwide have been acting in coordination to increase the technology readiness level of these approaches, for the last ten years.

In this work, I study in significant detail the one aspect that makes high-order methods ideal candidates for enabling the use of more advanced turbulence models in industry: the cost-efficiency of their discretization—they offer minimal amounts of dispersion and dissipation errors, for a given number of degrees of freedom. I consider three research objects:

- The discontinuous Galerkin spectral method (DGSEM)
- All energy-stable variants of flux reconstruction (FR), also known as correction procedure via reconstruction (CPR), that can be parameterized through a single scalar parameter
- A novel B-spline–based discontinuous Galerkin formulation, stabilized via algebraic flux correction (DGIGA-AFC)

and characterize their order of accuracy, linear and nonlinear stability characteristics, as well as dispersion and dissipation errors as a function of wavenumber. Afterwards, I experimentally investigate their relative suitability towards scale-resolving simulation of compressible and turbulent flows, by solving a number of simple test cases of increasing difficulty (linear advection, inviscid Burgers and Euler equations; all in 1D) using a purpose-made MATLAB implementation.

The proposed isogeometric method (DGIGA) has been found to be at least as viable as the other two, a priori, for the resolution of high-speed turbulent flows. Moreover, I have found that low dispersion and dissipation need not always be associated to high order, but to a high number of degrees of freedom per patch instead; these two coincide in the more conventional schemes, yet not necessarily in DGIGA. As a nonlinear stabilization mechanism, however, the proposed combination of DGIGA with AFC has turned out to be inferior to existing limiters.

Preface

This document summarizes all the knowledge and insight I have obtained after understanding, implementing, and experimenting with discontinuous Galerkin methods for over two years. The entire project has been the consequence of the following simple idea: *to understand a numerical method, you need to implement it yourself*.

I have had a hard time bringing this project to a reasonably satisfying conclusion, as the time it has taken me might suggest. Perhaps the topic was too ambitious and/or ambiguous from the start. That certainly didn't help. Most of all, I think, this was an instance of me falling prey to my own lack of control: I should simply have gotten more to the point. I realize now, in hindsight, that the prolonged lack of bandwidth when communicating with my supervisors and peers due to remote working conditions forced upon all of us due to the COVID-19 health crisis, has probably contributed to this tendency of mine to "do my own thing", losing track of time and priorities. I am sure that I could have done better. All in all, the completion of this project has been a bit less enjoyable that I had hoped. Still, overall, I am quite content with how my adventure in The Netherlands has turned out to be.

I would like to end by first thanking my supervisors, Stefan and Matthias, for their effort in helping me see my own work in a more positive light; worse people would simply not have cared. Thank you, also (even if that led me slightly astray) for giving me the freedom to shape this project as I saw fit, in every step of the way. Then, finally, I wish to say that I am grateful to my parents for their unconditional support. I acknowledge that, in addition to not seeing me in person for almost a year-and-a-half now, they have had to live with the angst of not being able to help—in fact, of barely being able to even understand what I was struggling with—for such a long time.

Miquel Herrera
The Hague, May 2021

Contents

List of Figures xiii

List of Tables xvii

Nomenclature xix
Symbols . . . xix
Abbreviations . . . xx

1 Introduction 1
1.1 Motivation: next-generation CFD . . . 2
1.2 Research objectives . . . 4
1.3 Relevance of this work . . . 5
1.4 Outline . . . 5

I Theory 7

2 Hyperbolic Conservation Laws 9
2.1 Conservation laws in one dimension . . . 9
2.1.1 Quasilinear form . . . 10
2.1.2 Integral form . . . 10
2.2 Hyperbolicity . . . 10
2.2.1 Characteristic fields . . . 11
2.3 Discontinuous solutions . . . 12
2.4 Riemann problem . . . 12
2.5 Advection . . . 13
2.5.1 Eigendecomposition . . . 13
2.5.2 Riemann flux . . . 14
2.6 Wave equation . . . 14
2.6.1 Eigendecomposition . . . 14
2.6.2 Riemann flux . . . 14
2.6.3 Equivalent systems . . . 14
2.7 Inviscid Burgers equation . . . 15
2.7.1 Eigendecomposition . . . 15
2.7.2 Riemann flux . . . 15
2.8 Euler equations . . . 16
2.8.1 Eigendecomposition . . . 17
2.8.2 Riemann solvers . . . 18

3 Compact Discontinuous High-Order Discretization 21
3.1 Domain discretization . . . 21
3.1.1 Reference element . . . 22
3.2 Solution discretization . . . 22
3.2.1 Reference element space . . . 23
3.2.2 Trial basis functions . . . 23
3.2.3 Degrees of freedom . . . 23
3.2.4 Flux representation . . . 23
3.3 Equation discretization . . . 24
3.3.1 Spatial residuals (method of lines) . . . 24
3.3.2 Weak formulation . . . 24
3.3.3 Inter-element coupling . . . 25
3.3.4 Semi-discrete hyperbolic conservation law . . . 26

3.4 The low-order case: finite volume discretization . . . 26
3.5 Initial condition projection . . . 27
3.6 Numerical boundary conditions . . . 28
3.6.1 Periodic . . . 28
3.6.2 Farfield . . . 28
3.6.3 Transmissive . . . 29
3.6.4 Reflective . . . 29

4 Discontinuous Galerkin Spectral Element Method (DGSEM) 31
4.1 Spectral basis functions . . . 31
4.1.1 Legendre polynomials . . . 32
4.1.2 Lagrange polynomials . . . 33
4.2 DGSEM semi-discretization . . . 33
4.2.1 Modal representation . . . 33
4.2.2 Nodal representation . . . 35
4.2.3 Collocated quadrature . . . 36
4.2.4 Semi-discrete DGSEM operators . . . 37
4.2.5 Polynomial aliasing . . . 38

5 Flux Reconstruction (FR) or Correction Procedure via Reconstruction (CPR) 41
5.1 FR/CPR semi-discretization . . . 41
5.1.1 Differential formulation . . . 42
5.1.2 Flux correction . . . 42
5.1.3 Flux derivative . . . 43
5.1.4 Semi-discrete FR/CPR operators . . . 44
5.2 Correction functions . . . 45
5.2.1 DG correction function . . . 45
5.2.2 SG/SD correction function . . . 45
5.2.3 Huynh's correction functions . . . 46
5.2.4 Energy-stable correction functions . . . 47
5.3 FR/CPR and the discontinuous Galerkin method . . . 48
5.3.1 Initialization . . . 48
5.3.2 FR/CPR and the finite volume method . . . 50

6 Isogeometric Analysis (IGA) 53
6.1 Basis splines (B-splines) . . . 53
6.1.1 Knot vector . . . 53
6.1.2 Basis functions . . . 53
6.1.3 B-spline curves . . . 54
6.2 Related bases . . . 55
6.2.1 Non-uniform rational B-splines (NURBS) . . . 55
6.2.2 Bernstein polynomials and Bézier curves . . . 56
6.2.3 Classical FEA vs. IGA . . . 56
6.3 Discontinous Galerkin isogeometric analysis (DGIGA) . . . 57
6.3.1 Flux expansion coefficients . . . 58
6.3.2 Semi-discrete DGIGA operators . . . 60
6.4 Algebraic flux correction (AFC) . . . 66
6.4.1 A predictor-corrector approach to high-resolution . . . 66
6.4.2 Mass matrix lumping . . . 67
6.4.3 Artificial viscosities . . . 68

7 Time Discretization 69
7.1 The method of lines . . . 69
7.1.1 Courant number . . . 70
7.1.2 Amplification factor . . . 70

7.2 Strong stability preserving (SSP) time discretization . . . 71
7.3 Explicit SSP Runge-Kutta methods (SSP-RK) . . . 72
7.3.1 SSP-RK1(1) or Euler's method: 1st order, 1 stage . . . 72
7.3.2 SSP-RK2(2): 2nd order, 2 stages . . . 73
7.3.3 SSP-RK3(3): 3rd order, 3 stages . . . 73
7.3.4 SSP-RK4(5): 4th order, 5 stages . . . 73
7.3.5 SSP-RK4(10): 4th order, 10 stages . . . 74
7.4 Alternative time discretization schemes . . . 74

8 Nonlinear Stabilization **77**
8.1 Total variation stability . . . 77
8.1.1 Total variation diminishing (TVD) . . . 79
8.1.2 Monotonicity . . . 80
8.1.3 Total variation bounded (TVB) . . . 80
8.1.4 Other nonlinear stability criteria . . . 80
8.2 Legendre-based limiting . . . 81
8.3 Discontinuity sensing . . . 81
8.3.1 KXRCF sensor . . . 81
8.3.2 AP-TVD sensor . . . 82
8.4 Generalized slope limiting . . . 83
8.4.1 Modified minmod function . . . 83
8.4.2 Local characteristic variables . . . 84
8.4.3 TVB limiter . . . 84
8.5 Generalized moment limiting . . . 84
8.5.1 BDF limiter . . . 85
8.5.2 BSB limiter . . . 85
8.5.3 Krivodonova's limiter . . . 86
8.6 Weighted essentially non-oscillatory (WENO) limiting . . . 87
8.6.1 HWENO limiter . . . 87
8.7 Flux corrected transport (FCT) limiting . . . 88
8.7.1 Linearized antidiffusive fluxes . . . 89
8.7.2 Synchronized sequential FCT limiter . . . 89
8.7.3 Constrained initialization . . . 91
8.7.4 Troubled element detection . . . 93
8.8 Fail-safe limiting for the Euler equations . . . 93
8.8.1 Fail-safe slope limiter . . . 94
8.8.2 Last-resort fail-safe limiter . . . 94
8.8.3 Invalid element criteria for inter-cell fail-safe limiters . . . 94
8.8.4 Sub-cell FCT fail-safe limiter . . . 95

II Experiments **97**

9 Methodology **99**
9.1 MATLAB implementation . . . 99
9.2 Test matrices . . . 99
9.3 List of model problems . . . 100
9.3.1 Monochromatic wave (linear) . . . 100
9.3.2 Monochromatic wave (nonlinear) . . . 100
9.3.3 Monochromatic wave (Euler) . . . 101
9.3.4 Gaussian hump (linear) . . . 101
9.3.5 Gaussian hump (nonlinear) . . . 101
9.3.6 Triangular pulse (linear) . . . 102
9.3.7 Triangular pulse (nonlinear) . . . 102
9.3.8 Jiang-Shu problem . . . 102
9.3.9 Toro's transonic shock tube . . . 103
9.3.10 The 1-2-3 problem . . . 103
9.3.11 Blast wave interaction . . . 103

9.3.12 Acoustic wave–shock wave interaction . . . 104

10 Order of Accuracy **105**
10.1 Time schemes . . . 105
10.2 DGSEM . . . 106
10.3 FR/CPR . . . 106
10.4 DGIGA . . . 107

11 Dispersion, Dissipation and Linear Stability **133**
11.1 DGSEM . . . 133
11.2 FR/CPR . . . 133
11.3 DGIGA . . . 134

12 Optimal FR/CPR and DGIGA Configurations **175**
12.1 Identification of optimal configurations . . . 175
12.1.1 Objective function . . . 175
12.1.2 Optimization problems . . . 176
12.1.3 Optimization results and discussion . . . 177
12.2 Combined-mode dispersion and dissipation errors . . . 177
12.3 Balance between dispersion and dissipation . . . 180
12.4 Relative cost at a fixed resolution . . . 182
12.4.1 Number of time-steps . . . 183
12.4.2 Number of elements or patches . . . 183
12.4.3 Cost model . . . 184
12.4.4 Results and discussion . . . 184

13 Nonlinear Physics **187**
13.1 Burgers equation . . . 187
13.1.1 Verification of combined-mode analysis results . . . 187
13.1.2 Influence of the initial condition . . . 188
13.1.3 Linear vs. nonlinear advection . . . 189
13.2 Euler equations . . . 190
13.2.1 Modified wavenumber analysis a la Hickel et al. . . . 192
13.2.2 A posteriori vs. combined-mode analyses . . . 199
13.2.3 Spatial schemes . . . 199
13.2.4 Riemann solvers . . . 199

14 Non-smooth Solutions, Sensors and Limiters **203**
14.1 Hierarchical limiters . . . 204
14.2 Sensors . . . 205
14.3 DGIGA-AFC . . . 206
14.4 IGA-AFC . . . 207
14.5 Final comparison . . . 208
14.6 DGIGA-AFC revisited . . . 210
14.7 Do limiters preserve high order? . . . 210

15 Conclusions **231**
15.1 Order of accuracy . . . 231
15.1.1 Limitations . . . 232
15.1.2 Recommendations . . . 232
15.2 Dispersion, dissipation and linear stability . . . 232
15.2.1 Limitations and recommendations . . . 233
15.3 Optimal FR/CPR and DGIGA configurations . . . 233
15.3.1 Limitations . . . 233
15.3.2 Recommendations . . . 234
15.4 Nonlinear physics . . . 234
15.4.1 Limitations and Recommendations . . . 234
15.5 Discontinuous solutions, limiters and sensors . . . 234
15.5.1 Limitations and recommendations . . . 235

15.6 Is DGIGA-AFC well-suited to LES of high-speed flows? . . . 235
15.7 Recommendations for future work . . . 235

A Modified Wavenumber Analysis **237**
A.1 Discrete wavenumbers . . . 237
A.2 Wave propagation . . . 238
A.2.1 Exact dispersion relation . . . 240
A.2.2 Modified wavenumber . . . 241
A.2.3 Multiplicity of eigenmodes . . . 242
A.2.4 Dominant eigenmode . . . 243
A.2.5 Combined mode semi-discrete analysis . . . 243
A.3 Residual operators in matrix form . . . 245
A.3.1 DGSEM . . . 245
A.3.2 FR/CPR . . . 246
A.3.3 DGIGA . . . 247
A.4 Linear stability . . . 247
A.4.1 Amplification factor . . . 247
A.4.2 Fourier footprint . . . 248
A.4.3 Time-step size limits . . . 248
A.5 Quantifying dispersion and dissipation . . . 249
A.5.1 Order of accuracy . . . 249
A.5.2 Resolving efficiency . . . 250
A.5.3 Numerical cutoff wavenumber . . . 250
A.5.4 Dispersion to dissipation ratio . . . 250

B Time Complexity Estimation **253**
B.1 Discontinuous coupling . . . 254
B.2 Residual evaluation (spatial schemes) . . . 256
B.2.1 DGSEM . . . 256
B.2.2 FR/CPR . . . 256
B.2.3 DGIGA . . . 257
B.3 Solution update (time scheme) . . . 258

Bibliography **259**

List of Figures

1.1 Order of accuracy . . . 1
1.2 Roofline model . . . 3

2.1 The Riemann problem . . . 13

3.1 Schematic representation of a one-dimensional mesh . . . 22
3.2 Indexing convention for all meshes in this report . . . 22
3.3 Mapping to/from reference element . . . 22
3.4 Periodic boundary condition . . . 28
3.5 Reflective vs. transmissive boundary conditions . . . 30

4.1 Examples of polynomial bases . . . 34
4.2 Robustness indicators for various DGSEM nodal distributions. . . . 37
4.3 Quadrature point distributions (Gauss-Lobatto vs. Gauss-Legendre) . . . 38
4.4 Polynomial aliasing in DGSEM . . . 39

5.1 The flux correction procedure in FR/CPR . . . 43
5.2 Classical FR/CPR correction functions . . . 47
5.3 Examples of VCJH-type correction functions . . . 49
5.4 Initial condition projection in FR/CPR . . . 50

6.1 Truncated triangular table representation of a B-spline basis . . . 55
6.2 Examples of B-spline bases . . . 56
6.3 B-spline curves and control polygons . . . 57
6.4 Variation diminishing Bézier interpolants . . . 58
6.5 Basis functions: CG vs. IGA . . . 59
6.6 IGA flux expansion coefficients: modal vs. nodal (solution comparison) . . . 61
6.7 DGIGA flux expansion coefficients: modal vs. nodal (solution comparison) . . . 62
6.8 IGA flux expansion coefficients: modal vs. nodal (norm comparison) . . . 63
6.9 DGIGA flux expansion coefficients: modal vs. nodal (norm comparison) . . . 64
6.10 Condition number of B-spline mass matrices . . . 66
6.11 Reciprocal condition number of B-spline patch Vandermonde matrices . . . 67
6.12 Resolution of a shock in DGIGA . . . 68

7.1 Indexing convention for time discretizations . . . 70
7.2 Amplification factor contours of the backward Euler method . . . 71
7.3 Domains of stability of the optimal SSP Runge-Kutta schemes . . . 75

8.1 Overview of shock capturing approaches for finite element methods . . . 78
8.2 Krivodonova's recommended limiting intensity range . . . 87
8.3 Inter-patch coupling in FCT . . . 91
8.4 FCT limiter for DGIGA . . . 92
8.5 Effect of fail-safe limiting . . . 96

10.1 Order of accuracy in time . . . 110
10.2 DGSEM, time-step size independence . . . 110
10.3 DGSEM, Δx refinement . . . 111
10.4 DGSEM, p refinement . . . 111
10.5 FR/CPR, time-step size independence . . . 112
10.6 FR/CPR, Δx refinement . . . 112

10.7 FR/CPR, p refinement . . . 113
10.8 DGIGA's four refinement directions, time-step size independence . . . 113
10.9 DGIGA, $\mathrm{K} \gg 1$, time-step size independence . . . 114
10.10 DGIGA, $\mathrm{k} \gg 1$, time-step size independence . . . 114
10.11 DGIGA, $\mathrm{p} \gg 1$, time-step size independence . . . 115
10.12 DGIGA's four refinement directions . . . 115
10.13 DGIGA, $\mathrm{k} = 1$, Δx refinement . . . 116
10.14 DGIGA, $\mathrm{p} = 2$, Δx refinement . . . 116
10.15 DGIGA, $\mathrm{p} = 3$, Δx refinement . . . 117
10.16 DGIGA, $\mathrm{p} = 4$, Δx refinement . . . 117
10.17 CG, k refinement . . . 118
10.18 IGA, k refinement . . . 118
10.19 DGIGA, $\mathrm{p} = 2$, k refinement . . . 119
10.20 DGIGA, $\mathrm{p} = 3$, k refinement . . . 119
10.21 DGIGA, $\mathrm{p} = 4$, k refinement . . . 120
10.22 DGIGA, p refinement . . . 120
10.23 DGIGA, combined p and $\varkappa$ refinement . . . 121
10.24 DGSEM, selected Δt runs . . . 122
10.25 DGSEM, selected Δx runs . . . 123
10.26 DGSEM, selected p runs . . . 124
10.27 FR/CPR, selected η runs . . . 125
10.28 FR/CPR, selected p runs . . . 126
10.29 DGIGA, selected runs at $\mathrm{N}_{\mathrm{dofs}} = 60$ . . . 127
10.30 DGIGA, selected p and $\varkappa$ runs at $\mathrm{N}_{\mathrm{dofs}} = 100$ and $\mathrm{N}_{\mathrm{dofs}} = 120$ . . . 128
10.31 DGIGA, selected Δx runs . . . 129
10.32 DGIGA, selected k runs . . . 130
10.33 DGIGA, selected p runs . . . 131
10.34 DGIGA, selected $\varkappa$ runs . . . 132

11.1 All eigenmodes; DGSEM, $\mathrm{p} = 2$ . . . 135
11.2 All eigenmodes; DGSEM, $\mathrm{p} = 3$ . . . 135
11.3 All eigenmodes; DGSEM, $\mathrm{p} = 4$ . . . 136
11.4 Physical eigenmode; DGSEM, $0 \leq \mathrm{p} \leq 5$. . . . 137
11.5 Physical eigenmode; DGSEM, $5 \leq \mathrm{p} \leq 119$. . . . 138
11.6 All eigenmodes; FR/CPR, $\mathrm{p} = 2$, $\eta_{-1/2}$ . . . 139
11.7 All eigenmodes; FR/CPR, $\mathrm{p} = 2$, η_{DG} . . . 139
11.8 All eigenmodes; FR/CPR, $\mathrm{p} = 2$, η_{Ga} . . . 140
11.9 All eigenmodes; FR/CPR, $\mathrm{p} = 2$, η_2 . . . 140
11.10 All eigenmodes; FR/CPR, $\mathrm{p} = 2$, η_∞ . . . 141
11.11 All eigenmodes; FR/CPR, $\mathrm{p} = 3$, $\eta_{-1/2}$ . . . 141
11.12 All eigenmodes; FR/CPR, $\mathrm{p} = 3$, η_{DG} . . . 142
11.13 All eigenmodes; FR/CPR, $\mathrm{p} = 3$, η_{Ga} . . . 142
11.14 All eigenmodes; FR/CPR, $\mathrm{p} = 3$, η_2 . . . 143
11.15 All eigenmodes; FR/CPR, $\mathrm{p} = 3$, η_∞ . . . 143
11.16 Physical eigenmode; FR/CPR, $2 \leq \mathrm{p} \leq 5$, $\eta_{-1/2}$ . . . 144
11.17 Physical eigenmode; FR/CPR, $2 \leq \mathrm{p} \leq 5$, η_{DG} . . . 145
11.18 Physical eigenmode; FR/CPR, $2 \leq \mathrm{p} \leq 5$, η_{Ga} . . . 146
11.19 Physical eigenmode; FR/CPR, $2 \leq \mathrm{p} \leq 5$, η_2 . . . 147
11.20 Physical eigenmode; FR/CPR, $2 \leq \mathrm{p} \leq 5$, η_∞ . . . 148
11.21 Physical eigenmode; FR/CPR, $\eta_{-1/2} \leq \eta \leq \eta_\infty$, $\mathrm{p} = 2$ . . . 149
11.22 Physical eigenmode; FR/CPR, $\eta_{-1/2} \leq \eta \leq \eta_\infty$, $\mathrm{p} = 3$ . . . 150
11.23 Physical eigenmode; FR/CPR, $\eta_{-1/2} \leq \eta \leq \eta_\infty$, $\mathrm{p} = 4$ . . . 151
11.24 Physical eigenmode; FR/CPR, $\eta_{-1/2} \leq \eta \leq \eta_\infty$, $\mathrm{p} = 5$ . . . 152
11.25 Physical eigenmode; FR/CPR, $\eta_{-1/2} \leq \eta \leq \eta_\infty$, $\mathrm{p} = 119$ . . . 153
11.26 All eigenmodes; DGIGA, $\mathrm{k} = 2$, $\mathrm{p} = 7$, C^0 . . . 154
11.27 All eigenmodes; DGIGA, $\mathrm{k} = 2$, $\mathrm{p} = 8$, C^0 . . . 154

11.28 Physical eigenmode; DGIGA, $\mathrm{k} = 2$, $2 \leq \mathrm{p} \leq 5$, C^0 . . . 155
11.29 Physical and "bubble" eigenmodes; DGIGA, $\mathrm{k} = 4$, $2 \leq \mathrm{p} \leq 5$, C^0 . . . 156
11.30 Physical and "bubble" eigenmodes; DGIGA, $\mathrm{k} = 8$, $2 \leq \mathrm{p} \leq 5$, C^0 . . . 157
11.31 Physical eigenmode; DGIGA, $\mathrm{k} = 2$, $2 \leq \mathrm{p} \leq 5$, $C^{\mathrm{p}-1}$ . . . 158
11.32 Physical eigenmode; DGIGA, $\mathrm{k} = 4$, $2 \leq \mathrm{p} \leq 5$, $C^{\mathrm{p}-1}$ . . . 159
11.33 Physical eigenmode; DGIGA, $\mathrm{k} = 8$, $2 \leq \mathrm{p} \leq 5$, $C^{\mathrm{p}-1}$ . . . 160
11.34 Physical eigenmode; DGIGA, $\mathrm{k} = 16$, $2 \leq \mathrm{p} \leq 5$, $C^{\mathrm{p}-1}$ . . . 161
11.35 Physical and "bubble" eigenmodes; DGIGA, $\mathrm{k} = 32$, $2 \leq \mathrm{p} \leq 5$, $C^{\mathrm{p}-1}$ . . . 162
11.36 Physical eigenmode; DGIGA, $1 \leq \mathrm{k} \leq 8$, $\mathrm{p} = 2$, C^0 . . . 163
11.37 Physical and "bubble" eigenmodes; DGIGA, $1 \leq \mathrm{k} \leq 8$, $\mathrm{p} = 3$, C^0 . . . 164
11.38 Physical and "bubble" eigenmodes; DGIGA, $1 \leq \mathrm{k} \leq 8$, $\mathrm{p} = 4$, C^0 . . . 165
11.39 Physical and "bubble" eigenmodes; DGIGA, $1 \leq \mathrm{k} \leq 8$, $\mathrm{p} = 5$, C^0 . . . 166
11.40 Physical and "bubble" eigenmodes; DGIGA, $1 \leq \mathrm{k} \leq 32$, $\mathrm{p} = 2$, C^1 . . . 167
11.41 Physical eigenmode; DGIGA, $1 \leq \mathrm{k} \leq 32$, $\mathrm{p} = 3$, C^2 . . . 168
11.42 Physical eigenmode; DGIGA, $1 \leq \mathrm{k} \leq 32$, $\mathrm{p} = 4$, C^3 . . . 169
11.43 Physical eigenmode; DGIGA, $1 \leq \mathrm{k} \leq 32$, $\mathrm{p} = 5$, C^4 . . . 170
11.44 Physical eigenmode; DGIGA, $\mathrm{k} = 2$, $\mathrm{p} = 4$, C^0 to C^3 . . . 171
11.45 Physical eigenmode; DGIGA, $\mathrm{k} = 4$, $\mathrm{p} = 4$, C^0 to C^3 . . . 172
11.46 Physical and "bubble" eigenmodes; DGIGA, $\mathrm{k} = 8$, $\mathrm{p} = 4$, C^0 to C^3 . . . 173
11.47 Physical and "bubble" eigenmodes; DGIGA, $\mathrm{J} = 22$ (several combinations) . . . 174

12.1 Combined-mode errors, optimum vs. baseline ($\mathrm{J} = 6$) . . . 180
12.2 Combined-mode errors, optimum vs. baseline ($\mathrm{J} = 8$) . . . 181
12.3 Combined-mode errors, optimum vs. baseline ($\mathrm{J} = 11$) . . . 181
12.4 Combined-mode errors, optimum vs. baseline ($\mathrm{J} = 15$) . . . 181
12.5 Combined-mode errors, optimum vs. baseline ($\mathrm{J} = 20$) . . . 182
12.6 Dispersion to dissipation ratios, optima vs. baselines . . . 182
12.7 Cost of DGSEM as a function of resolution . . . 185
12.8 Cost of FR as a function of resolution . . . 185
12.9 Cost of DGIGA as a function of resolution . . . 186

13.1 Initial conditions in wavenumber domain . . . 191
13.2 Unexplained nonlinear instability; all DGIGA variants vs. DGSEM . . . 191
13.3 Shock formation and transit across the domain; DGIGA vs. DGSEM . . . 192
13.4 Solution and error norm for linear advection, $\mathrm{J} = 3$ . . . 193
13.5 Solution and error norm for Burgers' equation, $\mathrm{J} = 3$ . . . 194
13.6 Solution and error norm for linear advection, $\mathrm{J} = 6$ . . . 195
13.7 Solution and error norm for Burgers' equation, $\mathrm{J} = 6$ . . . 196
13.8 Solution and error norm for linear advection, $\mathrm{J} = 20$ . . . 197
13.9 Solution and error norm for Burgers' equation, $\mathrm{J} = 20$ . . . 198
13.10 Limitations of a posteriori modified wavenumber analysis . . . 200
13.11 A posteriori modified wavenumber analysis; spatial discretizations . . . 201
13.12 A posteriori modified wavenumber analysis; Riemann solvers . . . 202

14.1 Conditioning issues in modal IGA discretizations . . . 208
14.2 Inter-cell limiters, $\mathrm{p} = 2$ DGSEM . . . 213
14.3 Inter-cell limiters, $\mathrm{p} = 5$ DGSEM . . . 214
14.4 Inter-cell limiters, $\mathrm{p} = 2$ DGSEM . . . 215
14.5 Inter-cell limiters, $\mathrm{p} = 5$ DGSEM . . . 216
14.6 Limiting $\mathrm{J} = 3$ DGIGA . . . 217
14.7 Limiting, $\mathrm{J} = 6$ DGIGA . . . 218
14.8 FCT limiting for IGA-AFC . . . 219
14.9 Final comparison: DGSEM vs. FR/CPR vs. DGIGA; $\mathrm{J} = 3$ . . . 220
14.10 Final comparison: DGSEM vs. FR/CPR vs. DGIGA; $\mathrm{J} = 6$ . . . 221
14.11 Final comparison: DGSEM vs. FR/CPR vs. DGIGA; $\mathrm{J} = 20$ . . . 222
14.12 Effect of the number of patches in FCT-limited DGIGA-AFC . . . 223
14.13 High-order preservation (or lack thereof) of the three main limiters considered . . . 224

A.1 Example of resolvable wavenumbers in a discrete setting 239
A.2 Spectral resolution examples 240
A.3 Dispersion, dissipation and Fourier footprint of low-order finite volume schemes . 248
A.4 Dispersion/dissipation Bloch waves of low-order finite volume schemes 249

List of Tables

2.1 All possible Riemann problems for Burgers' equation . . . 16

4.1 DGSEM reference element mass matrices and their condition numbers . . . 36
4.2 Quadrature rules (Gauss-Legendre vs. Gauss-Lobatto) . . . 36

5.1 Summary of reported properties of various FR/CPR variants . . . 46
5.2 Correction parameters c vs. polynomial degree . . . 48

7.1 Comparison among optimal explicit SSP-RK methods . . . 72

8.1 Summary of stability criteria . . . 80

9.1 Empty test matrix, for example purposes . . . 99

10.1 Verifying the order of accuracy of the SSP-RK schemes . . . 106
10.2 Exploring DGSEM's parameter space . . . 106
10.3 Exploring FR/CPR's parameter space . . . 106
10.4 Exploring DGIGA's parameter space . . . 107

12.1 Baseline DGSEM semi-discretizations . . . 178
12.2 Optimal FR/CPR semi-discretizations . . . 178
12.3 Optimal DGIGA semi-discretizations . . . 178
12.4 All DGIGA semi-discretizations with $\mathrm{J} = 3, 4, 5, 6, 8, 11, 15, 20$ . . . 179
12.5 Dispersion to dissipation ratio norms . . . 183

13.1 Confirming the advantage of optimized schemes (most favorable conditions) . . . 187
13.2 Testing the advantage of optimized schemes (unfavorable conditions) . . . 188
13.3 Exploring the behavior of optimized schemes applied to the Burgers equation . . . 189
13.4 Investigating the unexplained nonlinear instability of DGIGA . . . 190

14.1 Comparing inter-cell limiters used in combination with DGSEM . . . 204
14.2 Comparing sensors, against each other and none at all, for DGSEM . . . 205
14.3 Comparing AFC against the best inter-cell limiter, for DGIGA . . . 206
14.4 Attempting IGA-AFC, the extreme case of DGIGA with a single patch . . . 208
14.5 Comparing optimal vs. baseline DG bases, for $\mathrm{J} = 3, 6, 20$ . . . 209
14.6 Isolating the effect of the number of patches in FCT-limited DGIGA-AFC . . . 210
14.7 Checking whether the three main limiters considered preserve accuracy . . . 211
14.8 Inter-cell limiters, DGSEM . . . 225
14.9 Sensors, DGSEM . . . 226
14.10 Limiters and sensors, DGIGA . . . 227
14.11 IGA-AFC . . . 228
14.12 Limiters and sensors, DGSEM vs. FR/CPR vs. DGIGA . . . 229
14.13 Effect of the number of patches in FCT-limited DGIGA-AFC . . . 229
14.14 Order of accuracy when limiting smooth local extrema . . . 230
14.15 Effect of p-refinement (for a fixed number of elements/patches) in practice . . . 230

Nomenclature

Scalars are typeset in regular typeface. Vectors use lower case letter symbols and bold typeface; their default orientation is along columns. Matrices, also in bold, are represented by upper-case letters only.

Symbols

Latin letters

$\boldsymbol{A}$	Jacobian matrix
A_T	spectral order of accuracy
C^*	differentiability class
c	speed of sound
$\boldsymbol{\mathcal{C}}$	discrete gradient operator matrix
c_{SSP}	SSP coefficient
$\boldsymbol{\mathcal{D}}$	Lagrange nodal derivative matrix
E	total energy
e	internal energy
e_1	resolving efficiency
E_T	spectral error
$\boldsymbol{f}$	flux vector
$\boldsymbol{f}^h$	approximate flux vector
$\mathcal{F}$	Fourier transform
$\check{\boldsymbol{f}}$	Riemann flux vector
$\hat{\boldsymbol{f}}$	vector of flux expansion coefficients
$\check{\boldsymbol{f}}$	nodal flux vector values
$\boldsymbol{f}$	raw antidiffusive flux vector
$\tilde{\boldsymbol{f}}$	flux vector in reference coordinates
G	amplification factor
g	correction function
$\mathcal{G}^h$	set of all ghost elements or patches in a mesh
$\boldsymbol{h}$	corrected flux vector
H	enthalpy
$\check{\boldsymbol{h}}$	corrected flux vector at nodes
$\tilde{\boldsymbol{h}}$	corrected flux vector in reference coordinates
i	equation index
$\boldsymbol{I}$	identity matrix
I	number of equations in a system of PDEs
$\Im(\bullet)$	imaginary part
i	imaginary unit
$\inf$	infimum
j	degree of freedom index
J	number of basis functions in an element or patch
k	element or patch index
K	number of elements or patches in the mesh
k	number of breakpoint spans in a patch
l	Lagrange polynomial
$\boldsymbol{\mathcal{M}}$	mass matrix
m	multiplicity (of a knot)
N	B-spline basis function
nnz	number of nonzero entries
$\mathcal{O}$	big O notation
p	polynomial degree
p	pressure
$\mathcal{P}$	Legendre polynomial
$\boldsymbol{q}$	state vector
$\boldsymbol{q}^h$	approximate state vector
$\boldsymbol{q}$	vector of characteristic variables
$\check{\boldsymbol{q}}$	Riemann state vector
$\hat{\boldsymbol{q}}$	vector of solution expansion coefficients
$\hat{\boldsymbol{q}}$	vector of solution expansion coefficients in characteristic variables
$\check{\boldsymbol{q}}$	nodal solution vector values
$\tilde{\boldsymbol{q}}$	state vector in reference coordinates
$\widehat{q}$	time-dependent Fourier coefficient
$\widehat{\boldsymbol{q}}$	time-dependent, degree of freedom–wise Fourier coefficients vector
$\widehat{\boldsymbol{q}}$	time-dependent, eigenmode-wise Fourier coefficients vector
$\widehat{\widehat{q}}$	Fourier coefficient
$\widehat{\widehat{\boldsymbol{q}}}$	degree of freedom–wise Fourier coefficients vector
$\widehat{\widehat{\boldsymbol{q}}}$	eigenmode-wise Fourier coefficients vector
$\boldsymbol{r}$	residual function vector
$\boldsymbol{\mathcal{R}}$	residual matrix (linear operator)
$\Re(\bullet)$	real part
$\hat{\boldsymbol{r}}$	vector of residual expansion coefficients
$\tilde{\boldsymbol{r}}$	reference residual function vector
Re	Reynolds number
S^h	space of shape functions
$\sup$	supremum
$\mathcal{T}^h$	set of all elements or patches in a mesh
$\mathcal{T}^*$	set of all troubled elements or patches in a mesh
Δt	time-step size
ΔT	total simulated time span
$\boldsymbol{T}$	state to primitive variables transformation matrix
u	velocity
V	vector space

V^h	space of test functions
$\boldsymbol{V}$	eigenvector matrix of the spatial discretization operator
$\boldsymbol{v}$	vector of primitive variables
$\boldsymbol{\mathcal{V}}$	Vandermonde matrix
$\boldsymbol{w}$	Roe state vector
Δx	element or patch length

Greek letters

γ	ratio of specific heats
κ	wavenumber
$\varkappa$	smoothness
κ_f	highest well-resolved wavenumber
$\kappa_{1\%}$	cutoff (1% criterion) wavenumber
$\widetilde{\kappa}$	modified wavenumber
λ	eigenvalue
Ξ	B-spline knot vector
ξ	position in reference coordinates
ρ	density
Σ	knot span
$\widetilde{\Sigma}$	reference knot span
ς	Courant number
ϕ	shape (basis) function
φ	test (weighting) function
χ	dispersion vs. dissipation ratio
$\mathcal{X}$	affine mapping to reference element
$\Delta\Psi$	phase shift angle
Ω	(sub)domain
$\partial\Omega$	boundary of a (sub)domain
$\widetilde{\Omega}$	reference subdomain (element or patch)

Others

$\lvert\bullet\rvert$	absolute value
$\bar{\bullet}$	average
$\lceil\bullet\rceil$	ceiling (round up to nearest integer)
$\bullet^*$	dimensionless
$\oplus$	direct sum
$\emptyset$	empty set
$\odot$	Hadamard product
$\langle\bullet,\bullet\rangle$	inner product
$[\bullet]_\bullet^\bullet$	e.g.: $[f]_a^b = f(b) - f(a)$
$\lVert\bullet\rVert_\bullet$	norm (or norm-like operation)
$\bullet^\intercal$	transposed

Abbreviations

ADIGMA adaptive higher-order variational methods for aerodynamic applications in industry

AFC algebraic flux correction

AI artificial intelligence

BC boundary condition

BDF Biswas-Devine-Flaherty (DG limiter)

BSB Burbeau-Sagaut-Bruneau (DG limiter)

CAD computer-aided design

CFD computational fluid dynamics

CFL Courant-Friedrichs-Lewy (stability condition)

CG continuous Galerkin

CPR correction procedure via reconstruction

CPU central processing unit

DG discontinuous Galerkin method

DGIGA discontinuous Galerkin isogeometric analysis

DGIGA-AFC discontinuous Galerkin isogeometric analysis stabilized via algebraic flux correction

DGSEM discontinuous Galerkin spectral element method

DNS direct numerical simulation

ENO essentially non-oscillatory

FCT flux corrected transport

FDM finite difference method

FEA finite element analysis

FEM finite element method

FFT fast Fourier transform

FLOP floating point operation

FR flux reconstruction

FVM finite volume method

HiFi-TURB high-fidelity LES/DNS data for innovative turbulence models

HLL Harten-Lax-van Leer (Riemann solver)

HLLC Harten-Lax-van Leer-contact (Riemann solver)

HLLE Harten-Lax-van Leer-Einfeldt (Riemann solver)

HWENO Hermite weighted essentially non-oscillatory

IC initial condition

IDIHOM industrialization of high-order methods

IGA isogeometric analysis

KEP kinetic energy preserving (centered numerical flux)

KXRCF Krivodonova-Xin-Remacle-Chevaugeon-Flaherty (troubled element indicator)

LED local extremum diminishing

LES large eddy simulation

MATLAB matrix laboratory (programming language and environment)

MWA modified wavenumber analysis

NURBS non-uniform rational basis splines

ODE ordinary differential equation

PDE partial differential equation

RANS Reynolds-averaged Navier-Stokes

RK Runge-Kutta (a numerical integration technique)

RKDG Runge-Kutta discontinuous Galerkin

SD spectral differences

SEM spectral element method

SG staggered grid

SM spectral method

SSP strong stability preserving

TILDA towards industrial LES/DNS in aeronautics

TV total variation

TVB total variation bounded

TVBM total variation bounded in the means

TVD total variation diminishing

TVDM total variation diminishing in the means

TVM total variation in the means

VCJH Vincent-Castonguay-Jameson-Huynh (class of FR correction functions)

WENO weighted essentially non-oscillatory

1 Introduction

In computational fluid dynamics (CFD), a spatial discretization method is said to be high-order when its order of accuracy[1] is higher than two [129]. High order is not a new concept, and the number of high-order methods in the literature is considerable; these are reviewed in [126] and references therein. This work focuses on one particular type of high-order methods: compact, discontinuous, finite element methods. The epitome of these schemes is the discontinuous Galerkin (DG) method.

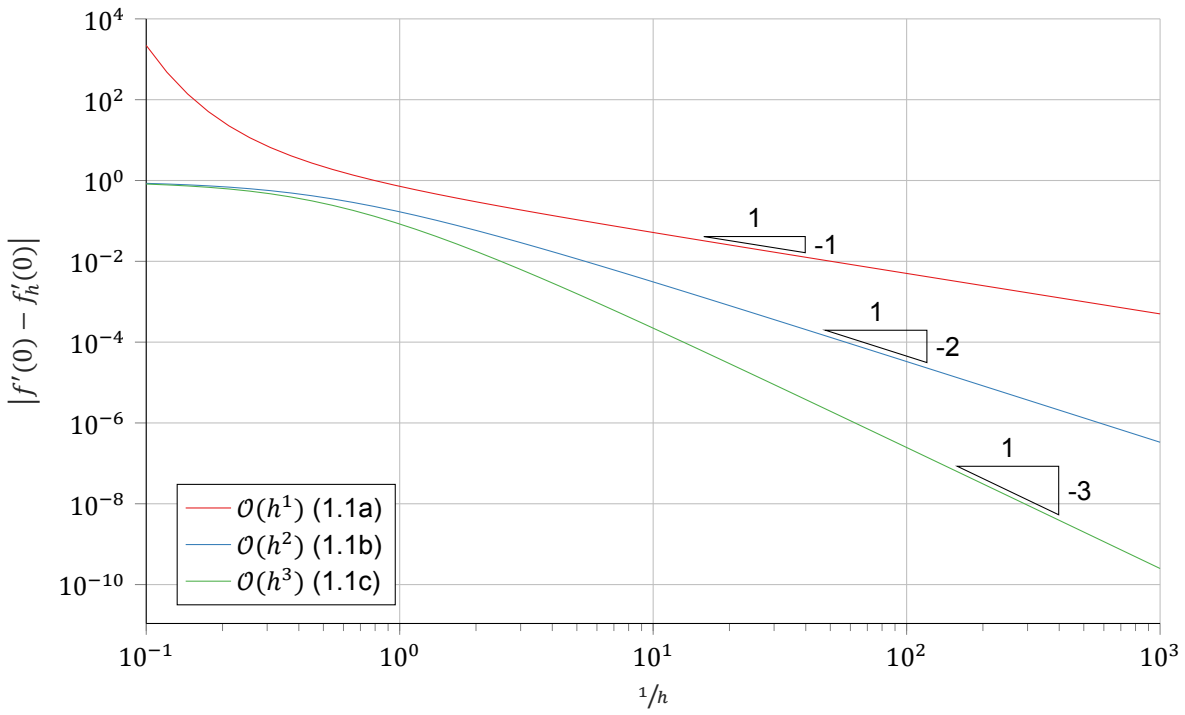


Figure 1.1: Convergence rate of the three finite difference schemes in (1.1).

High-order methods are *potentially* more cost-effective than their low-order counterparts, provided that a sufficiently-high level of accuracy is required. Allow me to motivate this through an example. Assume that we wish to use finite differences to approximate $f'(x)$, the first derivative of $f(x) = \mathrm{e}^x$, on a uniform grid—every two points separated by a distance h. Its 1st, 2nd and 3rd order (fully backward,

[1] Order of accuracy of a numerical scheme refers to the rate at which the error in the approximate solution it produces is reduced, as the discrete problem it solves is refined. A method of order n has an error that grows as $\mathcal{O}(h^n)$, where h is a representative grid length. See figure 1.1.

uniform) finite difference approximations are, respectively [18]:

$$f'(x) \approx \frac{f(x) - f(x-h)}{h}, \tag{1.1a}$$

$$f'(x) \approx \frac{3f(x) - 4f(x-h) + f(x-2h)}{2h}, \tag{1.1b}$$

$$f'(x) \approx \frac{11f(x) - 18f(x-h) + 9f(x-2h) - 2f(x-3h)}{6h}. \tag{1.1c}$$

Let us now plot the error of each of the approximations at $x = 0$, as a function of $1/h$, in logarithmic axes; the result is figure 1.1. To reach an error of 1% at $x = 0$, methods (1.1b) and (1.1a) require roughly 2 and 20 times as many points, respectively, than method (1.1c). At the same time, however, each evaluation of (1.1c) involves four points; those are 33% more than (1.1b), and twice as many as (1.1a). This is the core feature of high-order methods: for a fixed accuracy, they reduce the total number of unknowns in the discrete problem in exchange of using each of them an increased number of times.

1.1. Motivation: next-generation CFD

Unstructured, low-order, finite volume-based Navier-Stokes solvers, despite having been developed in the 80s and 90s, have remained in predominant use for aerospace applications to this day [61, 63, 123]. High-order discretizations are now making their way into a new generation of production Navier-Stokes solvers [79, 126]. These methods are considered to be necessary for certain cases of commercial interest where current-generation codes fail to provide accurate results, namely [123]:

- To increase precision in boundary layers, needed to predict the onset of turbulent transition
- Simulating vortex-dominated flows (e.g. in the wake of an aircraft in high-lift configuration)
- To simulate aero-acoustics
- Predicting drag forces
- Simulating turbulence using large eddy simulation (LES)

High-order methods have been receiving renewed attention since the 2000s, perhaps most notably through the sequence of European projects ADIGMA [70] (2006–2009), IDIHOM [71] (2010–2014) and TILDA [51] (2015–2018). Universities, public research institutions and industry have all collaborated in these projects. Development has been focused on the following pacing items [126]:

- Commercial quality high-order mesh generation tools
- Robust error estimates and hp-adaptations
- Scalable, efficient, robust and low-memory implicit solvers
- Parameter free, accuracy preserving and convergent shock capturing

Broadly speaking, high-order schemes possess two advantages. The first has to do with the balance between accuracy and cost:

$$\text{efficiency} = \frac{\text{accuracy}}{\text{cost}} = \underbrace{\left(\frac{\text{accuracy}}{\text{unknowns} \times \text{time steps}}\right)}_{\textbf{Discretization}} \underbrace{\left(\frac{\text{unknowns} \times \text{time steps}}{\text{cost}}\right)}_{\textbf{Implementation}}. \tag{1.2}$$

High-order methods tend to have superior accuracy per unknown; we have seen this through an example. In order to be efficient, however, discretization efficiency is only half of the picture—implementation efficiency is just as important. Yet, some high-order schemes (such as DG) are quite good at maximizing that as well.

The throughput (number of unknowns processed at a given cost) of high-order methods can never be higher than that of low-order ones [73]—in the example, evaluating (1.1c) is more costly (it takes longer, since it involves additional terms) than (1.1b) or (1.1a). Nevertheless, because they possess a

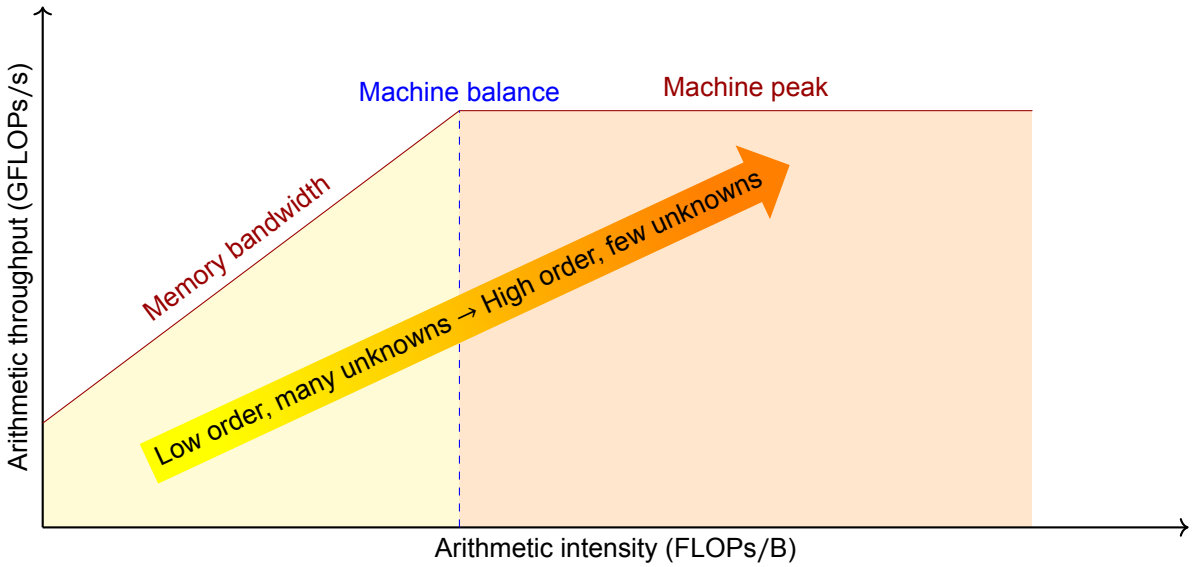


Figure 1.2: Roofline model [130] representing the arithmetic throughput (number of operations per second, i.e. performance) of a multi-core architecture. The arithmetic intensity—number of floating point operations done on each byte of information retrieved from memory—of a *program* (notice that this is hardware-independent) determines if execution will be memory-bound (yellow-shaded, left) or compute-bound (orange-shaded, right). For optimal use of hardware resources, the algorithm should operate at the balance point. High-order methods may *simultaneously* increase the arithmetic intensity and lower the total amount of memory required in CFD solvers, possibly making a more efficient use of current hardware. Supercomputers are typically being used nowadays at merely 5 % to 20 % of their theoretical peak performance [80].

higher degree of data reuse, high-order methods tend to have a higher *arithmetic intensity* than well-optimized low-order ones [86]: for every byte of information transferred from main memory to cache, more floating point operations are made. High performance computing technology evolution is currently experiencing a so-called “memory wall” barrier: there is an exponentially increasing disparity between compute and memory speeds (the latter is lagging behind). For example [73]:

- Nvidia A100 (108 streaming multiprocessors, 765 MHz, released in 2020): 6.20 FLOPs/B
- Intel Xeon Platinum 8280 (28 cores, 2.70 GHz; released in 2019): 17.20 FLOPs/B
- Intel Xeon W5590 (4 cores, 3.33 GHz; released in 2009): 1.70 FLOPs/B

while Kronbichler and Persson [73] estimate 10 FLOPs/B for a fine-tuned DG implementation. This means that high-order methods tend to utilize modern hardware more efficiently, compensating (to some extent) the increase in floating point operations with a higher attained fraction of peak machine throughput (figure 1.2). In more concise terms: they exploit the “FLOPs are for free” paradigm to attain a cost per unknown comparable to their low-order counterparts. On a modern supercomputer architecture, this translates as them having good scaling *within* a node, i.e. minimizing the overhead due to slow data transfer rates between main memory and CPU/GPU caches.

Moreover, in compact variants (such as DG), each grid element uses data from a single layer of points of each of its nearest neighbors only, regardless of order. This means that communication of data between nodes is not usually a bottleneck, and lends DG an excellent parallel scalability [73]. And, on top of it all, DG has very good synergy with adaptive refinement: it readily supports unstructured meshes and non-conforming discretizations, and the finite element formulation it is based on gives it a strong mathematical foundation on which to build adjoint-based error estimators [46].

The second advantage of high-order methods has to do with novel approaches to tackle vortex-dominated and turbulent flows. High order is widely thought to be required for LES and direct numerical simulation (DNS) [111, 118, 123]. This is because, in relation to low-order finite volumes, higher-order discretizations significantly reduce dispersion and dissipation errors per unknown. These have a dominant contribution on the overall error in high–Reynolds number flows [72], since in that case physical

diffusion of momentum and dissipation of energy due to viscosity is small relative to convection. This can, in principle, be compensated with more elements using a low-order discretization; the issue then, however, is that meshes become impractically large.

Even with high-order methods, LES is likely to remain too expensive to replace Reynolds-Averaged Navier Stokes (RANS) turbulence models in the foreseeable future. In fact, the latest trend in turbulence modeling is to build improved RANS models based on flow data, using machine learning techniques. This data must come from somewhere; typically, and most conveniently, a numerical simulation. High order is therefore far from unnecessary in this new AI-accelerated paradigm: it is, in fact, instrumental to its success. Proof of this is the European project HiFi-TURB [35].

It is clear that high-order methods have a crucial role to play in the ongoing initiative to satisfy the ambitious performance, efficiency and sustainability goals required of the aviation industry in the upcoming decades.

1.2. Research objectives

In this thesis, I focus on a particular approach of achieving high order, known as discontinuous Galerkin (DG)—and a very close relative, flux reconstruction (FR). The goal is to study fundamental properties of three high-order methods:

- Discontinuous Galerkin spectral element method (DGSEM)
- Flux reconstruction (FR)
- Discontinuous Galerkin isogeometric analysis (DGIGA), particularly when coupled with algebraic flux correction as its limiter (DGIGA-AFC)

such that a preliminary assessment can be made on whether the third, DGIGA-AFC, is more or less suitable to LES of compressible flows relative to the first two.

In terms of research questions, this work hopes to answer:

Is the DGIGA-AFC method, in comparison to selected alternatives (DGSEM and FR/CPR, with modal limiters), well-suited to LES of high-speed flows?

a response, in turn, steered by the following research sub-questions:

1. *Does DGIGA achieve high order of accuracy, in the same conditions that FR/CPR and DGSEM do?*
2. *What are DGIGA's spectral (dispersion, dissipation) and linear stability characteristics, and how do they compare to those of DGSEM and FR/CPR?*
3. *Which FR/CPR and DGIGA settings are most advantageous, in terms of spectral and/or linear stability features, in relation to DGSEM?*
4. *Are DGIGA, FR/CPR and DGSEM's wave propagation characteristics actually representative of their performance in nonlinear physics cases?*
5. *Does DGIGA retain any advantage over FR/CPR and DGSEM when used with limiters? Is the AFC limiter applied to DGIGA (DGIGA-AFC) as effective as alternatives?*

Which translate, in terms of objectives, to:

1. Confirm the order of accuracy of DGIGA, FR/CPR and DGSEM
2. Obtain, for the three methods (several representative configurations of each) applied to a linear PDE:
   (a) Dispersion relation (dispersive error vs. wavenumber)
   (b) Dissipation relation (dissipative error vs. wavenumber)
   (c) Linear stability bounds (largest stable time-step size)

3. Optimize the configuration of FR/CPR and DGIGA, so that dispersion/dissipation are minimized (also under linear PDE assumptions)
4. Obtain numerical evidence supporting the linear wave propagation analysis predictions in nonlinear PDE cases
5. Test whether DGIGA-AFC is a successful limited scheme or not, comparing its performance with alternatives on several benchmark problems

1.3. Relevance of this work

DGIGA-AFC addresses two of the pacing items of high-order methods:

- High-order mesh generation: indirectly, through the inherent advantages of isogeometric analysis (I do not actually measure these, since all tests in this thesis are in one spatial dimension)
- Limiting: by directly testing a novel limiter-discretization combination

This will be, to the best of my knowledge, the first exhaustive exploration of spectral and stability features of DGIGA in the literature. It will also be the first time that modified wavenumber analysis has been formulated for and applied to it.

The first two schemes, DGSEM and FR, are relatively mature; their properties have been studied before. Still, they will serve as baselines (I see them as control and placebo, respectively) with which to compare the characteristics and effectiveness of the novel DGIGA. Additionally, they are included in the study for verification purposes. By taking a step back, and focusing on the fundamentals, this work offers the opportunity to judge the performance and suitability of DGIGA against more mature alternatives in a controlled manner, i.e. with full knowledge of the conditions in which the comparison is made. This research should therefore be of most value to researchers and/or university departments considering an investment of resources in the isogeometric analysis idea.

Lastly, the fact that all formulation is particularized to 1D and self-contained in this report, and that I have made an effort to present high-order schemes in an easy to understand way for someone familiar with FV only (my own situation at the start of this thesis), should make this work adequate as a first introduction to discontinuous and isogeometric finite element methods.

1.4. Outline

This report is divided in two parts. Part I is theoretical; it offers a succinct overview of all concepts involved in this thesis, making it essentially self-contained. Its purpose is to make the results and methodology as transparent and reproducible as possible, by removing all ambiguity from the formulation of the schemes and numerical analysis tools involved. I start by presenting relevant aspects of hyperbolic partial differential equations (chapter 2), as well as a complete mathematical formulation of all ingredients necessary for constructing the three spatial discretization methods under consideration (chapters 3–6). Next, in chapter 7, I give a brief overview of the kind of explicit time integration schemes I later use in part II. This first part ends with some basic aspects about nonlinear stabilization, including the formulation of all sensors and limiters in the precise way that I later employ them (chapter 8). That includes AFC; in particular, my own extension of it to multi-patch IGA.

In part II, all actual experimental results obtained in this project are presented and discussed. First, in chapter 9, I clarify some methodology aspects; these include details about all test problems solved, and the implementation used. Five chapters of experimental results then follow, 10–14, each addressing one of the research sub-questions and objectives above. Lastly, in chapter 15, the aforementioned research (sub-)questions are answered one by one—recounting the entirety of tasks performed in the process. It also includes limitations of this thesis, and recommendations for future work.

I

Theory

2

Hyperbolic Conservation Laws

The Navier-Stokes equations are the set of scientific laws which describe the evolution of fluid flows. They arise from the following conservation argument[1]: *in any given portion of space filled with fluid, any change in the total amount of mass, momentum or energy over time is only possible if there exists a net flux of that quantity in (or out) of such domain, and/or there is a source (or sink) of the same within said domain* [50, §1.1]. Each of these three quantities then has a partial differential equation (PDE) that encodes the previous relationship mathematically. This system of equations happens to be, in general, of mixed mathematical type; its (unsteady) inviscid counterpart, however, is hyperbolic [50, §2]. This is the reason why the present work focuses on conservation laws of this particular kind.

The three research objects of this work will be studied for a general hyperbolic conservation law, which is defined in §2.1. The definition of hyperbolicity for a system of PDEs is given in §2.2. The fact that (the integral form) of these problems allows discontinuous solutions is addressed §2.3. Section 2.4 does the same for the Riemann problem, which plays a fundamental role in discontinuous discretizations of the type considered. Finally, relevant details of a canonical example of each combination of types of hyperbolic conservation laws, from simplest (scalar and linear) to most complex (vector and nonlinear) are presented in §§ 2.5-2.8. In consonance with the scope of this work, only the one-dimensional version of such equations is addressed and none of them includes source terms (cf. [82, §§ 17-18]).

2.1. Conservation laws in one dimension

The idea of conservation is intuitive for the particular example of fluid flows. Mathematically, however, it can be extended to a generic quantity, regardless of any eventual physical interpretation. This report is concerned with systems of PDEs of the following general form:

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{f}}{\partial x} = 0\,. \tag{2.1}$$

In (2.1), $t \in \mathbb{R}^+$ and $x \in \Omega$ are independent variables. The (one-dimensional) spatial domain of the PDEs is $\Omega \subseteq \mathbb{R}$. Its boundary can be split into left and right portions, such that $\partial\Omega = \partial\Omega^L \cup \partial\Omega^R$; their coordinates are x_L and x_R. With no loss of generality, time is assumed to start at 0 ($t \geq 0$) and to have no upper bound. Dependent ones are grouped into a vector of conserved variables $\boldsymbol{q}(t,x) \in \mathbb{R}^{\mathrm{I}}$ (or *state vector*),

$$\boldsymbol{q}(t,x) \coloneqq \begin{bmatrix} q_1(t,x) & q_2(t,x) & \cdots & q_{\mathrm{I}}(t,x) \end{bmatrix}^{\mathrm{T}}\,, \tag{2.2}$$

and $\boldsymbol{f} \colon \mathbb{R}^{\mathrm{I}} \to \mathbb{R}^{\mathrm{I}}$, the vector of conserved variable fluxes (or *flux vector*)

$$\boldsymbol{f}(\boldsymbol{q}) \coloneqq \begin{bmatrix} f_1(\boldsymbol{q}) & f_2(\boldsymbol{q}) & \cdots & f_{\mathrm{I}}(\boldsymbol{q}) \end{bmatrix}^{\mathrm{T}}\,, \tag{2.3}$$

[1] It can be argued that the Navier-Stokes equations are *not* conservation laws in a strict sense, because they include viscous dissipation terms. Regardless, the definition of conservation given above is broad enough to still apply.

which, for the physical models under consideration, will be some vector-valued function of the state vector only[2]. Thus, $\boldsymbol{q}(t,x)$ is the solution of an initial-boundary value problem defined by (2.1) together with some *initial condition*

$$\boldsymbol{q}^0(x) \coloneqq \boldsymbol{q}(0,x)\,, \tag{2.4}$$

and compatible *boundary conditions*

$$\boldsymbol{q}^L(t) \coloneqq \boldsymbol{q}(t,x_L)\,, \tag{2.5}$$
$$\boldsymbol{q}^R(t) \coloneqq \boldsymbol{q}(t,x_R)\,. \tag{2.6}$$

2.1.1. Quasilinear form

Applying the chain rule to (2.1) leads to

$$\frac{\partial \boldsymbol{q}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{q}}{\partial x} = 0\,, \tag{2.7}$$

which is known as the quasilinear form of the PDEs [82, §2.6]. The so-called *Jacobian* of this system is, for a one-dimensional domain, a single square matrix $\boldsymbol{A} \in \mathbb{R}^{\mathrm{I}\times\mathrm{I}}$, representing the gradient of the flux vector in state space, i.e. :

$$\boldsymbol{A}(\boldsymbol{q}) = \begin{bmatrix} a_{11}(\boldsymbol{q}) & \cdots & a_{1\mathrm{I}}(\boldsymbol{q}) \\ \vdots & & \vdots \\ a_{\mathrm{I}1}(\boldsymbol{q}) & \cdots & a_{\mathrm{II}}(\boldsymbol{q}) \end{bmatrix} \coloneqq \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_\mathrm{I}} \\ \vdots & & \vdots \\ \frac{\partial f_\mathrm{I}}{\partial q_1} & \cdots & \frac{\partial f_\mathrm{I}}{\partial q_\mathrm{I}} \end{bmatrix}, \tag{2.8}$$

with each of its entries being a function of the state vector. In this most general situation, the system is said to be nonlinear (e.g. Euler equations, §2.8). If the entries of $\boldsymbol{A}$ were to dependent on t and/or x instead, the system would be linear with variable coefficients. This case is not considered in this work. The simplest scenario corresponds to a linear system of constant coefficients, were $\boldsymbol{A}$ is constant (e.g. wave equations, §2.6). For the particular case that $\mathrm{I} = 1$, (2.1) becomes a scalar conservation law.

2.1.2. Integral form

Integration of (2.1) over the whole spatial domain leads to

$$\frac{\partial}{\partial t}\left(\int_\Omega \boldsymbol{q}\ \mathrm{d}x\right) + \boldsymbol{f}\left(\boldsymbol{q}^R\right) - \boldsymbol{f}\left(\boldsymbol{q}^L\right) = 0\,. \tag{2.9}$$

Equations (2.1) and (2.9) are equivalent by the divergence theorem[3] provided that the flux function is differentiable (and under the assumption that Ω itself does not change in time). Should $\boldsymbol{f}$ be discontinuous in space, (2.1) would break down but (2.9) would not. For this reason, the latter is said to be a *weaker* representation of the general conservation law, since it relaxes the smoothness required in the solution. Equation (2.9) elegantly casts our introductory definition of a conservation law in precise mathematical terms. It should be noted that (2.9) is but one among several integral representations of (2.1); others can be found in the literature, e.g. [114, §2.4.1].

2.2. Hyperbolicity

A system of equations is said to be hyperbolic if its Jacobian matrix is diagonalizable with real eigenvalues [114, §2.1]. An eigenvalue $\lambda \in \mathbb{C}$ of matrix $\boldsymbol{A}$ is defined as a root of its *characteristic polynomial*, i.e. for which $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$. Also, for every eigenvalue there exists a right eigenvector $\boldsymbol{r} = \begin{bmatrix} r_1 & \cdots & r_\mathrm{I} \end{bmatrix}^\mathsf{T}$ that satisfies $\boldsymbol{A}\boldsymbol{r} = \lambda \boldsymbol{r}$, and a left eigenvector $\boldsymbol{l} = \begin{bmatrix} l_1 & \cdots & l_\mathrm{I} \end{bmatrix}^\mathsf{T}$, that solves $\boldsymbol{l}\boldsymbol{A} = \lambda \boldsymbol{l}$.

[2] More generally (e.g. in the Navier-Stokes case), the flux vector may be a function of both state vector and its first spatial derivative [114, §2.4.2]: $\boldsymbol{f} = \boldsymbol{f}\left(\boldsymbol{q}, \frac{\partial \boldsymbol{q}}{\partial x}\right)$.

[3] The divergence theorem reduces to the fundamental theorem of calculus in one dimension.

By definition, for any *hyperbolic* conservation law there must exist $\boldsymbol{\Lambda}, \boldsymbol{R}, \boldsymbol{L} \in \mathbb{R}^{\mathrm{I}\times\mathrm{I}}$:

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = \begin{bmatrix} \lambda_1(\boldsymbol{q}) & & 0 \\ & \ddots & \\ 0 & & \lambda_{\mathrm{I}}(\boldsymbol{q}) \end{bmatrix}, \tag{2.10a}$$

$$\boldsymbol{R}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{r}_1(\boldsymbol{q}) & \cdots & \boldsymbol{r}_{\mathrm{I}}(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} r_{11}(\boldsymbol{q}) & \cdots & r_{1\mathrm{I}}(\boldsymbol{q}) \\ \vdots & & \vdots \\ r_{\mathrm{I}1}(\boldsymbol{q}) & \cdots & r_{\mathrm{II}}(\boldsymbol{q}) \end{bmatrix}, \tag{2.10b}$$

$$\boldsymbol{L}(\boldsymbol{q}) = \boldsymbol{R}^{-1}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{l}_1(\boldsymbol{q}) & \cdots & \boldsymbol{l}_{\mathrm{I}}(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} l_{11}(\boldsymbol{q}) & \cdots & l_{1\mathrm{I}}(\boldsymbol{q}) \\ \vdots & & \vdots \\ l_{\mathrm{I}1}(\boldsymbol{q}) & \cdots & l_{\mathrm{II}}(\boldsymbol{q}) \end{bmatrix}; \tag{2.10c}$$

such that $\boldsymbol{A} = \boldsymbol{R\Lambda L} \iff \boldsymbol{\Lambda} = \boldsymbol{LAR}$, where $\boldsymbol{\Lambda}$ is the (diagonal) matrix of (real) eigenvalues and $\boldsymbol{A}$ is the Jacobian matrix, as in (2.7); if every λ_i is unique, the system is said to be *strictly* hyperbolic. Each column $\boldsymbol{l}_i$ and $\boldsymbol{r}_i$ of matrices $\boldsymbol{L}$ and $\boldsymbol{R}$, corresponds to the left and right eigenvectors associated with eigenvalue λ_i, respectively. Without loss of generality, these are assumed to be normalized such that $\boldsymbol{LR} = \boldsymbol{RL} = \boldsymbol{I}$. Note that these eigenvalues and eigenvectors, just like the Jacobian matrix that they factorize, are generally functions of the state vector.

2.2.1. Characteristic fields

As discussed at the start of this section, any hyperbolic conservation law is guaranteed to have a set of real eigenvalues $\{\lambda_1(\boldsymbol{q}), \ldots, \lambda_{\mathrm{I}}(\boldsymbol{q})\}$. This enables a more intuitive interpretation of the concept of hyperbolicity: each $\lambda_i(\boldsymbol{q})$ acts as the *propagation speed*[4] of its associated λ_i*-characteristic field*.

In one dimension, each λ-field is a *curve* in space-time, defined as such by the ordinary differential equation (ODE):

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \lambda(q) . \tag{2.11}$$

We can focus on the solution along one of such curves, leaving time as the only independent variable, i.e. $x = x(t)$; in the scalar case, this leads (by virtue of the chain rule) to the following result:

$$\frac{\mathrm{d}q}{\mathrm{d}t} = \frac{\partial q}{\partial t} + \frac{\mathrm{d}x}{\mathrm{d}t}\frac{\partial q}{\partial x} \implies \frac{\mathrm{d}q}{\mathrm{d}t} = \frac{\partial q}{\partial t} + a(q)\frac{\partial q}{\partial x} \implies \frac{\mathrm{d}q}{\mathrm{d}t} \equiv 0 . \tag{2.12}$$

Hence, the scalar hyperbolic conservation law reduces to a constraint: *the value of the solution remains invariant along characteristic curves*. In the linear case with constant coefficients, substitution of (2.10) in (2.7) leads to the *canonical* form of (2.1),

$$\frac{\partial(\boldsymbol{Lq})}{\partial t} + \boldsymbol{\Lambda}\frac{\partial(\boldsymbol{Lq})}{\partial x} = 0 , \tag{2.13}$$

which shows that, in characteristic space, each equation becomes decoupled from the rest, indicating that component $\mathscr{q}_i \in \mathbb{R}$ of the vector of *characteristic variables*,

$$\boldsymbol{\mathscr{q}}(t,x) := \boldsymbol{Lq}(t,x) , \tag{2.14}$$

is the invariant associated with the λ_i-field. In nonlinear situations, it is useful to define analogous *local characteristic variables*, obtained as in (2.14) but using a linearized eigenvector matrix (i.e. evaluated at some reference state) instead. Also noteworthy are the notions of *genuinely nonlinear* and *linearly degenerate* λ-fields, the latter being those for which:

$$\frac{\partial \lambda_i(\boldsymbol{q})}{\partial \boldsymbol{q}} \cdot \boldsymbol{r}_i(\boldsymbol{q}) = 0, \quad \boldsymbol{q} \in \mathbb{R}^{\mathrm{I}} . \tag{2.15}$$

The concepts laid out in this section play a role in shock capturing via solution limiting (§8), in addition to setting the groundwork for solving the Riemann problem (§2.4), which is, in turn, a fundamental building block of all discontinuous discretizations considered for review in this part of the report. For further details, the reader is referred to [50, §3], [114, §2.3.1, §2.4.3] and [82, §11.2, §13.3].

[4] Dimensional analysis of (2.13) readily reveals that λ_i has dimensions of length over time.

2.3. Discontinuous solutions

The hyperbolic nature of the systems of equations under consideration allows initially discontinuous[5] signals to remain so indefinitely. This is not the case in an elliptic or parabolic problem, where any discontinuity will be smeared out by the *physical diffusion* present in the system.

Furthermore, in the general case, even smooth initial conditions can evolve into discontinuous solutions over time. The qualitative explanation of this phenomena can be found in the λ-field structure of the solution: assume that two distinct characteristic lines intersect at some particular point of space-time; the solution at that point must still satisfy (2.11), but for two different initial values. The only possibility (in a genuinely nonlinear context) is for the solution to be multi-valued—hence, discontinuous. An example of this effect is the so-called *N-wave decay* that occurs for the the inviscid Burgers equation [82, §11.15].

The inviscid flow equations possess no mechanism to prevent the presence of arbitrarily large solution gradients, hence their solutions can, in principle, be truly discontinuous. When viscosity is taken into account, evidence[6] suggests that dissipative effects eventually become dominant, only allowing large (but finite) changes in flow quantities (e.g. density, pressure, temperature) across transition layers of small, but finite, thickness. Admittedly, these are not actual discontinuities; but considering them as such is often a reasonable approximation [114, p. 71]. Hence the treatment of both viscous and inviscid flows is typically the same in regards to discontinuity regularization strategies.

In CFD, the presence of (quasi-)discontinuous features (e.g. shock waves) is problematic. In a numerical simulation the number of degrees of freedom available to represent the solution is limited, making it unfeasible to resolve the flow across very sharp features such as shock waves (in most cases). Insufficient spatial resolution can lead to the physical dissipation present in the model, if any, being incompatible in terms of an energy balance with approximately accurate solution values. Unless there is some numerical (as opposed to physical) dissipative mechanism, the system may stabilize—if it does at all—at a highly oscillatory, erroneous state. In simpler terms: *the adequate equations, solved with insufficient resolution, result in an entirely inaccurate solution, and may even cause instability*. These nonphysical disturbances in the numerical solution are referred to in the literature as *spurious oscillations* or *wiggles*, and their appearance is known as the *Gibbs phenomenon* [47, §5.6].

Effectively coping with this issue is crucial for high-speed flow applications. Still, the lack of an entirely satisfactory approach (particularly so in multiple dimensions) has led some experts to argue that “we still do not really know how to capture shocks” [105]. *Shock capturing* consists in a controlled and localized increase of viscosity in the neighborhood of a shock (or discontinuity in general) so that the computed solution is smooth enough to be resolved discretely. Chapter 8 details various shock capturing schemes studied in this work, all of them designed for high-order methods. For a general overview of alternatives, see [99].

2.4. Riemann problem

Consider an infinite domain, $x \in [-\infty, +\infty]$. When the general hyperbolic conservation law (2.1) is solved for the following piecewise constant initial condition at $t = t_0$, we obtain a particularly interesting initial-value problem known as the *Riemann problem*:

$$\boldsymbol{q}^0(x) = \begin{cases} \boldsymbol{q}^L & x \leq x_0 \\ \boldsymbol{q}^R & x \geq x_0 \end{cases} . \tag{2.16}$$

Its solution will involve the transport of the initial data along λ-fields. Each of these characteristics can accommodate a jump discontinuity, thus the solution at some future instant $t_1 > t_0$ may consist (if there are no repeated eigenvalues) of a set of $\mathrm{I} + 1$ piece-wise constant states: $\{\boldsymbol{q}^1, \ldots, \boldsymbol{q}^{\mathrm{I}+1}\}$. Alternatively, a characteristic may split into an expansion fan, joining its two adjacent states smoothly via an infinite number of intermediate states. Additional details on the Riemann problem for each of the conservation laws studied in this work are included in the remaining sections of this chapter.

This model problem plays the key role of *coupling adjacent elements* in compact discontinuous methods, both in Godunov’s method and other finite volumes, and also in the high-order finite element schemes considered in this work. All these methods use the solution of the Riemann problem to

[5] In this report, C^k is the set of functions with continuous k-th derivative. Discontinuous functions belong to C^{-1} by convention.

[6] In [60], shocks are resolved (and, hence, the Gibbs phenomenon is not allowed to occur) for moderate Reynolds numbers with a non-dissipative 2nd order finite volume scheme.

evaluate a so-called *Riemann flux*,

$$\check{\boldsymbol{f}}\left(\boldsymbol{q}^L, \boldsymbol{q}^R\right) \coloneqq \boldsymbol{f}\left(\check{\boldsymbol{q}}\right) \tag{2.17}$$

(where

$$\check{\boldsymbol{q}} \coloneqq \boldsymbol{q}(t_1, x_0) \tag{2.18}$$

is the Riemann state), which is the flux computed from the solution of the Riemann problem at the location of the interface, some small time after the initial instant (see figure 2.1). In this way, solving the Riemann problem determines how to compute the flux at element interfaces (see §3).

For coupling purposes, computing the exact solution is not desirable, as it can be a computationally intensive process for nonlinear conservation laws; moreover, only one of the solution states is of interest In practice, an *approximate Riemann solver* can be employed to directly obtain $\check{\boldsymbol{f}}$ relatively cheaply.

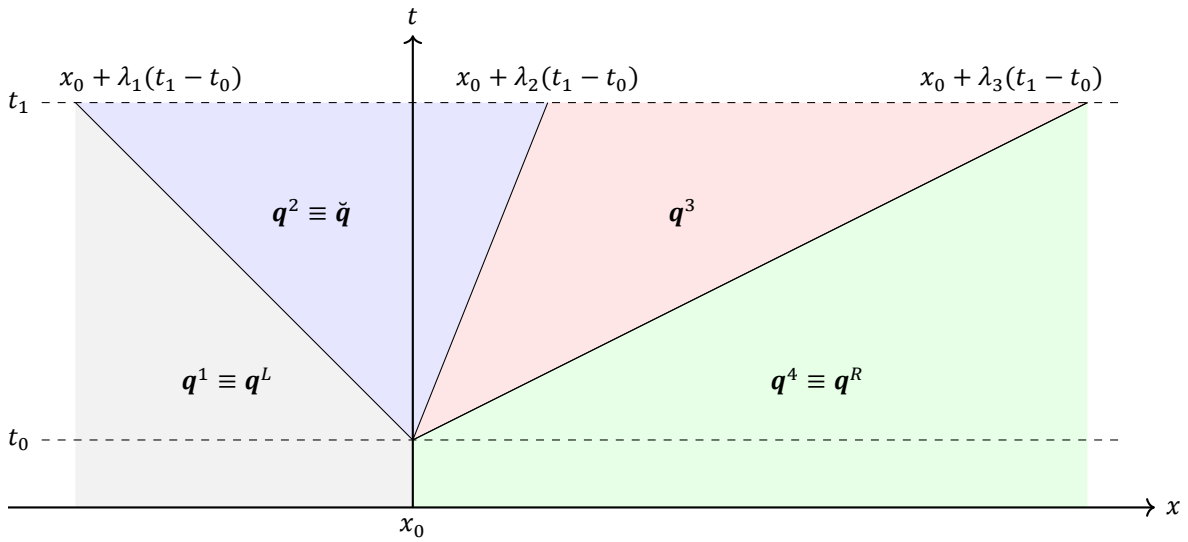


Figure 2.1: Solution to the Riemann problem in space-time. Depicted is a system with 3 *distinct* and *constant* eigenvalues, with $\boldsymbol{q}(t, x) = \boldsymbol{q}^0(x)$ for $t < t_0$. In the situation shown: $\check{\boldsymbol{q}} = \boldsymbol{q}^2$.

2.5. Advection

In the remainder of this chapter, four particular examples of hyperbolic conservation laws are given. The objective is to address in more detail some of the mathematical aspects that play a role in the numerical methods formulated in subsequent chapters, namely: their eigendecomposition and the solution of their Riemann problem in terms of a Riemann flux. All results reported in §II correspond to these PDEs.

The simplest hyperbolic conservation law is obtained when $\mathrm{I} = 1$ and $f(q) = aq$; a is a non-zero constant. Equation (2.1) becomes *scalar*, *linear* and of *constant coefficients*:

$$\frac{\partial q}{\partial t} + a\frac{\partial q}{\partial x} = 0\,. \tag{2.19}$$

This problem is useful to assess the behavior of errors in a linear setting, in particular dispersion, diffusion and stability (see appendix A).

2.5.1. Eigendecomposition

In this scalar case:

$$\boldsymbol{\Lambda} \equiv a\,, \qquad \boldsymbol{R} \equiv 1\,, \qquad \boldsymbol{L} \equiv 1\,, \qquad \boldsymbol{q}_\star \equiv q\,, \tag{2.20}$$

i.e. eigenvalues and eigenvectors are trivial; characteristic and conserved variables are one and the same.

2.5.2. Riemann flux

Given that the PDE is scalar, a single λ-field will be present in the solution. The additional restriction of the Jacobian being constant means that said characteristic curve is a straight line of slope a; the solution of the Riemann problem will consist on the two initial states, separated by a contact discontinuity traveling in space-time at a rate a, both of which remain unchanged. Consequently, the Riemann flux is:

$$\check{f}\left(q^L, q^R\right) = \begin{cases} aq^L & \text{if } a > 0 \\ aq^R & \text{if } a < 0 \end{cases} \quad \Leftrightarrow \quad \check{f}\left(q^L, q^R\right) = \frac{a}{2}\left(q^L + q^R\right) + \frac{|a|}{2}\left(q^L - q^R\right). \tag{2.21}$$

Note that this corresponds to selecting the state on the *upwind* side (i.e. the side opposite to the propagation of information, or "from where the wind blows").

2.6. Wave equation

The wave equation is usually thought of as a *scalar second-order PDE*:

$$\frac{\partial^2 q}{\partial t^2} + c^2 \frac{\partial^2 q}{\partial x^2} = 0. \tag{2.22}$$

Nevertheless, it can be rewritten as the following system of *two* first-order PDEs [82, §2.9.1]:

$$\frac{\partial}{\partial t}\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} + \begin{bmatrix} 0 & c^2 \\ 1 & 0 \end{bmatrix} \frac{\partial}{\partial x}\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = 0. \tag{2.23}$$

This problem will be used to study the generalization of the different methods to vector equations (i.e. $\mathrm{I} > 1$, in this case $\mathrm{I} = 2$). In it, each equation is still linear and the Jacobian matrix is still constant.

2.6.1. Eigendecomposition

The Jacobian matrix in (2.23) has two real and distinct eigenvalues. Its diagonalization and associated characteristic variables are not trivial:

$$\boldsymbol{\Lambda} = \begin{bmatrix} c & 0 \\ 0 & -c \end{bmatrix}, \qquad \boldsymbol{R} = \begin{bmatrix} c & -c \\ 1 & 1 \end{bmatrix}, \qquad \boldsymbol{L} = \frac{1}{2c}\begin{bmatrix} 1 & c \\ -1 & c \end{bmatrix}, \qquad \boldsymbol{q} = \frac{1}{2c}\begin{bmatrix} q_1 + cq_2 \\ cq_2 - q_1 \end{bmatrix}. \tag{2.24}$$

2.6.2. Riemann flux

Once the system 2.23 has been diagonalized, it becomes equivalent to two decoupled advection equations. Thanks to the linear nature of the same, the exact solution to its Riemann problem can then be conveniently obtained by linear superposition of each of these equations' solutions. Assuming that the left and right states are known, the Riemann flux is given by:

$$\check{f}\left(\boldsymbol{q}^L, \boldsymbol{q}^R\right) = \frac{1}{2}\begin{bmatrix} |c| & c^2 \\ 1 & |c| \end{bmatrix}\boldsymbol{q}^L + \frac{1}{2}\begin{bmatrix} -|c| & c^2 \\ 1 & -|c| \end{bmatrix}\boldsymbol{q}^R. \tag{2.25}$$

2.6.3. Equivalent systems

The Euler equations (§2.8) can be linearized—see [82, §2.7]—to give the following system of 2 PDEs which models the propagation of weak pressure and velocity fluctuations in a fluid flow:

$$\frac{\partial}{\partial t}\begin{bmatrix} p \\ u \end{bmatrix} + \begin{bmatrix} u_0 & K_0 \\ 1/\rho_0 & u_0 \end{bmatrix} \frac{\partial}{\partial x}\begin{bmatrix} p \\ u \end{bmatrix} = 0. \tag{2.26}$$

Here, p and u are pressure and velocity perturbations around a reference state p_0, u_0. K_0 is the bulk modulus of compressibility of the fluid. If we let $u_0 = 0$, (2.26) becomes a model for the propagation of sound waves within a fluid at rest. Its dimensionless form turns out to be identical to that of (2.23).

This equivalence holds true also for the equations of solid mechanics, in the elastic regime, for a one-dimensional solid [82, §2.12]:

$$\frac{\partial}{\partial t}\begin{bmatrix} \sigma_{11} \\ u \end{bmatrix} + \begin{bmatrix} 0 & -(\lambda + 2\mu) \\ -1/\rho & 0 \end{bmatrix} \frac{\partial}{\partial x}\begin{bmatrix} \sigma_{11} \\ u \end{bmatrix} = 0, \tag{2.27a}$$

$$\frac{\partial}{\partial t}\begin{bmatrix}\sigma_{12}\\ v\end{bmatrix}+\begin{bmatrix}0 & -\mu\\ -1/\rho & 0\end{bmatrix}\frac{\partial}{\partial x}\begin{bmatrix}\sigma_{12}\\ v\end{bmatrix}=0\,, \tag{2.27b}$$

where $\boldsymbol{\sigma}\in\mathbb{R}^{1\times 2\times 1}$ is the stress tensor and u, v are velocity components normal and tangential to the length of the solid (respectively); λ and μ are, here, the Lamé parameters characterizing the material.

The same applies for Maxwell's equations in the case of a planar electromagnetic wave [82, §2.14]:

$$\frac{\partial}{\partial t}\begin{bmatrix}E_2\\ B_3\end{bmatrix}+\begin{bmatrix}0 & \frac{1}{\varepsilon\mu}\\ 1 & 0\end{bmatrix}\frac{\partial}{\partial x}\begin{bmatrix}E_2\\ B_3\end{bmatrix}=0\,, \tag{2.28}$$

with $\vec{E},\vec{B}\in\mathbb{R}^3$ being the electric and magnetic field intensities, ε the permittivity of the medium and μ its magnetic permeability.

All of these systems model essentially the same phenomenon: a wave-like propagation of the solution, at a rate associated with a constant $c\equiv|\lambda|$, be it the speed of sound in a fluid, normal or tangential wave propagation speeds in a solid, or the speed of light in some medium.

2.7. Inviscid Burgers equation

All PDEs until this point have been examples of *linear* hyperbolic conservation laws. The simplest nonlinear case is constructed from (2.19), generalizing its Jacobian to make it a function of the solution. For convenience, we may choose this dependency to be a one-to-one equivalence: $a(u)\coloneqq u$. Thus we obtain the (inviscid) *Burgers equation*, a scalar nonlinear hyperbolic conservation law which models a convection process in which the advection speed depends on the advected quantity itself:

$$\frac{\partial u}{\partial t}+\frac{\partial}{\partial x}\left(\frac{u^2}{2}\right)=0\,. \tag{2.29}$$

For this conservation law, neglecting any influence from boundary conditions and assuming $u^0(x)\in C^0$, it can be shown that $u(t,x)$ becomes discontinuous at an instant t_{shock}, given by [82, p. 224, exercise 11.1]

$$t_{\text{shock}}=-\left(\min_x\left\{\frac{\mathrm{d}u^0}{\mathrm{d}x}\right\}\right)^{-1}\,, \tag{2.30}$$

corresponding to the earliest moment in which any two characteristic fields intersect each other—if $t_{\text{shock}}<0$, characteristics never cross. Under these conditions, for $t<t_{\text{shock}}$, the exact solution of (2.29) is:

$$u(t,x)=u^0\left(x-u(t,x)t\right)\,. \tag{2.31}$$

When necessary in part II of this report, equation (2.31) is evaluated recursively: stopping when the maximum norm of the absolute error between the result of two successive estimates is $<1\times10^{-12}$, starting from $u(t,x)=u^0\left(x-u^0(x)t\right)$.

2.7.1. Eigendecomposition

Diagonalization of (2.29)'s Jacobian is analogous to (2.20):

$$\boldsymbol{\Lambda}\equiv u\,,\qquad \boldsymbol{R}\equiv 1\,,\qquad \boldsymbol{L}\equiv 1\,. \tag{2.32}$$

The only difference in this case being that the single eigenvalue depends on the solution.

2.7.2. Riemann flux

Equation (2.29) having a non-constant characteristic speed complicates the exact solution of its Riemann problem quite substantially. We are interested in a particular state $\check{u}$ (see §2.4) such that:

$$\check{f}\left(u^L,u^R\right)=f\left(\check{u}\right)=\frac{1}{2}\check{u}^2\,. \tag{2.33}$$

However, it is not trivial to decide which side is the upwind one, in general. It will depend on the specific combination of sign and magnitude of the eigenvalues seen from each side of x_0. Moreover, there is one situation in which $\check{u}$ is neither u^L nor u^R. Four cases are possible in total, summarized in table 2.1. They are explained next.

Case	u^L	u^R	$\breve{u}$
1	> 0	> 0	u^L
2	< 0	< 0	u^R
3	> 0	< 0	u^L if $u^L \geq \left\lvert u^R \right\rvert$, u^R if $u^L \leq \left\lvert u^R \right\rvert$
4	< 0	> 0	0

Table 2.1: Summary of initial condition combinations in the Riemann problem for Burgers equation.

Case 1: right-going shocks
Both left and right state eigenvalues have the same, positive sign. Their associated λ-fields are shock waves, both traveling in the positive direction of the x axis; therefore, the upwind state is the left one:

$$\breve{u} \equiv u^L \implies \breve{f}\left(u^L, u^R\right) = f(u^L)\,. \tag{2.34}$$

Case 2: left-going shocks
Reciprocal of the previous case, the two characteristics are traveling towards decreasing x-axis values:

$$\breve{u} \equiv u^R \implies \breve{f}\left(u^L, u^R\right) = f(u^R)\,. \tag{2.35}$$

Case 3: centered shock-shock interaction
In this situation the right-going characteristics from the left state are intersecting the left-going ones coming from the right. This will result in a shock wave that, at the instant t_1, moves with speed [82, §11.8]:

$$\overline{u} = \frac{u^L + u^R}{2}\,. \tag{2.36}$$

If the right-going characteristic dominates, $\overline{u} > 0$; the shock resulting from the interaction will be moving to the right. Otherwise, if $\overline{u} < 0$, the combined show wave travels to the left. A stationary shock can occur, if $\left\lvert u^L \right\rvert = \left\lvert u^R \right\rvert \iff u^L = -u^R$; this case is trivial, since $f(u) \equiv f(-u)$. In conclusion:

$$\breve{f}\left(u^L, u^R\right) = \begin{cases} f(u^L), & u^L + u^R \geq 0 \\ f(u^R), & u^L + u^R < 0 \end{cases}\,. \tag{2.37}$$

Case 4: centered expansion
Two characteristic fields are adjacent but moving away from each other at (t_0, x_0). Let us assume a linear expansion fan profile for $t_1 \approx t_0$. At t_1, the left crest will have moved (to the left) to $x_0 + u^L \Delta t$, while the right one will be at $x_0 + u^R \Delta t$ (right of its initial position). The left and right states, initially adjacent, are now joined by a straight line, $y(x) = c_1\,(x - x_0) + c_0$; therefore:

$$\begin{cases} u^L &= c_1 u^L \Delta t + c_0 \\ u^R &= c_1 u^R \Delta t + c_0 \end{cases} \implies \frac{c_0}{u^R} - \frac{c_0}{u^L} = 0 \implies y\,(x_0) = 0 \implies \breve{u} = 0\,.$$

Which means that the expansion fan will be centered around zero at t_1 regardless of the specific values of u^L and u^R, i.e. :

$$\breve{f}\left(u^L, u^R\right) = f\,(0) = 0\,. \tag{2.38}$$

2.8. Euler equations

The Euler equations model the behavior of a compressible, inviscid fluid. They are the most general case considered in this work: a system of nonlinear PDEs,

$$\frac{\partial}{\partial t}\begin{bmatrix} \rho \\ \rho u \\ \rho E \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} \rho u \\ \rho u^2 + p \\ u\,(\rho E + p) \end{bmatrix} = 0\,. \tag{2.39}$$

The conserved variables are: *mass*, *momentum* and *total energy* (each of them per unit volume), and each has its own corresponding flux vector component. The flow velocity is u.

An additional equation of state is needed in order to close (2.39). Under the assumption of a *calorically perfect gas*, such a relation is given by

$$\frac{p}{\gamma-1} = \rho e = \rho E - \frac{1}{2}\rho u^2 \,, \tag{2.40}$$

where p is the pressure, e is the *internal energy* (per unit mass) and γ is the *diatomic gas constant*.

Several additional relations and quantities can be derived from the current set based on thermodynamic considerations [114, section 1.2]. Most relevant among these are the *total enthalpy* (per unit volume),

$$\rho H = \rho E + p \,, \tag{2.41}$$

and the *speed of sound*,

$$c = \sqrt{\frac{\gamma p}{\rho}} = \sqrt{(\gamma-1)\left(H - \frac{1}{2}u^2\right)} \,. \tag{2.42}$$

In addition to the vector of conserved variables, it is useful for the Euler equations to define two additional state vectors, *primitive* and *Roe* variables ($\boldsymbol{v}$ and $\boldsymbol{w}$, respectively):

$$\boldsymbol{v} \coloneqq \begin{bmatrix} \rho & u & p \end{bmatrix}^\mathsf{T} \,, \qquad\qquad \boldsymbol{w} \coloneqq \begin{bmatrix} \sqrt{\rho} & \sqrt{\rho}u & \sqrt{\rho}H \end{bmatrix}^\mathsf{T} \,. \tag{2.43}$$

Let the transformation from conserved to primitive variables be designated by the nonlinear operator $\boldsymbol{T} \colon \mathbb{R}^3 \to \mathbb{R}^{3\times 3}$, defined such that $\boldsymbol{v} = \boldsymbol{T}\boldsymbol{q}$:

$$\boldsymbol{T}(\boldsymbol{q}) \coloneqq \begin{bmatrix} 1 & 0 & 0 \\ 0 & {}^1\!/_{q_1} & 0 \\ 0 & \frac{1-\gamma}{2}\frac{q_2}{q_1} & \gamma-1 \end{bmatrix} . \tag{2.44}$$

And let a so-called *Roe average* between two vectors of Roe variables be:

$$\widetilde{\boldsymbol{w}}\left(\boldsymbol{w}^L, \boldsymbol{w}^R\right) \coloneqq \begin{bmatrix} w_1^L w_1^R & \frac{w_2^L + w_2^R}{w_1^L + w_1^R} & \frac{w_3^L + w_3^R}{w_1^L + w_1^R} \end{bmatrix}^\mathsf{T} . \tag{2.45}$$

2.8.1. Eigendecomposition

In conserved variables, the Jacobian matrix of 2.39 and its eigendecompostion are [82, §14.8]:

$$\boldsymbol{A}(\boldsymbol{q}) = \begin{bmatrix} 0 & 1 & 0 \\ \frac{\gamma-3}{2}u^2 & (3-\gamma)\,u & \gamma-1 \\ \left(\frac{\gamma-1}{2}u^2 - H\right)u & H - (\gamma-1)\,u^2 & \gamma u \end{bmatrix} , \tag{2.46a}$$

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = \begin{bmatrix} u-c & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & u+c \end{bmatrix} , \tag{2.46b}$$

$$\boldsymbol{R}(\boldsymbol{q}) = \begin{bmatrix} 1 & 1 & 1 \\ u-c & u & u+c \\ H-uc & \frac{1}{2}u^2 & H+uc \end{bmatrix} , \tag{2.46c}$$

$$\boldsymbol{L}(\boldsymbol{q}) = \frac{\gamma-1}{2c^2}\begin{bmatrix} H + \frac{c}{\gamma-1}(u-c) & -\left(u + \frac{c}{\gamma-1}\right) & 1 \\ -2H + \frac{4}{\gamma-1}c^2 & 2u & -2 \\ H + \frac{c}{\gamma-1}(u+c) & -u + \frac{c}{\gamma-1} & 1 \end{bmatrix} . \tag{2.46d}$$

For primitive variables, however, (2.46a) and (2.46c) are simplified substantially (the eigenvalues, of course, remain the same) [82, 14.7]:

$$\boldsymbol{A}(\boldsymbol{v}) = \begin{bmatrix} u & \rho & 0 \\ 0 & u & {}^1\!/_{\rho} \\ 0 & \gamma p & u \end{bmatrix} , \qquad\qquad \boldsymbol{R}(\boldsymbol{v}) = \begin{bmatrix} -{}^\rho\!/_c & 1 & {}^\rho\!/_c \\ 1 & 0 & u+c \\ -\rho c & 0 & \rho c \end{bmatrix} . \tag{2.47}$$

Recalling (2.15), it turns out that

$$\frac{\partial \lambda_1(\boldsymbol{v})}{\partial \boldsymbol{v}} \cdot \boldsymbol{r}_1(\boldsymbol{v}) = \begin{bmatrix} \frac{c}{2\rho} & 1 & -\frac{c}{2\rho} \end{bmatrix} \begin{bmatrix} -\rho/c \\ 1 \\ -\rho c \end{bmatrix} = \frac{\gamma+1}{2}, \tag{2.48a}$$

$$\frac{\partial \lambda_2(\boldsymbol{v})}{\partial \boldsymbol{v}} \cdot \boldsymbol{r}_2(\boldsymbol{v}) = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \equiv 0, \tag{2.48b}$$

$$\frac{\partial \lambda_3(\boldsymbol{v})}{\partial \boldsymbol{v}} \cdot \boldsymbol{r}_3(\boldsymbol{v}) = \begin{bmatrix} -\frac{c}{2\rho} & 1 & \frac{c}{2\rho} \end{bmatrix} \begin{bmatrix} \rho/c \\ 1 \\ \rho c \end{bmatrix} = \frac{\gamma+1}{2}, \tag{2.48c}$$

i.e. the λ_2-field is always linearly degenerate in the Euler equations [82, §14.7]. This is the reason why a discontinuity associated with it is not referred to as a shock but a *contact discontinuity*. The other two characteristics can either be a shock or a rarefaction, both genuinely nonlinear since $\gamma > 0$.

2.8.2. Riemann solvers

In this section some of the many different approaches to solving the Riemann problem for the Euler equations—those which have been studied and tested during the course of this work—are simply listed; a thorough description of each can be found in their respective chapters of [114].

Exact Riemann solver

Similarly to the case of Burgers equation, this approach consists on deducing future states at a time in the very near future based on the characteristic speeds of the system. Toro [114, §4] gives a very detailed explanation, including a Fortran implementation, of a simple yet effective approach based on using pressure to determine which of the four possible combinations involving left shock, left rarefaction, right shock and right rarefaction is taking place. Knowing that, it is possible to deduce the state at the interface in a similar way as it was shown for the Burger's case and eventually use it to compute the Riemann flux.

Single wave-speed estimate

The simplest kind of Riemann solvers use a single estimated wave-speed to compute an approximated Riemann flux. Note that the diagonalization of (2.46a) revealed that there are in fact 3 distinct eigenvalues in the 1D situation; these methods sacrifice accuracy in favour of efficiency, and are the most diffusive among those reviewed. One of the most popular among these is the **Local Lax-Friedrichs** or **Rusanov** solver [114, p. 329].

Two wave-speed estimates

Quite less diffusive are the methods that use two characteristic speed estimates to solve the problem approximately. Consequently, they are slightly more computationally-intensive than those of the previous class. Most popular in this category is the **HLL** solver [114, §10.3]. An alternative estimation of the two wave speeds leads to a variant of it known as **HLLE** [82, §15.3.7], which is reported to have positivity preserving properties [33, 34].

Three wave-speed estimates

Adding a third characteristic speed makes the Riemann solver significantly more accurate. A successful example of one such schemes is **HLLC**, which adds an additional characteristic speed estimate to HLL [114, §10.4]. Numerical tests reported in [114] suggest that this method is as accurate or more than HLL (particularly at contact discontinuities) while being only slightly more expensive than the methods using only two wave-speed estimates.

Last, and arguably the most popular of them all, **Roe's Riemann solver** [114, §11] also uses three wave-speeds. Its estimation employs eigendecomposition of the Jacobian matrix, which leads to a more accurate flux at the price of a more costly scheme. In practice, however, Roe's solver is known to be more prone to failure e.g. near-vacuum situations than alternatives. Furthermore, it is the only

one among those considered here that requires an *entropy fix* to produce a physical solution at sonic rarefactions [114, §11.4]. In this work only one of such fixes has been studied, that of **Harten–Hyman** [114, §11.4.2].

3

Compact Discontinuous High-Order Discretization

Discretization, broadly speaking, consists on replacing some *continuous* system—not amenable to representation on a digital computer—with some *finite number* of *entities* of some kind, eventually leading to a finite set of *degrees of freedom*[1] the behavior of which is modeled instead.

This chapter is meant as an introduction to the discretization framework targeted by the present thesis. It describes, in a general setting, all aspects that the three particular methods selected as research objects share—leaving the specifics of each to its own dedicated chapter (§§ 4–6). The derivation is within the framework of the method of lines; details about the treatment of the time operator are postponed until §7. The relationship that exists between compact discontinuous high-order methods and the final volume method is explored in §3.4. Sections 3.5 and 3.6, specifically address the enforcement of initial and boundary conditions in a unified manner for all the methods considered. There is a large amount of literature covering these topics; some suggestions are: Blazek [14], Brenner and Scott [16], Hartmann and Leicht [46], Hesthaven and Warburton [47], Hirsch [50], Kopriva [66].

3.1. Domain discretization

The result of spatial discretization is a *mesh*, $\mathcal{T}^h \coloneqq \{\Omega_k\}_{k=1}^{\mathrm{K}}$. It is obtained by dividing Ω into K non-overlapping sub-domains, called *elements* (also cells or patches, depending on the context), such that:

$$\Omega = \bigcup_{k=1}^{\mathrm{K}} \Omega_k \,, \qquad \bigcap_{k=1}^{\mathrm{K}} \Omega_k = \emptyset \,, \qquad \mathrm{K} \geq 1 \,. \tag{3.1}$$

In the most general case, each Ω_k is a 3D shape (not necessarily a polyhedron) composed of faces, edges and vertices. For the one-dimensional case within scope, the situation is greatly simplified: each element is a line segment, with two end-points that are the equivalent of vertices, edges and faces—all collapsed into a single entity (see figure 3.1). Consequently, only one type of connectivity exists: every line is delimited by points, and every one of such points is connected to two adjacent line segments (which share it). In short, to generate a mesh in 1D we simply need to specify $x_1 < x_2 < \ldots < x_{\mathrm{K}+1}$, a partitioning of x into K intervals.

Each element and its left and right edges are defined as:

$$\Omega_k \coloneqq [x_k, x_{k+1}] \,, \qquad \partial\Omega_k^L \coloneqq x_k \,, \qquad \partial\Omega_k^R \coloneqq x_{k+1} \,. \tag{3.2}$$

The "volume" of Ω_k (i.e. it's length) and the position of its centroid (i.e. midpoint) are:

$$\Delta x_k \coloneqq x_{k+1} - x_k \,, \qquad \overline{x}_k \coloneqq \frac{x_k + x_{k+1}}{2} \,. \tag{3.3}$$

[1] The term "degree of freedom" is used in FE to refer to each of the unknowns that the discrete system is to be solved for. This generalizes the various particular choices used in each method. For example: in FD, the degrees of freedom are solution values at grid points; in cell-centered FV, the unknowns are cell-averaged solution values, etc.

Among the set of all element boundaries $\{x_1, \dots, x_{\mathrm{K}+1}\}$, let us distinguish between *exterior* ones (i.e. the boundaries of the domain) $\{x_1, x_{\mathrm{K}+1}\}$ and *interior* ones (i.e. element interfaces) $\{x_2, \dots, x_{\mathrm{K}}\}$.

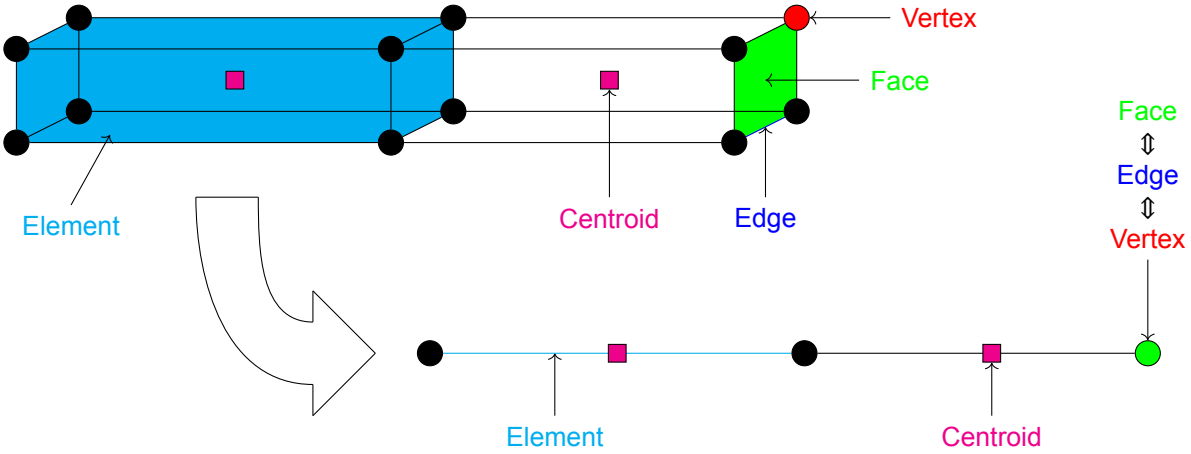


Figure 3.1: Analogies between three-dimensional (hexahedral) and one-dimensional meshes.

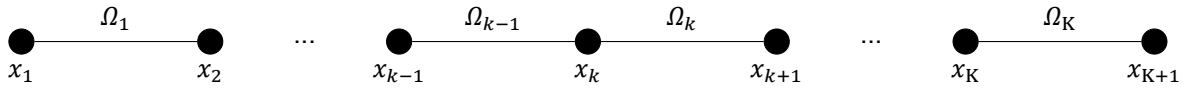


Figure 3.2: Indexing convention for the one-dimensional meshes used in this report.

3.1.1. Reference element

Let $\xi \in \widetilde{\Omega} \coloneqq [-1, 1]$ define a *reference element* and its 1D coordinate system. Any point $x \in \mathcal{T}^h$ can be transformed to reference element coordinates via the invertible *affine mapping* $\mathcal{X}_k : \Omega_k \to \widetilde{\Omega}$, defined as [66, p. 298]:

$$\mathcal{X}_k(\xi) \coloneqq \overline{x}_k + \frac{\Delta x_k}{2}\xi\,, \qquad \mathcal{X}_k^{-1}(x) = \frac{2}{\Delta x_k}\left(x - \overline{x}_k\right)\,. \tag{3.4}$$

Its Jacobian is simply a scalar constant:

$$\frac{\mathrm{d}\mathcal{X}_k}{\mathrm{d}\xi} \coloneqq \left(\frac{\mathrm{d}x}{\mathrm{d}\xi}\right)\Big|_{\Omega_k} = \frac{\Delta x_k}{2}\,. \tag{3.5}$$

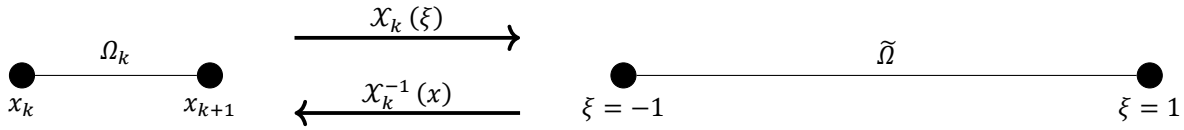


Figure 3.3: Mapping between physical element Ω_k, and reference element $\widetilde{\Omega}$.

3.2. Solution discretization

Discretization of the solution consists on replacing its exact counterpart with an *approximate solution* $\boldsymbol{q}^h(t,x) \simeq \boldsymbol{q}(t,x)$. To that aim, high-order finite element methods introduce $S^h \in L^2(\Omega)$, a finite-dimensional *trial* function space in which to "look for" $\boldsymbol{q}^h$. Discontinuous methods (unlike standard finite elements) do not enforce continuity of the approximate solution across element boundaries. Instead:

$$\boldsymbol{q}^h(t,x) \coloneqq \bigoplus_{k=1}^{\mathrm{K}} \boldsymbol{q}_k^h(t,x)\,, \qquad q_{ik}^h(t,x) \in C^0\,; \tag{3.6}$$

i.e. the global approximated solution is constructed by "stitching together"[2] several element-wise continuous solutions, rendering the approximate solution and flux components twice-valued at every interior edge.

3.2.1. Reference element space

In the most general situation, each component of $\boldsymbol{q}_k^h$ exists in its element's own function space. Even in that case, it is convenient to work in reference coordinates (see §3.1.1), such that:

$$\widetilde{\boldsymbol{q}}_k^h(t,\xi) \coloneqq \boldsymbol{q}_k^h\left(t, \mathcal{X}_k(\xi)\right) , \qquad \widetilde{q}_{ik}^h \in S_k^h\left(\widetilde{\Omega}\right) . \tag{3.7}$$

Given that $S_k^h\left(\widetilde{\Omega}\right)$ is independent on the element's geometry, is is possible for multiple elements to share it; e.g. a discretization where all elements use the same number of degrees of freedom (and the same spatial discretization scheme) only uses a single unique reference element space. Without loss of generality, I assume that each element possesses its own approximate solution space.

3.2.2. Trial basis functions

Each reference element space, $S_k^h\left(\widetilde{\Omega}\right)$, needs to be specified via a basis. Given that we are dealing with a function space, its basis' components are functions—hence their name. Each discretization scheme uses a particular type of basis; more information about the methods reviewed can be found in their respective chapters of this report. Given a generic set $\{\phi_{1k}(\xi), \ldots, \phi_{\mathrm{J}k}(\xi)\}$ of basis functions, the reference element space of J dimensions[3] is constructed as:

$$S_k^h\left(\widetilde{\Omega}\right) \coloneqq \operatorname{span}\{\phi_{jk}(\xi)\}_{j=1}^{\mathrm{J}} . \tag{3.8}$$

For convenience, let us also define the following vector basis function $\boldsymbol{\phi}_k : \mathbb{R} \to \mathbb{R}^{\mathrm{J}}$:

$$\boldsymbol{\phi}_k(\xi) \coloneqq \begin{bmatrix} \phi_{1k}(\xi) & \phi_{2k}(\xi) & \ldots & \phi_{\mathrm{J}k}(\xi) \end{bmatrix}^{\mathrm{T}} . \tag{3.9}$$

3.2.3. Degrees of freedom

The approximate solution vector on Ω_k is encoded as a linear combination of the basis functions of S_k^h; I shall refer to the coefficients of such a series expansion as the *degrees of freedom* of Ω_k. It is convenient to group all degrees of freedom of Ω_k into a two-dimensional array $\hat{\boldsymbol{Q}}_k \in \mathbb{R}^{\mathrm{I}\times\mathrm{J}}$,

$$\hat{\boldsymbol{Q}}_k(t) \coloneqq \begin{bmatrix} \hat{q}_{11k}(t) & \hat{q}_{12k}(t) & \cdots & \hat{q}_{1\mathrm{J}k}(t) \\ \vdots & \vdots & & \vdots \\ \hat{q}_{\mathrm{I}1k}(t) & \hat{q}_{\mathrm{I}2k}(t) & \cdots & \hat{q}_{\mathrm{IJ}k}(t) \end{bmatrix} \equiv \begin{bmatrix} \hat{\boldsymbol{q}}_{1k}(t) & \hat{\boldsymbol{q}}_{2k}(t) & \cdots & \hat{\boldsymbol{q}}_{\mathrm{J}k}(t) \end{bmatrix} , \tag{3.10}$$

so that the approximate solution components within Ω_k are obtained (in reference coordinates) as:

$$\widetilde{\boldsymbol{q}}_k^h(t,\xi) = \hat{\boldsymbol{q}}_{1k}(t)\phi_1(\xi) + \hat{\boldsymbol{q}}_{2k}(t)\phi_2(\xi) + \ldots + \hat{\boldsymbol{q}}_{\mathrm{J}k}(t)\phi_{\mathrm{J}}(\xi) = \sum_{j=1}^{\mathrm{J}} \hat{\boldsymbol{q}}_{jk}(t)\phi_{jk}(\xi) \equiv \hat{\boldsymbol{Q}}_k(t)\boldsymbol{\phi}_k(\xi) . \tag{3.11}$$

These uniquely define the global approximate solution at any (t,x), with $\hat{q}_{ijk}(t) \in \mathbb{R}$ being a particular degree of freedom associated with equation $1 \leq i \leq \mathrm{I}$, basis function $1 \leq j \leq \mathrm{J}$ and element $1 \leq k \leq \mathrm{K}$.

3.2.4. Flux representation

For economy of notation, let us define an approximate version of the flux function as

$$\boldsymbol{f}^h(t,x) \simeq \boldsymbol{f}\left(\boldsymbol{q}(t,x)\right) , \tag{3.12}$$

with its element-local and reference-element counterparts, $\boldsymbol{f}_k^h(t,x)$ and $\widetilde{\boldsymbol{f}}_k^h(t,\xi)$, defined analogously to those of the approximate solution. For every flux function considered in this report, f_{ik}^h is continuous if so is q_{ik}^h. With nonlinear and/or nonconstant flux Jacobians, choosing a suitable representation for this

[2] More precisely, $\boldsymbol{q}^h$ is the *direct sum* of all element-wise solutions [47, §2.2.1].
[3] Even if omitted for clarity of notation, the length of each basis is not assumed unique over the various elements, i.e. $\mathrm{J} = \mathrm{J}(k)$.

approximate flux vector is a delicate issue. The problem arises from the fact that, even if $\widetilde{q}_{ik}^h \in S^h(\widetilde{\Omega})$, in general:

$$f_i\left(\widetilde{\boldsymbol{q}}_k^h(t,\xi)\right) \notin S^h(\widetilde{\Omega})\,. \tag{3.13}$$

For all methods considered in this report, the flux vector components are expanded into the very same S_k^h as the state vector ones[4], i.e. :

$$\widetilde{\boldsymbol{f}}_k^h(t,\xi) = \hat{\boldsymbol{F}}_k(t)\boldsymbol{\phi}_k(\xi) \equiv \begin{bmatrix} \hat{f}_{11k}(t) & \hat{f}_{12k}(t) & \cdots & \hat{f}_{1\mathrm{J}k}(t) \\ \vdots & \vdots & & \vdots \\ \hat{f}_{\mathrm{I}1k}(t) & \hat{f}_{\mathrm{I}2k}(t) & \cdots & \hat{f}_{\mathrm{I}\mathrm{J}k}(t) \end{bmatrix} \begin{bmatrix} \phi_{1k}(\xi) \\ \phi_{2k}(\xi) \\ \vdots \\ \phi_{\mathrm{J}k}(\xi) \end{bmatrix}; \tag{3.14}$$

and therefore, except for linear flux functions:

$$\widetilde{\boldsymbol{f}}_k^h(t,\xi) \neq \boldsymbol{f}\left(\widetilde{\boldsymbol{q}}_k^h(t,\xi)\right)\,. \tag{3.15}$$

This mismatch that, in general, occurs between the number of dimensions available in the approximate solution space and number of those that would actually be necessary to exactly resolve its flux introduces some error, in addition to the truncation one, if any, into the discretization. Sometimes broadly referred to as a “variational crime”, this error can be understood as a consequence of *aliasing* in nodal finite element and collocation spectral methods [47, §5.3] (see §4.2 for additional details). Moreover, although tempting, in general $\hat{\boldsymbol{f}}_{jk} \neq \boldsymbol{f}\left(\hat{\boldsymbol{q}}_{jk}\right)$. The evaluation of these flux coefficients depends on each method and its choice of basis, and is thus delegated to subsequent chapters.

3.3. Equation discretization

Sections 3.1 and 3.2 in combination provide a mechanism with which to represent an approximation to the exact solution of (2.1) in a discrete setting, one that allows high order of accuracy under suitably chosen basis functions. All that remains is a way to actually compute said approximate solution. This is achieved by transforming the mathematical model (equation 2.1 in this case) from a set of partial differential equations to a system of *algebraic* equations, in which the unknowns are then not the exact state variables defined for a continuum, but the degrees of freedom (3.10) of the discrete solution.

3.3.1. Spatial residuals (method of lines)

The method of lines refers to a particular treatment of the spatial and temporal operators of the general conservation law. In this approach, one aims to obtain a *semi-discrete* version of the system of PDEs first, in which the temporal derivative still appears as a continuous operator. This decouples the spatial and temporal discretization schemes from each other (see §7).

After (spatial) semi-discretization, (2.1) for each degree of freedom is expected to become

$$\frac{\mathrm{d}\hat{q}_{ijk}}{\mathrm{d}t} = \hat{r}_{ijk}(t)\,, \tag{3.16}$$

which is an ordinary differential equation (ODE), and can be solved as such. I will refer to the scalar $\hat{r}_{ijk} \in \mathbb{R}$ as an *expansion coefficient of the (spatial) residual* [129] associated with the i,j,k-th degree of freedom. By analogy with the approximate solution, the matrices of residual coefficients $\hat{\boldsymbol{\mathcal{R}}}_k$ define a discretization-dependent *residual function* $\widetilde{\boldsymbol{r}}_k(t,\xi) \in S_k^h\left(\widetilde{\Omega}\right)$, as follows:

$$\widetilde{\boldsymbol{r}}_k(t,\xi) \coloneqq \hat{\boldsymbol{\mathcal{R}}}(t)\boldsymbol{\phi}_k^{\mathrm{T}}(\xi)\,, \qquad \boldsymbol{r}_k(t,x) = \widetilde{\boldsymbol{r}}_k(t,\mathcal{X}_k(\xi))\,, \qquad \boldsymbol{r}(t,x) \coloneqq \bigoplus_{k=1}^{\mathrm{K}} \boldsymbol{r}_k(t,x)\,. \tag{3.17}$$

3.3.2. Weak formulation

All finite element methods, including those considered in this report, start from a weak (or variational) formulation of the continuous problem. For a discontinuous method, a convenient weak form of (2.1)

[4] This approach is borrowed from continuous finite element methods, where it is known as Fletcher’s group formulation [37].

is obtained by multiplying it first by an arbitrary (sufficiently smooth) test function $v \in V(\Omega)$, and then integrating by parts over $\Omega_k \in \mathcal{T}^h$:

$$\frac{\partial}{\partial t}\int_{\Omega_k} \boldsymbol{q} v \,\mathrm{d}x + \boldsymbol{f}\Big|_{\partial\Omega_k^R} v(x_{k+1}) - \boldsymbol{f}\Big|_{\partial\Omega_k^L} v(x_k) - \int_{\Omega_k} \boldsymbol{f}\frac{\mathrm{d}v}{\mathrm{d}x}\,\mathrm{d}x = 0\,. \tag{3.18}$$

The weak form of the general hyperbolic conservation law on an element can now be stated as: find $\boldsymbol{q}(t,x)$ such that (3.18) holds for *every* $v(x)$, for $k = 1, 2, \ldots, \mathrm{K}$. Similarly to the difference between the differential and the integral forms of the hyperbolic conservation law, this formulation of the problem requires fewer conditions on its solution. More importantly, though, is the fact that, at the same time (and unlike the integral form itself), it provides a way to introduce the high-order approximate solution from the previous section into the discrete conservation law.

In a discrete setting, it is not possible to enforce (3.18) for *all* test functions. Instead, the space of test functions is replaced by a finite-dimensional subset of it, $V^h(\Omega) \subset V(\Omega)$. The exact specification of this subspace depends on the discretization scheme; the most common approach consists on reusing the trial space as test function space[5], i.e.: $V^h = S^h$. With discontinuous methods, it is convenient to consider element-wise test function spaces in reference element coordinates, i.e. $V_k^h\left(\widetilde{\Omega}\right)$. Furthermore, since v does not need to satisfy any particular boundary condition, each test function is independent from the rest [66, p. 309]. Equation (3.18) can be enforced discretely element-by-element by choosing a test function among the set

$$\left\{\varphi_{rk}(\xi) \;:\; V_k^h\left(\widetilde{\Omega}\right) = \mathrm{span}\{\varphi_{rk}\}_{r=1}^{\mathrm{J}}\right\}, \tag{3.19}$$

and replacing the exact solution vector $\boldsymbol{q}(t,x)$ with its local high-order approximation, $\boldsymbol{q}_k^h(t,x)$. The result, in reference element coordinates, is

$$\frac{\Delta x_k}{2}\frac{\mathrm{d}}{\mathrm{d}t}\int_{-1}^{1} \tilde{\boldsymbol{q}}_k^h \varphi_{rk}\,\mathrm{d}\xi + \boldsymbol{f}\Big|_{\partial\Omega_k^R}\varphi_{rk}(1) - \boldsymbol{f}\Big|_{\partial\Omega_k^L}\varphi_{rk}(-1) - \int_{-1}^{1}\tilde{\boldsymbol{f}}_k^h\frac{\mathrm{d}\varphi_{rk}}{\mathrm{d}\xi}\,\mathrm{d}\xi = 0\,, \tag{3.20}$$

which has to hold for every element and test function (r being its index) of the discretization. It is convenient to group such test functions into the vector $\boldsymbol{\varphi}_k : \mathbb{R} \to \mathbb{R}^{\mathrm{J}}$:

$$\boldsymbol{\varphi}_k^{\mathrm{T}}(\xi) \coloneqq \begin{bmatrix}\varphi_{1k}(\xi) & \varphi_{2k}(\xi) & \ldots & \varphi_{\mathrm{J}k}(\xi)\end{bmatrix}. \tag{3.21}$$

3.3.3. Inter-element coupling

One last issue remains unsolved in (3.20): how to evaluate the flux vector at element interfaces. Compact discontinuous high order methods borrow the answer directly from the finite volume method. Suppose that a *numerical flux function* $\check{\boldsymbol{f}} : \left(\mathbb{R}^{\mathrm{I}}, \mathbb{R}^{\mathrm{I}}\right) \to \mathbb{R}^{\mathrm{I}}$ is available; the numerical fluxes at each edge of Ω_k are approximated as:

$$\boldsymbol{f}\Big|_{\partial\Omega_k^L} \simeq \check{\boldsymbol{f}}_k^L(t) \coloneqq \check{\boldsymbol{f}}\left(\boldsymbol{q}_{k-1}^h(t,x_k), \boldsymbol{q}_k^h(t,x_k)\right) \equiv \check{\boldsymbol{f}}\left(\tilde{\boldsymbol{q}}_{k-1}^h(t,+1), \tilde{\boldsymbol{q}}_k^h(t,-1)\right), \tag{3.22a}$$

$$\boldsymbol{f}\Big|_{\partial\Omega_k^R} \simeq \check{\boldsymbol{f}}_k^R(t) \coloneqq \check{\boldsymbol{f}}\left(\boldsymbol{q}_k^h(t,x_{k+1}), \boldsymbol{q}_{k+1}^h(t,x_{k+1})\right) \equiv \check{\boldsymbol{f}}\left(\tilde{\boldsymbol{q}}_k^h(t,+1), \tilde{\boldsymbol{q}}_{k+1}^h(t,-1)\right). \tag{3.22b}$$

Such a flux couples each element with the rest *weakly*, and in an optimally *compact* way: an element's discrete conservation law requires information from its two immediate neighbors only. This can be exploited to achieve very high parallel scalability in solver implementations, and is one of the main virtues of these kind of schemes. Additionally, this coupling mechanism manages to carry over the *local conservation* property [107], as defined for finite volumes, to a high-order context.

With high-order methods, the typical numerical flux function used is the *Riemann flux*, defined in §2.4. This choice arises quite naturally if we treat the neighborhood of every edge as defining a local Riemann problem. Unlike in (low-order) finite volumes, where the order of the discretization is determined by the order of this numerical flux, discontinuous finite element methods of the kind considered in this report maintain their order of accuracy (in smooth regions of the solution) regardless—it is determined by the type of solution discretization employed. While this makes the choice of Riemann solver less critical, a discontinuous high order discretization may still benefit from a less diffusive numerical flux.

[5] In some contexts (e.g. spectral methods [19, §1.2.5]) this choice is known as Bubnov-Galerkin—or, simply, Galerkin.

3.3.4. Semi-discrete hyperbolic conservation law

All pieces are now in place to explicitly write (2.1) as (3.16), a semi-discrete ODE for the degrees of freedom as a function of time. Expansion of $\widetilde{\boldsymbol{q}}_k^h$ and $\widetilde{\boldsymbol{f}}_k^h$ into their respective linear combinations of basis functions turns (3.20) into the sought-after result, which has to hold for every $\Omega_k \in \mathcal{T}^h$,

$$\frac{\mathrm{d}\hat{\boldsymbol{Q}}_k}{\mathrm{d}t}\boldsymbol{\mathcal{M}}_k + \left[\check{\boldsymbol{f}}\boldsymbol{\varphi}_k^\intercal\right]_{\partial\Omega_k} - \hat{\boldsymbol{F}}_k\boldsymbol{\mathcal{C}}_k = 0\,, \tag{3.23}$$

where

$$\left[\check{\boldsymbol{f}}\boldsymbol{\varphi}_k^\intercal\right]_{\partial\Omega_k} \coloneqq \check{\boldsymbol{f}}_k^R\boldsymbol{\varphi}_k^\intercal(1) - \check{\boldsymbol{f}}_k^L\boldsymbol{\varphi}_k^\intercal(-1) \equiv \begin{bmatrix} \check{f}_{1k}^R\varphi_{1k}(1) - \check{f}_{1k}^L\varphi_{1k}(-1) & \cdots & \check{f}_{1k}^R\varphi_{\mathrm{J}k}(1) - \check{f}_{1k}^L\varphi_{\mathrm{J}k}(-1) \\ \vdots & & \vdots \\ \check{f}_{\mathrm{I}k}^R\varphi_{1k}(1) - \check{f}_{\mathrm{I}k}^L\varphi_{1k}(-1) & \cdots & \check{f}_{\mathrm{I}k}^R\varphi_{\mathrm{J}k}(1) - \check{f}_{\mathrm{I}k}^L\varphi_{\mathrm{J}k}(-1) \end{bmatrix} \tag{3.24}$$

can be interpreted as the *net boundary fluxes* associated to each equation (row) and test function (column) in Ω_k, and

$$\boldsymbol{\mathcal{M}}_k \coloneqq \frac{\Delta x_k}{2}\int_{-1}^{1}\boldsymbol{\phi}_k\boldsymbol{\varphi}_k^\intercal\,\mathrm{d}\xi \equiv \frac{\Delta x_k}{2}\begin{bmatrix} \int_{-1}^{1}\phi_{1k}\varphi_{1k}\,\mathrm{d}\xi & \cdots & \int_{-1}^{1}\phi_{1k}\varphi_{\mathrm{J}k}\,\mathrm{d}\xi \\ \vdots & & \vdots \\ \int_{-1}^{1}\phi_{\mathrm{J}k}\varphi_{1k}\,\mathrm{d}\xi & \cdots & \int_{-1}^{1}\phi_{\mathrm{J}k}\varphi_{\mathrm{J}k}\,\mathrm{d}\xi \end{bmatrix}, \tag{3.25a}$$

$$\boldsymbol{\mathcal{C}}_k \coloneqq \int_{-1}^{1}\boldsymbol{\phi}_k\frac{\mathrm{d}\boldsymbol{\varphi}_k^\intercal}{\mathrm{d}\xi}\,\mathrm{d}\xi \equiv \begin{bmatrix} \int_{-1}^{1}\phi_{1k}\frac{\mathrm{d}\varphi_{1k}}{\mathrm{d}\xi}\,\mathrm{d}\xi & \cdots & \int_{-1}^{1}\phi_{1k}\frac{\mathrm{d}\varphi_{\mathrm{J}k}}{\mathrm{d}\xi}\,\mathrm{d}\xi \\ \vdots & & \vdots \\ \int_{-1}^{1}\phi_{\mathrm{J}k}\frac{\mathrm{d}\varphi_{1k}}{\mathrm{d}\xi}\,\mathrm{d}\xi & \cdots & \int_{-1}^{1}\phi_{\mathrm{J}k}\frac{\mathrm{d}\varphi_{\mathrm{J}k}}{\mathrm{d}\xi}\,\mathrm{d}\xi \end{bmatrix} \tag{3.25b}$$

are, respectively, the *consistent mass matrix* and the *discrete gradient operator* in Ω_k. The residual matrix for this arbitrary element, defined in accordance to §3.3.1, is:

$$\hat{\boldsymbol{\mathcal{R}}}_k \coloneqq \left(\hat{\boldsymbol{F}}_k\boldsymbol{\mathcal{C}}_k - \left[\check{\boldsymbol{f}}\boldsymbol{\varphi}_k^\intercal\right]_{\partial\Omega_k}\right)\boldsymbol{\mathcal{M}}_k^{-1}\,. \tag{3.26}$$

The components of the discrete mass and gradient operators involve an integral over the reference domain which still counts as a continuous operator not yet discretized. In 1D, these are the only integrals to be approximated (additional boundary integrals would appear in higher dimensions). Each method tries to exploit the properties of the basis and test functions to do so as efficiently and accurately as possible; in fact, it is typically the case that these terms are computed *exactly*. More details are provided in subsequent chapters. A consequence of approximating the flux vector via (3.14) is that, not only the mass matrix, but also a discrete gradient operator exists and can be precomputed; this makes the evaluation of (3.26) rather efficient.

Throughout this report, the matrices of degrees of freedom and residuals are arranged such that each row corresponds to a state variable, and each column to a basis component. I made such a choice during the design stages of the solver used in part II; and I have chosen to maintain it in this document so that it better serves as a documentation of sorts for said implementation. Under this convention, the mass and gradient matrices (3.25) have been derived as the transposes of their usual definitions (cf. [47, §2.2], where the latter is referred to as stiffness matrix). This also translates in these operators appearing as right-multiplications in (3.23) and (3.26). More generally, operations involving trial or test basis functions are encoded as matrix right-multiplications, while left-multiplications will generally involve the state or flux vector components (e.g. transformation to/from primitive variables).

3.4. The low-order case: finite volume discretization

Equation (3.23) can be seen as a generalization of the finite volume (FV) method to arbitary order; more specifically, for the choice of numerical fluxes made in this report, the *first order upwind* finite volume method (also known as Godunov's method). Refer to [82] for details. To see this, consider the situation $\mathrm{J} = 1$, $\phi_k = \varphi_k \coloneqq 1$. The discrete solution is piecewise constant and there is a single degree of freedom per equation, thus:

$$\int_{\Omega_k}\boldsymbol{q}(t,x)\,\mathrm{d}x = \int_{\Omega_k}\boldsymbol{q}_k(t,x)\,\mathrm{d}x \equiv \Delta x_k\boldsymbol{q}_k^h(t,x)\,, \qquad \boldsymbol{q}_k(t,x) \approx \boldsymbol{q}_k^h(t,x) \equiv \widetilde{\boldsymbol{q}}_k^h(t,\xi) \equiv \hat{\boldsymbol{q}}_k(t)\,. \tag{3.27}$$

Mass and gradient operators (3.25) in this case are:

$$\boldsymbol{\mathcal{M}}_k = \Delta x_k \,, \qquad \boldsymbol{\mathcal{C}}_k = 0 \,. \tag{3.28}$$

Therefore, the semi-discrete conservation law (3.23) reduces to:

$$\Delta x_k \frac{\mathrm{d}\boldsymbol{q}_k^h}{\mathrm{d}t} + \check{\boldsymbol{f}}_k^R - \check{\boldsymbol{f}}_k^L = 0 \,, \tag{3.29}$$

and the spatial residuals, (3.26), to:

$$\boldsymbol{r}_k = -\frac{1}{\Delta x_k}\left(\check{\boldsymbol{f}}_k^R - \check{\boldsymbol{f}}_k^L\right) . \tag{3.30}$$

If we now combine this spatial discretization with *first-order upwind finite differences in time* (explicit Euler's time scheme, see §7) such that $t = t_0, t_1, \ldots, t_n, \ldots, t_N$, we obtain *Godunov's method*:

$$\boldsymbol{q}_k^h(t_{n+1}, x) = \boldsymbol{q}_k^h(t_n, x) + \frac{\Delta t}{\Delta x_k}\left(\check{\boldsymbol{f}}_k^L(t_n) + \check{\boldsymbol{f}}_k^R(t_n)\right) . \tag{3.31}$$

The spatial discretization in the second-order finite volume method is identical—its only difference being the use of a centered numerical flux. Thus we reach the fundamental limitation of finite volume methods: to achieve high-order with a piecewise-constant (i.e. FV) discretization, *compactness cannot be maintained*. Spatial discretizations of the type (3.23) manage to overcome this limitation.

Let us use this opportunity to appreciate the strengths of (3.31). It is an extremely simple and concise expression, and yet, ensures exact local (and global) conservation of the solution. Moreover, it is both consistent[6] and (in the first-order case) total variation diminishing—hence stable—even for nonlinear flux functions, no limiting required (see §8). It is guaranteed to converge to *a* weak solution of the exact PDE and, although not proven, overwhelming numerical evidence suggests that it does so to the entropy solution (i.e. the physically meaningful one)—assuming the use of an adequate numerical flux and/or an entropy fix.

3.5. Initial condition projection

Just like in the continuous situation (2.1), (3.23) encodes the evolution of a system in time and (discrete) space, starting from a known discrete initial condition and subject to numerical boundary conditions (addressed in §3.6). It is therefore necessary to obtain a discrete counterpart of (2.4); a general way to do so is presented in this section.

Consider some projection of a component of the exact initial condition to the discrete solution space of $\mathcal{T}^h$; let it be the approximate initial condition, i.e.

$$q_i^0(x) \simeq q_i^h(0, x) \in S^h(\Omega) \,. \tag{3.32}$$

The goal is to evaluate $\hat{q}_{ijk}(0), \forall i, j, k$; this can be done weakly—in precisely the same sense as in §3.3 with the conservation law—by requiring, for every i, r, k combination:

$$\int_{\Omega_k} q_i^0 \varphi_{rk} \,\mathrm{d}x = \int_{\Omega_k} q_{ik}^h \varphi_{rk} \,\mathrm{d}x \,. \tag{3.33}$$

This equality defines what is known in the literature as the L^2 *projection* of the exact initial condition into the discrete function space [78, §10].

Expanding the approximate solution into its basis components, the entire matrix of degrees of freedom that solves (3.33) can be computed as:

$$\hat{\boldsymbol{Q}}_k = \left(\int_{\Omega_k} \boldsymbol{q}^0 \boldsymbol{\varphi}_k^\intercal \,\mathrm{d}x\right) \boldsymbol{\mathcal{M}}_k^{-1} \,. \tag{3.34}$$

[6] A discretization is said to be consistent with the continuous PDE(s) it models when the local truncation error of its solution tends to zero as $\Delta t \to 0$, for all smooth exact solutions of the latter. Equation (3.31) is such a method, provided that a *consistent numerical flux* is employed; this, in turn, means that: $\boldsymbol{f}(\boldsymbol{q}) = \check{\boldsymbol{f}}(\boldsymbol{q}, \boldsymbol{q})$ [82, §4.3.1]. All numerical fluxes mentioned in this report are consistent, and so are e.g. the centered fluxes in [60].

The matrix integral in (3.34), unlike the mass and gradient matrices previously (in most cases), cannot be computed exactly for an arbitrary initial condition. In the implementation used to obtain the results in part II, I opt for an adaptive Lobatto-Kronrod quadrature algorithm [106] with tolerances 10^{-9} (relative) and 10^{-13} (absolute). An alternative would be to simply use Gauss or Gauss-Lobatto quadrature of fixed order.

3.6. Numerical boundary conditions

Compact discontinuous methods lend themselves to a weak enforcement of boundary conditions, in the sense that the known value (in the Dirichlet case) at a given boundary edge is not directly assigned to the approximate solution, but is instead used to calculate a numerical flux crossing it. The simplest and most convenient of such approaches is known as *weak-Riemann* prescription [89]. It is used in all results shown in this report.

Let $\mathcal{G}^h \coloneqq \{\Omega_0, \Omega_{\mathrm{K}+1}\}$ be a set of fictitious or *ghost* elements such that:

$$\Omega_0 \coloneqq [x_0, x_1] \,, \qquad x_0 \coloneqq x_1 - \Delta x_1 \,, \tag{3.35a}$$

$$\Omega_{\mathrm{K}+1} \coloneqq [x_{\mathrm{K}+1}, x_{\mathrm{K}+2}] \,, \qquad x_{\mathrm{K}+2} \coloneqq x_{\mathrm{K}} + \Delta x_{\mathrm{K}} \,, \tag{3.35b}$$

and let them possess all features of any actual element, e.g. $S_0^h\left(\widetilde{\Omega}\right)$, $\mathcal{X}_0(\xi)$. The weak-Riemann approach consists on updating $\hat{\boldsymbol{Q}}_0(t)$ and $\hat{\boldsymbol{Q}}_{\mathrm{K}+1}(t)$ at every solver iteration[7], with the aim to obtain $\boldsymbol{q}_0^h(t, x_1)$ and $\boldsymbol{q}_{\mathrm{K}+1}^h(t, x_{\mathrm{K}+1})$ values representative of each physical boundary condition function and type. The two ghost elements are eventually used to evaluate the numerical flux at the boundary edges, just as if they were regular elements, thus coupling the approximate solution to the boundary condition. Some relevant kinds of boundary conditions will be briefly explored next.

3.6.1. Periodic

A periodic boundary condition on the left edge (treatment of a right boundary is analogous; see figure 3.4) implies:

$$\boldsymbol{q}^L(t) = \boldsymbol{q}^h(t, x_{\mathrm{K}+1}) \,. \tag{3.36}$$

In the weak-Riemann framework, this relation is approximated by setting $\widetilde{\boldsymbol{q}}_0^h(t,\xi) = \widetilde{\boldsymbol{q}}_{\mathrm{K}}^h(t,\xi)$, implying:

$$\hat{\boldsymbol{Q}}_0(t) = \hat{\boldsymbol{Q}}_{\mathrm{K}}(t) \,. \tag{3.37}$$

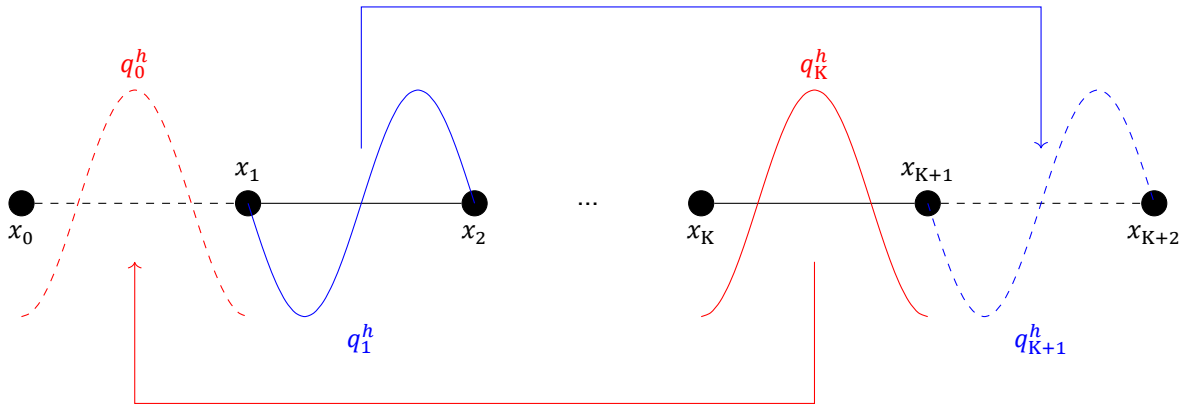


Figure 3.4: Left and right periodic boundary conditions (dashed lines denote ghost elements).

3.6.2. Farfield

Assume that the state vector is uniform and known for any position to the left of $\partial\Omega^L$, i.e. :

$$\boldsymbol{q}(t, x) = \boldsymbol{q}^L(t) \quad \text{for } x < x_1 \,. \tag{3.38}$$

[7] In my implementation, I do so at every Runge-Kutta stage (see §7).

Such a situation is modeled by setting $\mathrm{J} = 1$ in Ω_0 - so that S_0^h admits a constant function only—and requiring:

$$\hat{\boldsymbol{Q}}_0(t) \equiv \hat{\boldsymbol{q}}_0(t) = \boldsymbol{q}^L(t)\,. \tag{3.39}$$

Notice that inflow and outflow boundary conditions are automatically covered by this case because it is left to the Riemann solver to "decide" which characteristics go in and out of the computational domain. For the same reason, inflow and outflow constraints are guaranteed to be well-posed with this approach.

3.6.3. Transmissive

Also known as non-reflective [114, §14.2] or absorbing [82, §7.3.1], this boundary condition attempts to mimic an infinite domain by allowing any outgoing characteristic to exit it cleanly, i.e. without introducing any spurious wave that could influence the solution. For low-order methods, this is achieved numerically via the so-called *zero-order extrapolation* (described e.g. in [82, p. 134]). In the high-order case, I propose the following treatment, which generalizes the previous to a zero-gradient condition in all derivatives of the approximate solution at a boundary.

Consider $\Omega_1 \in \mathcal{T}^h$, and its corresponding ghost element, $\Omega_0 \in \mathcal{G}^h$. The opposite boundary case is treated analogously. Let us assume that:

$$S_1^h\left(\widetilde{\Omega}\right) = S_0^h\left(\widetilde{\Omega}\right) = \mathrm{span}\{\mathcal{P}_m\}_{m=0}^{\mathrm{p}}\,, \tag{3.40}$$

i.e. the first $\mathrm{J} = \mathrm{p} + 1$ Legendre polynomials are the basis functions of both (see §4.1.1 for details). The goal is to deduce a set of ghost element degrees of freedom which ensure that the approximate solution crosses the target boundary smoothly (in physical coordinates), e.g.:

$$\left.\frac{\partial^d \boldsymbol{q}_1^h}{\partial x^d}\right|_{x_1} = \left.\frac{\partial^d \boldsymbol{q}_0^h}{\partial x^d}\right|_{x_1} \quad \text{for } d = 0, 1, \dots, \mathrm{p}\,. \tag{3.41}$$

Expanding (3.41) into the Legendre basis functions, reveals the relationship between an element's degrees of freedom (left-hand-side) and those of its ghost counterpart (right-hand-side):

$$\hat{\boldsymbol{Q}}_1 \begin{bmatrix} \mathcal{P}_0(-1) & & & 0 \\ \mathcal{P}_1(-1) & \left.\frac{\mathrm{d}^1\mathcal{P}_1}{\mathrm{d}\xi^1}\right|_{-1} & & \\ \vdots & \vdots & \ddots & \\ \mathcal{P}_{\mathrm{p}}(-1) & \left.\frac{\mathrm{d}^1\mathcal{P}_{\mathrm{p}}}{\mathrm{d}\xi^1}\right|_{-1} & \cdots & \left.\frac{\mathrm{d}^{\mathrm{p}}\mathcal{P}_{\mathrm{p}}}{\mathrm{d}\xi^{\mathrm{p}}}\right|_{-1} \end{bmatrix} = \hat{\boldsymbol{Q}}_0 \begin{bmatrix} \mathcal{P}_0(1) & & & 0 \\ \mathcal{P}_1(1) & \left.\frac{\mathrm{d}^1\mathcal{P}_1}{\mathrm{d}\xi^1}\right|_{1} & & \\ \vdots & \vdots & \ddots & \\ \mathcal{P}_{\mathrm{p}}(1) & \left.\frac{\mathrm{d}^1\mathcal{P}_{\mathrm{p}}}{\mathrm{d}\xi^1}\right|_{1} & \cdots & \left.\frac{\mathrm{d}^{\mathrm{p}}\mathcal{P}_{\mathrm{p}}}{\mathrm{d}\xi^{\mathrm{p}}}\right|_{1} \end{bmatrix}. \tag{3.42}$$

The matrices in (3.42) are lower triangular because of the hierarchical nature of the Legendre basis. For $\mathrm{p} = 0$, the previous reduces to the approach in [82, p. 134]. Notice that each element's Jacobian can be neglected because of the definition in (3.35).

3.6.4. Reflective

Also known as the (oscillating) wall condition, this constraint models the opposite to the situation in the previous subsection: whenever a wave reaches the boundary, we wish to reflect it back in a physically meaningful way. This, of course, implies a dependence on the PDE being solved. For the Euler equations, the presence of a (non-porous) wall at $\partial\Omega^L$, moving at a speed $u_w(t)$ with negligible displacement amplitude, is modeled by setting [114, p. 496]:

$$\rho_0^h(t, x_1) = \rho_1^h(t, x_1)\,, \qquad u_0^h(t, x_1) = -u_1^h(t, x_1) + 2u_w(t)\,, \qquad p_0^h(t, x_1) = p_1^h(t, x_1)\,. \tag{3.43}$$

For the wave equation—interpreted as linearised acoustics i.e. (2.26)—the previous becomes [82, §7.3.4]:

$$q_{10}^h(t, x_1) = q_{11}^h(t, x_1)\,, \qquad q_{20}^h(t, x_1) = -q_{21}^h(t, x_1) + 2u_w(t)\,. \tag{3.44}$$

In both cases the wall constraint is recreated via a reflection symmetry[8] of the numerical solution about the position of the boundary, with the velocity component additionally being negated and displaced so that the average between left and right velocities, for all time instants, is the prescribed wall velocity.

As with the non-reflective case, I propose a methodology to enforce this condition with a compact, discontinuous, high order discretization. Assume, this time, that both ghost and non-ghost element at each side of the targeted boundary employ the same nodal (e.g. Lagrange, see §4) or quasi-nodal (in the sense of the B-spline functions, see §6) basis, or that their projection to one of these is available. Also, let $\boldsymbol{T}$ be defined as in (2.44) for the Euler case, and as $\boldsymbol{T} \coloneqq \boldsymbol{I} \in \mathbb{R}^{2\times 2}$ for the wave equation (assume that the velocity-like variable is q_2); the transformation to primary variables can be done for each vector of degrees of freedom, such that $\boldsymbol{T}_{jk} \coloneqq \boldsymbol{T}\left(\hat{\boldsymbol{q}}_{jk}\right)$. Then proceed as follows:

0. Consider a ghost element Ω_0 and its neighbor Ω_1 (left boundary; opposite case is analogous)
1. Deduce the (quasi-)nodal expansion coefficients of the solution *in primary variables*, i.e. :

$$\hat{v}_{ijk} \coloneqq \left(\boldsymbol{T}_{jk}\hat{\boldsymbol{q}}_{jk}\right)_i \quad \text{for} \begin{cases} i = 1, \dots, \mathrm{I} \\ j = 1, \dots, \mathrm{J} \\ k = 1 \end{cases} \tag{3.45}$$

2. Obtain the discrete reflection of pressure-like variables, i.e. :

$$\hat{v}_{ij0} = \hat{v}_{i\,(\mathrm{J}+1-j)\,1} \quad \text{for} \begin{cases} i \neq 2 \\ j = 1, \dots, \mathrm{J} \end{cases} \tag{3.46}$$

3. Prescribe the specified velocity at the boundary:

$$\hat{v}_{ij0} = 2u_w - \hat{v}_{i\,(\mathrm{J}+1-j)\,1} \quad \text{for} \begin{cases} i = 2 \\ j = 1, \dots, \mathrm{J} \end{cases} \tag{3.47}$$

4. Finally, transform the degrees of freedom of the ghost element back to conservative variables:

$$\hat{q}_{ijk} \coloneqq \left(\boldsymbol{T}_{jk}^{-1}\hat{\boldsymbol{v}}_{jk}\right)_i \quad \text{for} \begin{cases} i = 1, \dots, \mathrm{I} \\ j = 1, \dots, \mathrm{J} \\ k = 0 \end{cases} \tag{3.48}$$

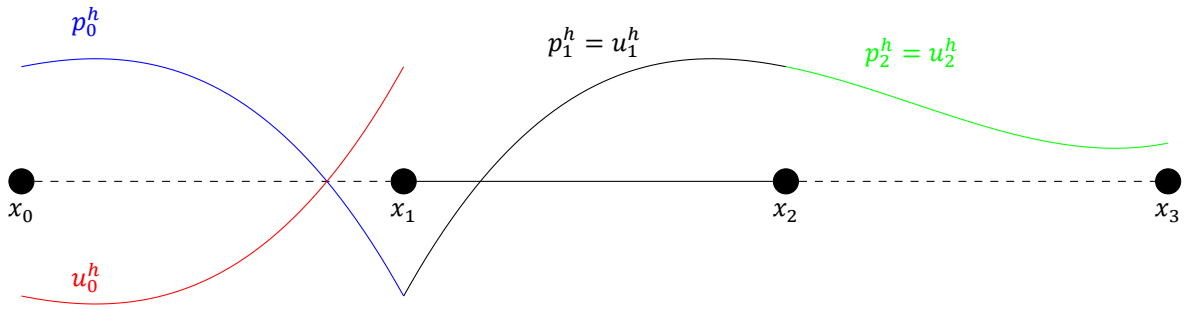


Figure 3.5: Reflective (left, $u_w = 0$) and non-reflective (right) boundary conditions for $\mathcal{T}^h = \{\Omega_1\}$.

[8] This can be inferred from the approach followed in the references cited above, which employ two ghost cells per side.

4

Discontinuous Galerkin Spectral Element Method (DGSEM)

The very first example in the literature of a high-order yet compact finite-element discretization for hyperbolic conservation laws is due to the work of Bernardo Cockburn and Chi-Wang Shu (and collaborators), during the last decade of the 20th century. These researchers, in a series of papers [22–24, 26, 27], combined an all but forgotten method developed in the 70s for the steady-state neutron transport equation of nuclear physics with an explicit Runge-Kutta time-integrator (see §7) and a non-linear stabilization mechanism (see §8). The result became known as *Runge-Kutta discontinuous Galerkin* (RKDG), which is nowadays often associated to the specific combination in which the order of the time scheme matches that of the spatial discretization. In this report, I shall use the term *modal* DG to refer to the particular subset of compact discontinuous high-order methods in which both trial and test function spaces employ *Legendre polynomials* as basis functions, independently of the time-discretization scheme or its order.

Discontinuous Galerkin (DG) is the oldest common ancestor of all members of the compact discontinuous high-order family. In this chapter, the general formulation of §3 is particularized to the first of the three discretization methods selected as research objects of this thesis. A concise description of this method for the three-dimensional Navier-Stokes equations, including implementation considerations, is found in [49]. The much simpler case of a one-dimensional hyperbolic conservation law described in this chapter borrows heavily from [66, section 8.1.4].

Essentially, DGSEM is the particular DG method that uses *Lagrange polynomial* basis functions, with a specific distribution of nodes such that the resulting discretization is mathematically identical—but computationally advantageous—to that of modal DG.

4.1. Spectral basis functions

Classical spectral methods (SM) are defined in [19, p. 3] as a particular subset of finite element methods (FEM) in which the trial basis functions $\phi(\xi)$ (see §3):

- Have nonzero support over the entire domain.
- Are infinitely differentiable.
- Are orthogonal (or nearly so).

Since they offer no possibility of h-refinement, an increase in the number of degrees of freedom in these methods is typically achieved through an increase in order of accuracy. As a consequence, they often experience exponential—sometimes called spectral—convergence. Moreover, for periodic boundary conditions, they inherit the non-diffusive nature of continuous finite element methods (since there is no mechanism by which numerical diffusion can be introduced). A canonical choice of basis functions for a spectral method is the set of sinusoids of a truncated Fourier series.

Modern spectral methods have evolved closer to finite elements and finite volumes, borrowing their multi-domain approach to discretization [20]. These are referred to as spectral *element* methods (SEM),

since their basis functions adhere to the definition above in an element-local fashion. The distinction between FEM and SEM has thus become rather ambiguous; in practice, the spectral denomination suggests that a method is designed to be used in a high order of accuracy/low number of elements setting—at least, this is the case for DGSEM [49]. Be it as it may, both DGSEM and modal DG satisfy the formal requirements to be considered spectral element methods.

4.1.1. Legendre polynomials

Assume that, rather than e.g. sinusoids, we would like to use a polynomial basis for the J-dimensional trial space, $S_k^h\left(\widetilde{\Omega}\right)$, of a spectral element discretization. The straight-forward choice: $\left\{1, \xi, \xi^2, \xi^3, \ldots, \xi^{\mathrm{p}}\right\}$ would theoretically allow an exact representation of any polynomial solution up to degree $\mathrm{p} = \mathrm{J} - 1$, making its *formal* order of accuracy $\mathrm{p}+1 = \mathrm{J}$. However, in practice, this basis turns out to be very badly conditioned[1] [47, §3.1].

To circumvent this setback (and in order to satisfy the definition of a spectral method above), we can simply *orthogonalize* the previous in an L^2 Gram-Schmidt sense. The result is the set $\{\mathcal{P}_m(\xi)\}_{m=0}^{\mathrm{J}-1}$ of *Legendre polynomials* [66, §1.8.1]:

$$\begin{aligned}
\mathcal{P}_0 &= 1\,, \\
\mathcal{P}_1(\xi) &= \xi\,, \\
\mathcal{P}_2(\xi) &= \frac{1}{2}\left(3\xi^2 - 1\right)\,, \\
\mathcal{P}_3(\xi) &= \frac{1}{2}\left(5\xi^3 - 3\xi\right)\,, \\
\mathcal{P}_4(\xi) &= \frac{1}{8}\left(35\xi^4 - 30\xi^2 + 3\right)\,, \\
&\vdots \\
\mathcal{P}_{m+1}(\xi) &= \frac{2m+1}{m+1}\xi\mathcal{P}_m(\xi) - \frac{m}{m+1}\mathcal{P}_{m-1}(\xi), \quad \text{for } m > 0\,.
\end{aligned} \tag{4.1}$$

Note that, keeping consistency with typical notation, these basis functions are indexed starting from 0; this is so that $\mathcal{P}_m$ is a polynomial of degree m. The following three-term relationship holds [66, §1.8.1]:

$$(2m+1)\mathcal{P}_m(\xi) = \mathcal{P}'_{m+1}(\xi) - \mathcal{P}'_{m-1}(\xi)\,. \tag{4.2}$$

Consequently, arbitrary κ-th order derivatives can be evaluated for $\xi \in [-1, 1]$ as:

$$\frac{\mathrm{d}^\kappa \mathcal{P}_{m+1}}{\mathrm{d}\xi^\kappa}(\xi) = (2m+1)\frac{\mathrm{d}^{\kappa-1}\mathcal{P}_m}{\mathrm{d}\xi^{\kappa-1}}(\xi) + \frac{\mathrm{d}^\kappa \mathcal{P}_{m-1}}{\mathrm{d}\xi^\kappa}(\xi)\,. \tag{4.3}$$

Some properties of these polynomials are listed next [2, 6].

Property 4.1. For $m > 0$, it holds that:

$$\int_{-1}^{1} \mathcal{P}_m \,\mathrm{d}\xi = 0\,. \tag{4.4}$$

As a consequence, the leading expansion coefficient of a Legendre-based discretization of Ω_k is equal to the average of the approximate solution over it:

$$\frac{1}{\Delta x_k}\int_{\Omega_k} q_{ik}^h(t,x)\,\mathrm{d}x = \frac{1}{2}\left(\int_{-1}^{1} \hat{q}_{i1k}(t)\mathcal{P}_0(\xi)\,\mathrm{d}\xi + \int_{-1}^{1} \hat{q}_{i2k}(t)\mathcal{P}_1(\xi)\,\mathrm{d}\xi + \ldots\right) \equiv \hat{q}_{i1k}(t)\,. \tag{4.5}$$

Property 4.2 (parity). Legendre polynomials are either even or odd, such that:

$$\mathcal{P}_m(-\xi) = (-1)^m \mathcal{P}_m(\xi)\,. \tag{4.6}$$

In particular, $\mathcal{P}_m(1) = 1$ for all m; $\mathcal{P}_m(-1) = -1$ if m is odd, and $\mathcal{P}_m(-1) = 1$ if m is even.

Property 4.3 (orthogonality). Legendre polynomials satisfy, for $j, r > 0$:

$$\int_{-1}^{1} \mathcal{P}_{j-1}\mathcal{P}_{r-1}\,\mathrm{d}\xi = \begin{cases} \frac{2}{2j-1} & \text{if } j = r \\ 0 & \text{otherwise} \end{cases}\,. \tag{4.7}$$

[1] The condition number of $\boldsymbol{\mathcal{M}}_k$ would become very large as p increases.

4.1.2. Lagrange polynomials

The Legendre functions (4.1) are but one of many sets of polynomials that span $S_k^h\left(\widetilde{\Omega}\right)$ (see figure 4.1). One of such alternatives is the set of *Lagrange polynomials* $\{l_j(\xi)\}_{j=1}^{\mathrm{J}}$, usually defined as

$$l_j(\xi) \coloneqq \prod_{\substack{n=1 \\ n\neq j}}^{\mathrm{J}} \frac{\xi - \xi_n}{\xi_j - \xi_n}\,, \tag{4.8}$$

or, equivalently, in so-called *barycentric* form—which is preferred in practice [66, §3.4]—

$$l_j(\xi) = \frac{w_j^b}{\left(\xi - \xi_j\right)\left(\sum_{j=1}^{\mathrm{J}} \frac{w_j^b}{\xi - \xi_j}\right)}\,, \qquad w_j^b \coloneqq \frac{1}{\prod_{\substack{n=1 \\ n\neq j}}^{\mathrm{J}} \left(\xi_j - \xi_n\right)}\,; \tag{4.9}$$

where w_j^b is the *barycentric weight* associated to $l_j(\xi)$. The defining feature of this basis is that:

$$l_j(\xi_n) = \delta_{jn} \coloneqq \begin{cases} 1 & \text{if } n = j \\ 0 & \text{otherwise} \end{cases}\,. \tag{4.10}$$

Derivatives of Lagrange polynomials can be evaluated in a number of ways [66, §3.5]. If they are only needed at the nodes (such as is the case in DGSEM), an efficient approach is to define the following *derivative matrix*:

$$\boldsymbol{\mathcal{D}} \coloneqq \begin{bmatrix} d_{11} & \cdots & d_{1\mathrm{J}} \\ \vdots & \ddots & \vdots \\ d_{\mathrm{J}1} & \cdots & d_{\mathrm{JJ}} \end{bmatrix} = \begin{bmatrix} l_1'(\xi_1) & \cdots & l_1'(\xi_{\mathrm{J}}) \\ \vdots & \ddots & \vdots \\ l_{\mathrm{J}}'(\xi_1) & \cdots & l_{\mathrm{J}}'(\xi_{\mathrm{J}}) \end{bmatrix}\,, \tag{4.11}$$

the entries of which which, using the barycentric form, are [66, §3.5.2]:

$$d_{jj} = -\sum_{n=1}^{\mathrm{J}} d_{jn}\,, \qquad d_{jn} = \frac{w_n^b}{w_j^b\left(\xi_j - \xi_n\right)} \quad \text{for } n \neq j\,. \tag{4.12}$$

Analogous higher-order derivative matrices, $\boldsymbol{\mathcal{D}}^{(\kappa)}$, can be defined recursively [66, p. 82]:

$$d_{jj}^{(\kappa)} = -\sum_{n=1}^{\mathrm{J}} d_{jn}^{(\kappa)}\,, \qquad d_{jn}^{(\kappa)} = \frac{\kappa}{\xi_j - \xi_n}\left(\frac{w_n^b}{w_j^b} d_{jj}^{(\kappa-1)} - d_{jn}^{(\kappa-1)}\right) \quad \text{for } n \neq j\,. \tag{4.13}$$

Unlike in the modal case, each Lagrange basis function is itself a polynomial of degree $\mathrm{p} = \mathrm{J} - 1$. Every $l_j(\xi)$ is associated to one coordinate, ξ_j—we say that a *node* exists there. The distribution of these $\mathrm{p}+1$ nodes is arbitrary, as long as they are unique; each distinct set of nodes results in a specific basis, with particular properties. This is discussed further in the context of DGSEM in §4.2.2.

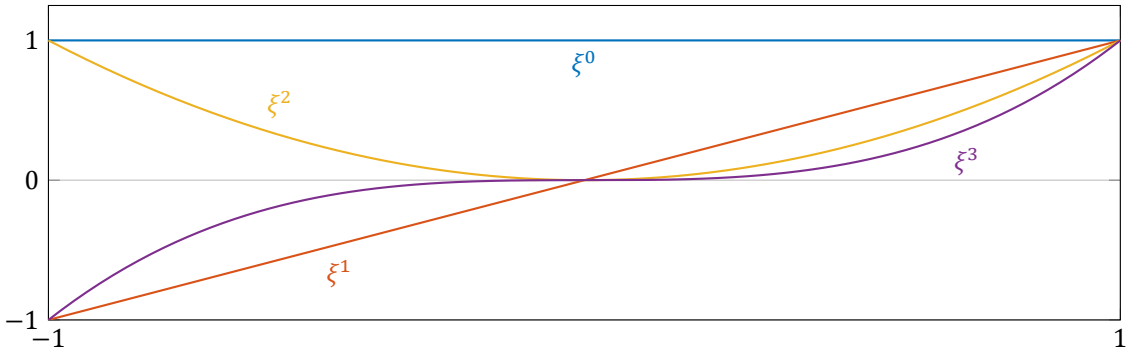
4.2. DGSEM semi-discretization

As in the general formulation, the domain Ω is discretized into K elements, $\Omega_k \in \mathcal{T}^h$ (§3.1). Each of those has a counterpart in reference coordinates, ξ, such that $\widetilde{\Omega} = [-1, 1]$ (see §3.1.1). Let trial and test spaces of each reference element be identical *polynomial* function spaces of degree p, i.e.: $S_k^h(\widetilde{\Omega}) = V_k^h(\widetilde{\Omega})$, $\dim S_k^h(\widetilde{\Omega}) = \mathrm{J}$ and $\mathrm{p} \equiv \mathrm{J} - 1$. As a consequence, the number of dimensions of these spaces is equal to the formal order of accuracy in any approximated function $\widetilde{q}_{ik}^h \in S_k^h(\widetilde{\Omega})$, which means that any polynomial of degree p or less can be represented exactly in the discretization.

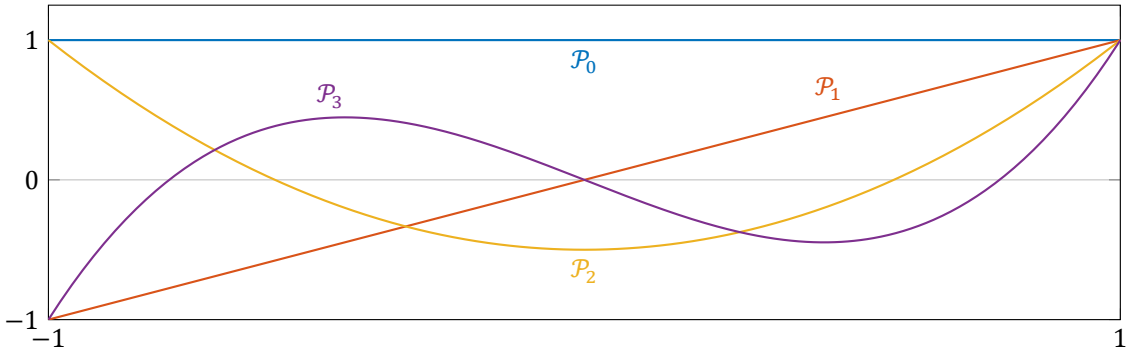
4.2.1. Modal representation

Let us first consider the Legendre basis of length J spanning $S_k^h\left(\widetilde{\Omega}\right)$. The general semi-discrete conservation law (3.23) particularized to an arbitrary mode $1 \leq r \leq \mathrm{J}$ and element $\Omega_k \in \mathcal{T}^h$, by properties 4.2 and 4.3, reads:

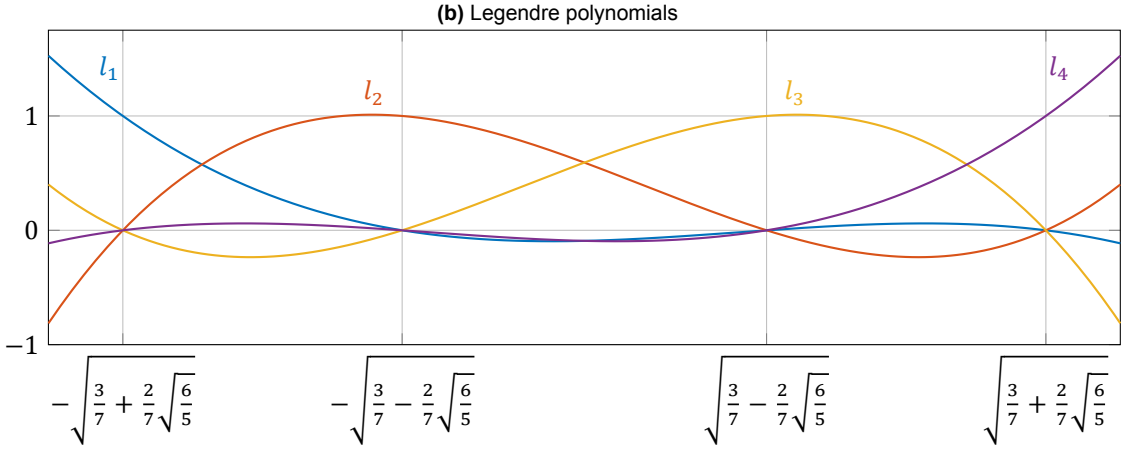
$$\frac{\Delta x_k}{2r-1}\frac{\mathrm{d}\hat{\boldsymbol{q}}_{rk}}{\mathrm{d}t} + \check{\boldsymbol{f}}_k^R + (-1)^r \check{\boldsymbol{f}}_k^L = \sum_{j=1}^{\mathrm{J}} \hat{\boldsymbol{f}}_{jk} \int_{-1}^{1} \mathcal{P}_{j-1}\mathcal{P}_{r-1}'\,\mathrm{d}\xi\,, \tag{4.14}$$



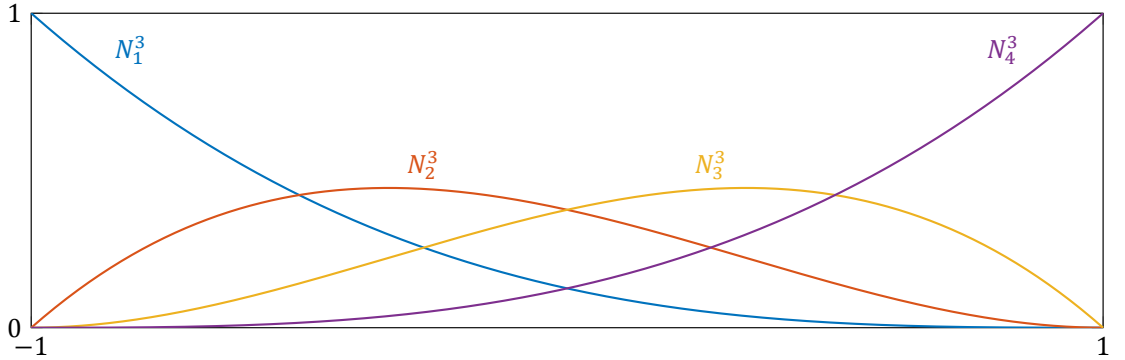
(a) Simple monomials



(b) Legendre polynomials



(c) Lagrange polynomials, Gauss-Legendre nodes



(d) Bernstein polynomials (see §6.2.2)

Figure 4.1: Polynomial bases spanning a 4-dimensional reference finite-element space.

with left and right edge Riemann fluxes as defined in (3.22).

The integrand on the right-hand-side of (4.14) is a polynomial of degree $2\mathrm{p}-1$, and can thus be computed exactly via either Gauss-Legendre or Gauss-Lobatto quadrature of only $\mathrm{p}+1$ points (see table 4.2). Should we attempt to avoid aliasing errors by evaluating the interior fluxes as $\boldsymbol{f}\left(\widetilde{\boldsymbol{q}}_k^h\right)$, the right-hand-side would instead be:

$$\int_{-1}^{1} \boldsymbol{f}\left(\widetilde{\boldsymbol{q}}_k^h\right) \mathcal{P}'_{r-1} \,\mathrm{d}\xi = \int_{-1}^{1} \boldsymbol{f}\left(\sum_{j=1}^{\mathrm{J}} \hat{\boldsymbol{q}}_{jk} \mathcal{P}_{j-1}\right) \mathcal{P}'_{r-1} \,\mathrm{d}\xi \,, \tag{4.15}$$

and could either not be evaluated exactly, or doing so would require a number of quadrature points well above J (see table 4.2)[2]. That being said, approximating the aforementioned integral with sufficiently high-order quadrature (dealiasing through over-integration or super-collocation) does mitigate aliasing-driven instabilities in under-resolved simulations [65]. The appeal of J-point quadrature will become clear in §4.2.3. Also, see §4.2.5 for some further remarks.

4.2.2. Nodal representation

Given a set of unique (assumed in increasing order, without loss of generality) nodal coordinates, $\xi_1 < \xi_2 < \cdots < \xi_{\mathrm{J}}$, there exists a set of Lagrange basis functions (§4.1.2) defining the polynomial of minimum degree that interpolates over any given function evaluated at each and every ξ_j. One of such so-called *Lagrange interpolants* (a linear combination of Lagrange basis functions) is used in DGSEM to represent the approximate counterpart of each exact state and flux vector component.

Introducing *nodal expansion coefficients* for the solution and flux interpolants:

$$\check{\boldsymbol{Q}}_k = \begin{bmatrix} \check{\boldsymbol{q}}_{1k} & \check{\boldsymbol{q}}_{2k} & \cdots & \check{\boldsymbol{q}}_{\mathrm{J}k} \end{bmatrix} \coloneqq \begin{bmatrix} \widetilde{\boldsymbol{q}}_k^h(t,\xi_1) & \widetilde{\boldsymbol{q}}_k^h(t,\xi_2) & \cdots & \widetilde{\boldsymbol{q}}_k^h(t,\xi_{\mathrm{J}}) \end{bmatrix} , \tag{4.16a}$$

$$\check{\boldsymbol{F}}_k = \begin{bmatrix} \check{\boldsymbol{f}}_{1k} & \check{\boldsymbol{f}}_{2k} & \cdots & \check{\boldsymbol{f}}_{\mathrm{J}k} \end{bmatrix} \coloneqq \begin{bmatrix} \widetilde{\boldsymbol{f}}_k^h(t,\xi_1) & \widetilde{\boldsymbol{f}}_k^h(t,\xi_2) & \cdots & \widetilde{\boldsymbol{f}}_k^h(t,\xi_{\mathrm{J}}) \end{bmatrix} ; \tag{4.16b}$$

the former satisfy:

$$\widetilde{\boldsymbol{q}}_k^h(t,\xi) = \sum_{j=1}^{\mathrm{J}} \hat{\boldsymbol{q}}_{jk}(t) \mathcal{P}_{j-1}(\xi) = \sum_{j=1}^{\mathrm{J}} \check{\boldsymbol{q}}_{jk}(t) l_j(\xi) \implies \check{\boldsymbol{q}}_{jk}(t) = \sum_{j=1}^{\mathrm{J}} \hat{\boldsymbol{q}}_{jk}(t) \mathcal{P}_{j-1}(\xi_n) \tag{4.17}$$

and, therefore, the latter can be conveniently evaluated as:

$$\check{\boldsymbol{f}}_{jk} = \boldsymbol{f}\left(\widetilde{\boldsymbol{q}}_k^h(\xi_j)\right) \equiv \boldsymbol{f}(\check{\boldsymbol{q}}_{jk}) \,. \tag{4.18}$$

The relationship between Legendre and Lagrange expansion coefficients is used in this report to extend hierarchical limiters to Lagrange-based methods, including DGSEM (see §8), and to define one type of numerical boundary condition (§3.6.3). Having chosen a set of nodes, we can define a linear operator relating these two representations:

$$\check{\boldsymbol{Q}}_k \equiv \hat{\boldsymbol{Q}}_k \boldsymbol{\mathcal{V}} \,, \qquad \boldsymbol{\mathcal{V}} \coloneqq \begin{bmatrix} \mathcal{P}_0(\xi_1) & \mathcal{P}_0(\xi_2) & \cdots & \mathcal{P}_0(\xi_{\mathrm{J}}) \\ \mathcal{P}_1(\xi_1) & \mathcal{P}_1(\xi_2) & \cdots & \mathcal{P}_1(\xi_{\mathrm{J}}) \\ \vdots & \vdots & & \vdots \\ \mathcal{P}_{\mathrm{p}}(\xi_1) & \mathcal{P}_{\mathrm{p}}(\xi_2) & \cdots & \mathcal{P}_{\mathrm{p}}(\xi_{\mathrm{J}}) \end{bmatrix} . \tag{4.19}$$

Known in the literature as a *generalized Vandermonde matrix*, $\boldsymbol{\mathcal{V}}$ is invertible and well-conditioned for non-uniform nodal distributions [47, §3.1] (see figure 4.2).

Using Lagrange polynomials as trial and test basis functions in (3.23), instead of (4.14), we have:

$$\frac{\Delta x_k}{2} \sum_{j=1}^{\mathrm{J}} \frac{\mathrm{d}\check{\boldsymbol{q}}_{jk}}{\mathrm{d}t} \int_{-1}^{1} l_j l_r \,\mathrm{d}\xi + \left[\check{\boldsymbol{f}} l_r\right]_{\partial\Omega_k} = \sum_{j=1}^{\mathrm{J}} \check{\boldsymbol{f}}_{jk} \int_{-1}^{1} l_j l'_r \,\mathrm{d}\xi \,. \tag{4.20}$$

[2] For example, in the Euler equations with polynomial conservative variables, where the momentum and energy flux components are rational functions (since because pressure involves a quotient of polynomials, in this case). If primitive variables are discretized instead, the velocity and pressure fluxes are polynomials of degree $3\mathrm{p}$ (in the compressible case) [36].

Nodes	$\widetilde{\mathcal{M}}$	$\operatorname{cond}\left(\widetilde{\mathcal{M}}\right)$
Equidistant	$\begin{bmatrix} 0.1524 & 0.1179 & -0.04286 & 0.02262 \\ 0.1179 & 0.7714 & -0.09643 & -0.04286 \\ -0.04286 & -0.09643 & 0.7714 & 0.1179 \\ 0.02262 & -0.04286 & 0.1179 & 0.1524 \end{bmatrix}$	9.361
Gauss-Lobatto	$\begin{bmatrix} 0.1429 & 0.05324 & -0.05324 & 0.02381 \\ 0.05324 & 0.7143 & 0.119 & -0.05324 \\ -0.05324 & 0.119 & 0.7143 & 0.05324 \\ 0.02381 & -0.05324 & 0.05324 & 0.1429 \end{bmatrix}$	8.65148
Chebyshev	$\begin{bmatrix} 0.2309 & 0.05211 & -0.03544 & 0.0167 \\ 0.05211 & 0.6619 & 0.05711 & -0.03544 \\ -0.03544 & 0.05711 & 0.6619 & 0.05211 \\ 0.0167 & -0.03544 & 0.05211 & 0.2309 \end{bmatrix}$	3.68089
Gauss-Legendre	$\begin{bmatrix} 0.3479 & 0 & 0 & 0 \\ 0 & 0.6521 & 0 & 0 \\ 0 & 0 & 0.6521 & 0 \\ 0 & 0 & 0 & 0.3479 \end{bmatrix}$	1.87476

Table 4.1: 4th-order DGSEM reference element mass matrices and their condition numbers, for various kinds of node distributions. Inner products between pairs of Lagrange basis functions computed using adaptive quadrature, with machine precision absolute tolerance. The sum of all entries adds to 2 in all cases.

4.2.3. Collocated quadrature

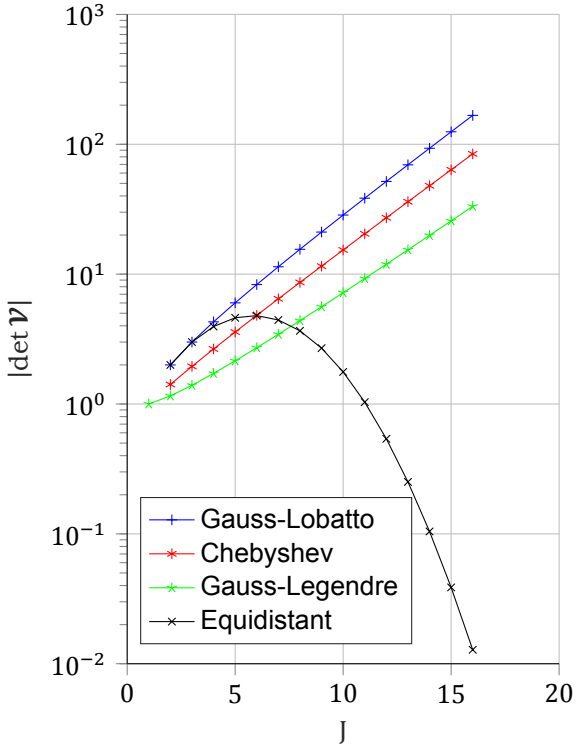
DGSEM employs *collocated* interpolation and integration points to evaluate the integrals in (4.20) [49]. This is done by selecting node locations that match the integration points of a chosen quadrature rule. Both Gauss-Legendre and Gauss-Lobatto quadratures are commonly employed with DGSEM; some basic differences between the two are summarized in table 4.2, and exemplified in figure 4.3. The resulting variants of DGSEM are compared in [38, 40, 67], and references therein. In this literature study, I consider Gauss-Legendre DGSEM only. For additional details about it, refer to [66, §8.1.4]. The alternative Gauss-Lobatto DGSEM variant can be seen as a particular case of the one-dimensional nodal DG method described in [47, §3]. Given that (standard) DGSEM assumes both approximate solution and flux to be Lagrange interpolants, each mass and discrete gradient matrix entry integrand is a polynomial, respectively, of degree $2\mathrm{p}$ and $2\mathrm{p}-1$. Therefore these operators can be precomputed exactly[3] and applied directly to the time-dependent degrees of freedom at each residual evaluation, avoiding any interpolation within[4] the element.

Quadrature	Number of nodes	Degree of exactness [104]	Nodes at $\xi = \pm 1$?
Gauss-Legendre	$\mathrm{p}+1 \geq 1$	$2\mathrm{p}+1$	No
Gauss-Lobatto	$\mathrm{p}+1 > 1$	$2\mathrm{p}-1$	Yes

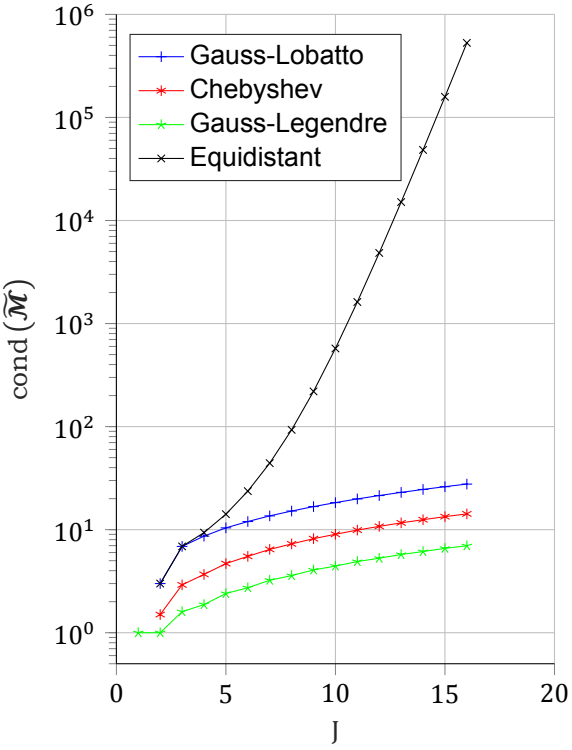
Table 4.2: Comparison between Gauss-Legendre and Gauss-Lobatto quadratures, for a polynomial approximation of degree p. Note that Gauss-Lobatto would not be defined for $\mathrm{p} = 0$, while Gauss-Legendre DGSEM inherently reduces to FVM.

[3] In Gauss-Lobatto DGSEM, it is still possible to obtain an exact mass matrix by exploiting an algebraic relationship that it has with the Vandermonde matrix associated with a normalized set of Legendre polynomials, see [47, §3.2].

[4] When using Gauss-Legendre nodes, however, the approximate state does need to be extrapolated at both edges of each element, in every residual evaluation, in order to compute the Riemann fluxes used for coupling.



(a) Vandermonde determinant; higher is better [47, p. 49]



(b) Mass matrix condition number; lower is better

Figure 4.2: Quantities associated with the robustness of a nodal discretization, for various types of node distributions, as a function of formal order of accuracy.

4.2.4. Semi-discrete DGSEM operators

Collocated Gauss-Legendre quadrature applied to the integral on the left-hand-side of (4.20) reveals:

$$\int_{-1}^{1} l_j l_r \,\mathrm{d}\xi = \sum_{n=1}^{\mathrm{J}} l_j(\xi_n) l_r(\xi_n) w_n = \begin{cases} w_r & \text{if } j = r \\ 0 & \text{otherwise} \end{cases}, \tag{4.21}$$

i.e. the Legendre polynomial are orthogonal when using collocated Gauss-Legendre nodes (see table 4.1). The diagonal matrix of Gauss-quadrature weights may be interpreted as a reference element mass matrix:

$$\widetilde{\mathcal{M}} \coloneqq \begin{bmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_{\mathrm{J}} \end{bmatrix}, \qquad \widetilde{\mathcal{M}}^{-1} = \begin{bmatrix} \frac{1}{w_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{w_{\mathrm{J}}} \end{bmatrix}. \tag{4.22}$$

The discrete gradient component on the right-hand-side of (4.20) becomes:

$$\int_{-1}^{1} l_j l_r' \,\mathrm{d}\xi = \sum_{n=1}^{\mathrm{J}} l_j(\xi_n) l_r'(\xi_n) w_n = l_r'(\xi_j) w_j \,, \tag{4.23}$$

and, in turn, warrants the definition of the following *modified derivative matrix*:

$$\widetilde{\mathcal{D}} \coloneqq -\widetilde{\mathcal{M}} \mathcal{D}^{\mathsf{T}} \widetilde{\mathcal{M}}^{-1} \,, \qquad \tilde{d}_{rj} \coloneqq -\frac{w_j}{w_r} d_{jr} \,. \tag{4.24}$$

The general hyperbolic conservation law (2.1) in semi-discrete form (3.23), for DGSEM, is:

$$\frac{\mathrm{d}\check{\boldsymbol{Q}}_k}{\mathrm{d}t} + \frac{2}{\Delta x_k}\left(\check{\boldsymbol{f}}_k^R \tilde{l}(1) - \check{\boldsymbol{f}}_k^L \tilde{l}(-1) + \check{\boldsymbol{F}}_k \widetilde{\mathcal{D}}\right) = 0 \,, \tag{4.25}$$

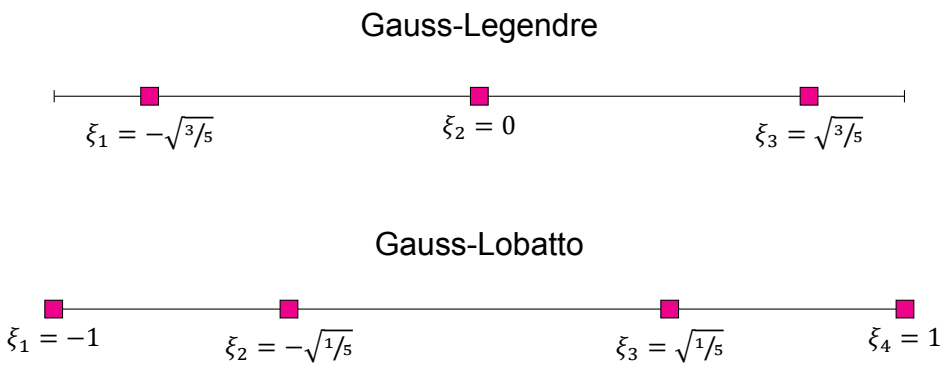


Figure 4.3: Gauss-Legendre and Gauss-Lobatto quadrature points. Both distributions would result in a quadrature rule of 5th degree of exactness (i.e. exact for polynomials up to 5th degree).

where:

$$\tilde{l}(\xi) \coloneqq l^{\mathsf{T}}(\xi)\widetilde{\boldsymbol{\mathcal{M}}}^{-1} = \left[\frac{l_1(\xi)}{w_1} \quad \frac{l_2(\xi)}{w_2} \quad \dots \quad \frac{l_{\mathrm{J}}(\xi)}{w_{\mathrm{J}}}\right]. \tag{4.26}$$

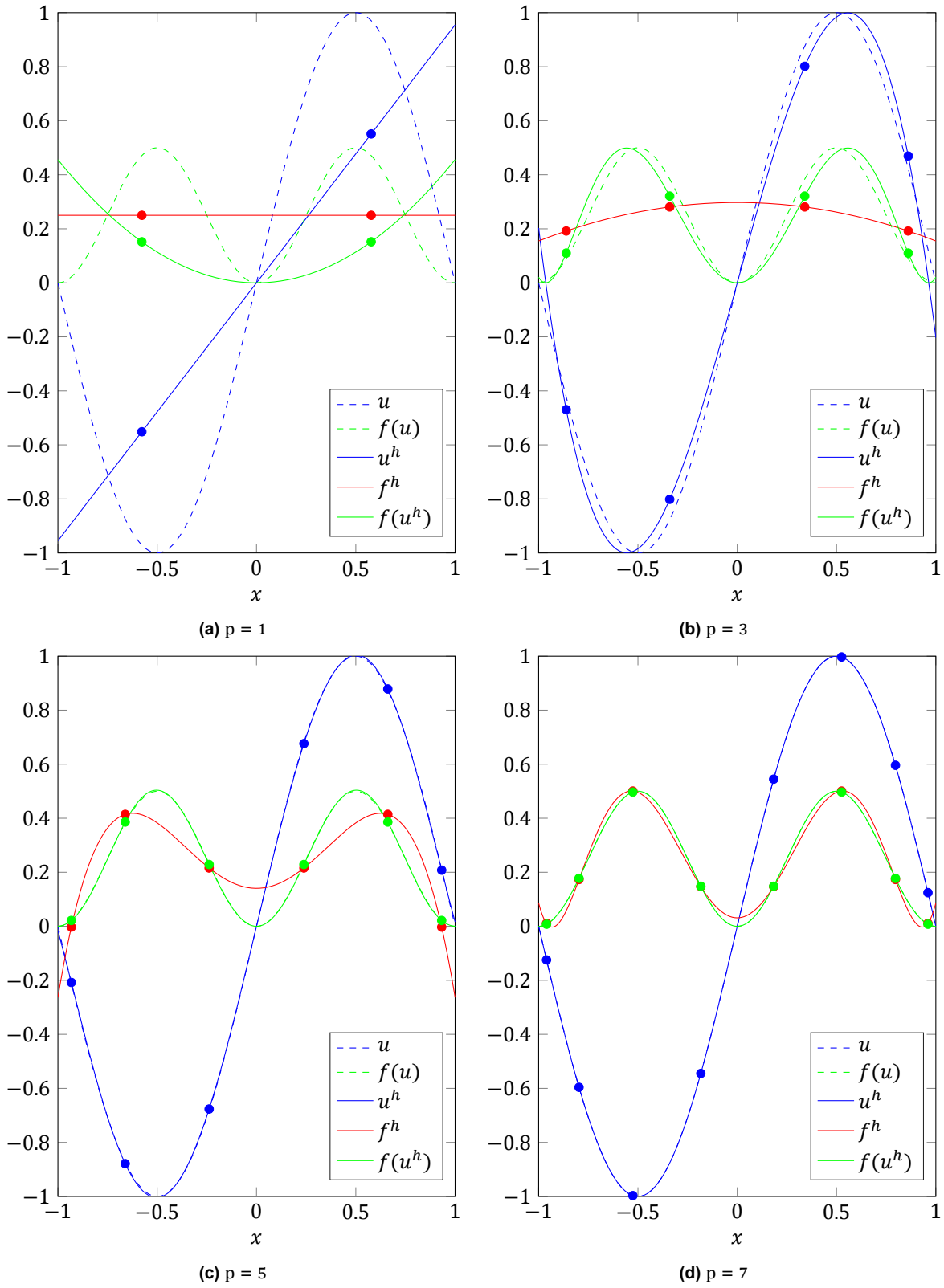
This corresponds to the particular case of (3.23) in which mass and gradient matrices are, respectively:

$$\boldsymbol{\mathcal{M}}_k = \frac{\Delta x_k}{2}\widetilde{\boldsymbol{\mathcal{M}}}\,, \qquad \boldsymbol{\mathcal{C}}_k = \widetilde{\boldsymbol{\mathcal{M}}}\boldsymbol{\mathcal{D}}^{\mathsf{T}}\,. \tag{4.27}$$

4.2.5. Polynomial aliasing

It has been assumed in the previous derivation that the flux vector component interpolants, which are polynomials of degree p, are representative of the (potentially non-polynomial, see §4.2.1) exact flux vector components. This is unfortunately not the case if the discretization does not have enough resolution—i.e. if the exact solution is *underresolved*. In such a situation, interpolation is prone to aliasing errors, which in the context of DG is often referred to as *under-integration*—since it is often seen as a failure to integrate the non-interpolatory flux term (4.15) exactly.

On the positive side, there is a simple (yet possibly expensive) remedy for this problem: by simply increasing the resolution of the discretization, aliasing effects should become negligible as the exact solution becomes better and better resolved. Of course, this only applies to smooth solutions, and is only feasible for exact solutions that are not prohibitively rich in terms of flow scales. An example of the feasibility of this “brute force” approach is shown in figure 4.4. In the present study, no explicit de-aliasing strategy is employed. For more details on this phenomenon, refer to [11, 39].



(a) p = 1

(b) p = 3

(c) p = 5

(d) p = 7

Figure 4.4: Demonstration of polynomial aliasing in DGSEM for $u(x) = \sin(\pi x)$ and $f(u) = \frac{u^2}{2}$. The approximate solution and flux, u^h and f^h, are obtained by projecting their exact counterparts (dashed) into the space of all polynomials of degree p (via adaptive quadrature with machine precision tolerance, see §3.5). Aliasing is manifesting as a discrepancy between $f(u^h)$ and f^h; for a sufficiently well-resolved solution, the aliased flux becomes acceptable. Dot markers represent Gauss-Legendre nodes; nodal distribution has no influence in these results.

5

Flux Reconstruction (FR) or Correction Procedure via Reconstruction (CPR)

The second method subject to study in this work was originally proposed for hyperbolic conservation laws by Huynh [55] under the name *flux reconstruction*, as a framework that unified DG (§4) and spectral difference (SD) or staggered grid (SG) multi-domain spectral methods. At the same time, the approach provided a mechanism (the choice of *correction function*) through which new methods could be constructed. Some of these (FR versions of DG and SG/SD) had already been formulated independently prior to its introduction; others have appeared thanks to it [55, 121]. The *correction procedure via reconstruction* denomination was adopted later; note that both FR and CPR refer to the exact same methodology. The main selling points of FR/CPR are its *simplicity*, *flexibility*, and alleged *computational efficiency* [128].

The fundamental idea behind FR/CPR is the realization that coupling between adjacent elements, when done in a discontinuous weak-Riemann fashion (§3.3.3), does not require a variational formulation of the problem; it instead can be applied, indirectly through a clever mechanism (the flux reconstruction/correction procedure, in this context), to the differential form of the PDE. This derivation of the method, given in §5.1.1, is consistent with both [55] (which is FR's original formulation) and [128] (in which both FR and CPR denominations are recognized as a unique method). Its ties to the general one introduced in §3.3, are highlighted in §5.1.4 and explored further in §5.3.

5.1. FR/CPR semi-discretization

The derivation of the FR/CPR semi-discrete hyperbolic conservation law is typically done starting from (2.1), the *strong formulation* (i.e. the differential form) of the problem. Regardless, solution discretization is exactly the same as for the general case (§3.2)—more specifically, in fact, it is the same as for (nodal) DGSEM: exact solution and flux vectors, $\boldsymbol{q}(t,x)$ and $\boldsymbol{f}(\boldsymbol{q})$, are respectively approximated as $\boldsymbol{q}^h(t,x) = \oplus_{k=1}^{\mathrm{K}} \boldsymbol{q}_k^h(t,x)$ and $\boldsymbol{f}^h(t,x) = \oplus_{k=1}^{\mathrm{K}} \boldsymbol{f}_k^h(t,x)$, in K piece-wise polynomial finite-dimensional spaces $\Omega_k \in \mathcal{T}^h$. Each component of the approximate solution and flux vectors is a polynomial of degree $\mathrm{p} \equiv \mathrm{J} - 1$; each element Ω_k has an associated J-dimensional trial function space $S_k^h\left(\widetilde{\Omega}\right)$, in which the aforementioned approximations exist.

The trial basis functions used in FR/CPR are, as in DGSEM, the *Lagrange polynomials*. An arbitrary element, Ω_k, contains J nodes (their number need not be the same for every element). The distribution of nodes within each element is commonly made according to either Gauss-Legendre or Gauss-Lobatto quadrature rules (some correction function may be more conveniently employed in one particular distribution). In this report, as with DGSEM in the previous chapter, only the former is considered. The degrees of freedom (i.e. nodal values) of flux and solution interpolants are, respectively (see §4.2.2) $\check{\boldsymbol{f}}_{jk} \coloneqq \boldsymbol{f}(\check{\boldsymbol{q}}_{jk})$ and $\check{\boldsymbol{q}}_{jk}$.

5.1.1. Differential formulation

Equation semi-discretization in FR/CPR starts from the strong statement of the general conservation law in differential form[1] (2.1)—as opposed to standard DG discretizations, which start from the weaker integral formulation, (2.9), which is enforced in a variational sense via test functions—under the premise that we wish to evaluate the term representing the divergence of the flux *directly*, that is, avoiding the use of calculus tools such as the divergence theorem or integration by parts.

For the approximate solution over a particular element Ω_k, (2.1) reads:

$$\frac{\partial \boldsymbol{q}_k^h}{\partial t} + \frac{\partial \boldsymbol{f}_k^h}{\partial x} = 0\,. \tag{5.1}$$

Two problems need to be overcome:

- The derivative of the flux is ill-defined at element interfaces, since $\boldsymbol{f}^h(x)$ will, in general, experience a discontinuity at such locations.
- There is no equation relating the solution on an element with that on another, since there is no coupling mechanism between elements.

The solution to both these issues consists on constructing a *continuous approximation* to the exact flux function, the **corrected** or **reconstructed flux**, which is the key ingredient of the FR/CPR framework:

$$\boldsymbol{f}(t,x) \simeq \boldsymbol{h}(t,x) = \bigoplus_{k=1}^{\mathrm{K}} \boldsymbol{h}_k(t,x)\,, \qquad \widetilde{\boldsymbol{h}}_k(t,\xi) = \sum_{r=1}^{\mathrm{J}+1} \check{\boldsymbol{h}}_{rk}(t) l_{r,k}(\xi)\,. \tag{5.2}$$

As with the regular flux vector, $\widetilde{\boldsymbol{h}}_k$ is the reference element version of $\boldsymbol{h}_k$. Notice that each h_{ik} is a polynomial of degree $p+1$; the reason for this will be made clear further on in the derivation. In fact, $\boldsymbol{h}_k$ has to satisfy three requirements:

1. h_{ik} is a polynomial of degree $\mathrm{J} \equiv p+1$, one higher than q_{ik}^h and f_{ik}^h.
2. h_{ik} approaches f_{ik}^h in some sense, i.e. $\left\| h_{ik} - f_{ik}^h \right\|$ is minimized in some norm.
3. $\boldsymbol{h}_k$ takes the value of the Riemann flux at each interface (see §3.3.3). As a consequence, each component of $\boldsymbol{h}$ is continuous over Ω (the entire domain).

The FR/CPR method reduces to using $\boldsymbol{h}_k$ instead of $\boldsymbol{f}_k^h$ in (5.1),

$$\frac{\partial \boldsymbol{q}_k^h}{\partial t} + \frac{\partial \boldsymbol{h}_k}{\partial x} = 0\,, \tag{5.3}$$

and (3.4) can be used to cast (5.3) into reference element coordinates. The result is:

$$\frac{\Delta x_k}{2} \frac{\partial \widetilde{\boldsymbol{q}}_k^h}{\partial t} + \frac{\partial \widetilde{\boldsymbol{h}}_k}{\partial \xi} = 0\,. \tag{5.4}$$

Given that $\widetilde{\boldsymbol{h}}_k$ and $\widetilde{\boldsymbol{q}}_k$ are Lagrange interpolants of different degree, it would seem that (5.4) requires using two different bases (and sets of nodes) for each element. Thanks to a clever definition of the former, we shall see that this is actually not the case.

5.1.2. Flux correction

In order to construct $\widetilde{\boldsymbol{h}}_k$ from known data, let us start by assigning to it a unique value at element interfaces (that of the Riemann flux, as mentioned previously):

$$\widetilde{\boldsymbol{h}}_k(t,-1) \equiv \widetilde{\boldsymbol{h}}_{k-1}(t,1) = \check{\boldsymbol{f}}_k^L(t)\,, \qquad \widetilde{\boldsymbol{h}}_k(t,1) \equiv \widetilde{\boldsymbol{h}}_{k+1}(t,-1) = \check{\boldsymbol{f}}_k^R(t)\,. \tag{5.5}$$

[1] FR/CPR can also be derived in its so-called *lifting collocation penalty formulation* as a particular case of the method of weighted residuals [125], thus starting from a weak formulation.

These equalities will be enforced by means of a pair of so-called *correction functions*, $g_L(\xi)$ and $g_R(\xi)$, associated with the left and right edges of the reference element $\widetilde{\Omega}$, respectively.

Consider the following (trivial) equalities:

$$\widetilde{\boldsymbol{h}}_k(t,\xi) = \widetilde{\boldsymbol{f}}^h_k(t,\xi) + \widetilde{\boldsymbol{h}}_k(t,\xi) - \widetilde{\boldsymbol{f}}^h_k(t,\xi)\,, \tag{5.6a}$$

$$\widetilde{\boldsymbol{h}}_k(t,1) = \widetilde{\boldsymbol{f}}^h_k(t,1) + \widetilde{\boldsymbol{h}}_k(t,1) - \widetilde{\boldsymbol{f}}^h_k(t,1)\,, \tag{5.6b}$$

$$\widetilde{\boldsymbol{h}}_k(t,-1) = \widetilde{\boldsymbol{f}}^h_k(t,-1) + \widetilde{\boldsymbol{h}}_k(t,-1) - \widetilde{\boldsymbol{f}}^h_k(t,-1)\,. \tag{5.6c}$$

The corrected flux function is defined by combining the three into one in such a way that the differences between corrected and uncorrected fluxes act as a weighting factors for the correction functions, as follows:

$$\widetilde{\boldsymbol{h}}_k(t,\xi) \coloneqq \widetilde{\boldsymbol{f}}^h_k(t,\xi) + \big(\widetilde{\boldsymbol{h}}_k(t,-1) - \widetilde{\boldsymbol{f}}^h_k(t,-1)\big)\, g_L(\xi) + \big(\widetilde{\boldsymbol{h}}_k(t,1) - \widetilde{\boldsymbol{f}}^h_k(t,1)\big)\, g_R(\xi)\,; \tag{5.7}$$

under the following constraints:

$$g_L\,(-1) = 1\,, \qquad g_L\,(1) = 0\,, \qquad g_R\,(-1) = 0\,, \qquad g_R\,(1) = 1\,, \tag{5.8}$$

and, because of a symmetry argument [55], it is generally possible to obtain the right correction function from the left one through a reflection about the origin:

$$g_R\,(\xi) = g_L\,(-\xi)\;, \qquad\qquad g'_R\,(\xi) = -g'_L\,(-\xi)\;.$$

These requirements ensure that $\boldsymbol{h}_k$ takes the Riemann flux values at the edges of Ω_k, at the same time that make it possible to separate each edge's contribution. The actual correction functions to employ are not uniquely defined by the previous constrains; they remain a design choice (as long as $p > 0$, see §5.3.2). Four well-known types of correction functions are detailed in §5.2.

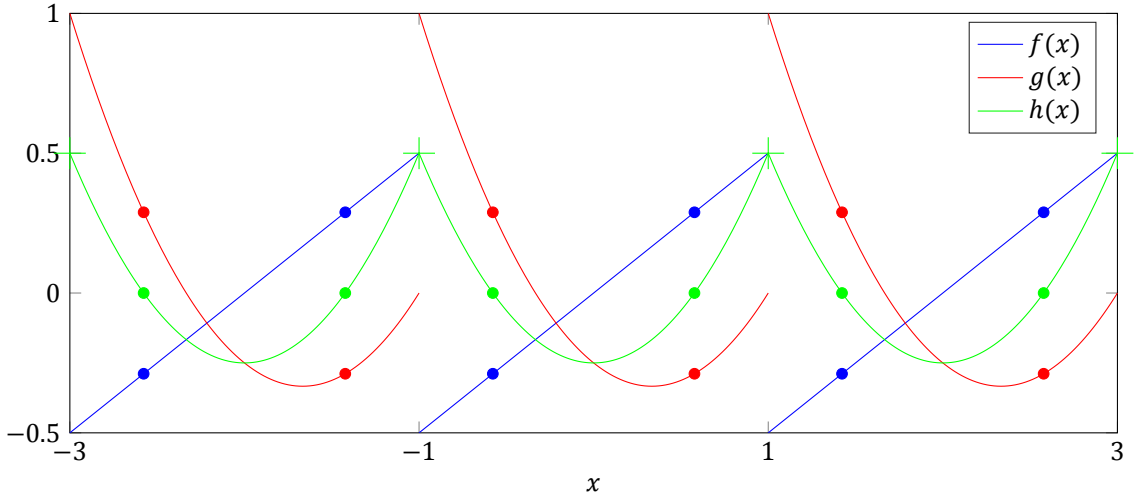


Figure 5.1: The correction procedure applied to three identical linear elements ($\Delta x = 2$). A continuous function h (green) is reconstructed from f (blue), which is discontinuous at element interfaces, by adding to it a correction function g (red). Dot markers represent Gauss-Legendre nodes.

5.1.3. Flux derivative

Application of the gradient operator to (5.7) results in:

$$\frac{\partial \widetilde{\boldsymbol{h}}_k}{\partial \xi} = \frac{\partial \widetilde{\boldsymbol{f}}^h_k}{\partial \xi} + \big(\widetilde{\boldsymbol{h}}_k(t,-1) - \widetilde{\boldsymbol{f}}^h_k(t,-1)\big)\, g'_L + \big(\widetilde{\boldsymbol{h}}_k(t,1) - \widetilde{\boldsymbol{f}}^h_k(t,1)\big)\, g'_R\,. \tag{5.9}$$

Given that the uncorrected flux vector components are Lagrange interpolants, their gradient can be conveniently computed at an arbitrary nodal location $\xi_n \in \Omega_k$ using the *derivative matrix* of the Lagrange

polynomials, $\boldsymbol{\mathcal{D}}$, as defined by (4.11):

$$\frac{\partial \tilde{\boldsymbol{f}}_k^h}{\partial \xi}(t,\xi_n) = \sum_{j=1}^{\mathrm{J}} \check{\boldsymbol{f}}_{jk}(t) d_{j,n} \tag{5.10}$$

The remaining terms in (5.9) are known; this includes the derivative of the correction function, which is the actual quantity defining a particular FR/CPR variant (as opposed to the correction function itself).

All FR/CPR results in this report have been obtained using (5.10), thus maintaining consistency with the other two methods reviewed. As an alternative, Wang and Gao [127] claim that the following (an application of the chain rule) may result in increased accuracy in some cases:

$$\frac{\partial \tilde{\boldsymbol{f}}_k^h}{\partial \xi}(t,\xi_n) = \boldsymbol{A}\left(\check{\boldsymbol{q}}_{nk}\right) \sum_{j=1}^{\mathrm{J}} \check{\boldsymbol{q}}_{jk} d_{j,n}\,, \tag{5.11}$$

where the Jacobian matrix $\boldsymbol{A}$ is evaluated at the state $\tilde{\boldsymbol{q}}_k^h(t,\xi_n) \equiv \check{\boldsymbol{q}}_{nk}(t)$. This second version of the uncorrected flux derivative term, however, I have not attempted.

All ingredients are now in place to evaluate (5.4) for the $\mathrm{p}+1$ nodal values of the approximate solution; this completes the spatial semi-discretization of the strong form of the conservation law.

5.1.4. Semi-discrete FR/CPR operators

Retaking the derivation back from (5.4), using (5.9) to evaluate the partial derivative of the corrected flux function, we obtain:

$$\frac{\Delta x_k}{2}\frac{\partial \tilde{\boldsymbol{q}}_k^h}{\partial t} + \frac{\partial \tilde{\boldsymbol{f}}_k^h}{\partial \xi} + \Delta\check{\boldsymbol{f}}_k^L g_L' + \Delta\check{\boldsymbol{f}}_k^R g_R' = 0\,, \tag{5.12}$$

where:

$$\Delta\check{\boldsymbol{f}}_k^L(t) \coloneqq \tilde{\boldsymbol{h}}_k(t,-1) - \tilde{\boldsymbol{f}}_k^h(t,-1) \equiv \check{\boldsymbol{f}}_k^L(t) - \tilde{\boldsymbol{f}}_k^h(t,-1)\,, \tag{5.13a}$$

$$\Delta\check{\boldsymbol{f}}_k^R(t) \coloneqq \tilde{\boldsymbol{h}}_k(t,1) - \tilde{\boldsymbol{f}}_k^h(t,1) \equiv \check{\boldsymbol{f}}_k^R(t) - \tilde{\boldsymbol{f}}_k^h(t,1) \tag{5.13b}$$

correspond to the differences between corrected and uncorrected fluxes at the boundaries of Ω_k.

It now becomes clear why the corrected flux function had to be defined as a polynomial of degree J: we are actually interested in its derivative; S_k^h being J-dimensional, it is most natural to consider a correction function such that $g'(\xi) \in S_k^h\left(\widetilde{\Omega}\right)$. Equation (5.12), using $g'(\boldsymbol{\xi}^\intercal) = \begin{bmatrix} g'(\xi_1) & g'(\xi_2) & \cdots & g'(\xi_{\mathrm{J}}) \end{bmatrix}$, may be rewritten in matrix form as:

$$\frac{\mathrm{d}\check{\boldsymbol{Q}}_k}{\mathrm{d}t} + \frac{2}{\Delta x_k}\left(\check{\boldsymbol{F}}_k\boldsymbol{\mathcal{D}} + \Delta\check{\boldsymbol{f}}_k^L g_L'(\boldsymbol{\xi}^\intercal) + \Delta\check{\boldsymbol{f}}_k^R g_R'(\boldsymbol{\xi}^\intercal)\right) = 0\,, \tag{5.14}$$

A term-by-term comparison between (5.14) and (3.23) reveals that the general formulation in §3.3.4 can be particularized to FR/CPR by setting the mass and gradient operators to:

$$\boldsymbol{\mathcal{M}}_k = \frac{\Delta x_k}{2}\boldsymbol{I}\,, \qquad\qquad \boldsymbol{\mathcal{C}}_k = \boldsymbol{\mathcal{D}}\,, \tag{5.15}$$

and replacing the element's net outward flux term, $\left[\check{\boldsymbol{f}}\boldsymbol{\varphi}_k^\intercal\right]_{\partial\Omega_k} = \check{\boldsymbol{f}}_k^R\boldsymbol{\varphi}_k^\intercal(1) - \check{\boldsymbol{f}}_k^L\boldsymbol{\varphi}_k^\intercal(-1)$, as follows:

$$\check{\boldsymbol{f}}_k^R\boldsymbol{\varphi}_k^\intercal(1) \leftarrow \Delta\check{\boldsymbol{f}}_k^R g_R'(\boldsymbol{\xi}^\intercal)\,, \qquad\qquad -\check{\boldsymbol{f}}_k^L\boldsymbol{\varphi}_k^\intercal(-1) \leftarrow \Delta\check{\boldsymbol{f}}_k^L g_L'(\boldsymbol{\xi}^\intercal)\,. \tag{5.16}$$

In relation to both (4.27) and (6.17), their FR/CPR counterparts (5.15) are indeed remarkably simple. Moreover, with its mass matrix being the identity, the conditioning of (5.14) does not depend on the discretization order or the nodal distribution chosen[2]. Also, no numerical integration has been necessary in any step of the derivation (nor evaluation) of the FR/CPR spatial residual operator.

[2] Nevertheless, it has been reported in the literature that using nodal locations matching the quadrature points of a "high-strength" quadrature rule is critical to achieve nonlinear stability [56, §5.4].

It should be noted that, in general, the extrapolation of the uncorrected flux function to a boundary is not equal to the flux function applied to the state extrapolated to that same boundary; that is: $\boldsymbol{f}^h(t,\pm 1) \neq \boldsymbol{f}\left(\boldsymbol{q}^h(t,\pm 1)\right)$. Therefore, when using a Gauss-Legendre nodal distribution, it will be necessary in FR/CPR to extrapolate *both* $\boldsymbol{q}_k^h$ and $\boldsymbol{f}_k^h$ at both edges of every $\Omega_k \in \mathcal{T}^h$, in order to evaluate (5.13). This could be avoided by employing e.g. Gauss-Lobatto nodes (any distribution in which a node is placed at each edge).

5.2. Correction functions

Specification of an actual correction function (or rather, its derivative at a set of nodes) is needed to fully define the FR/CPR discretization. In this section, the most common of such are listed—specifically, all definitions are given for the left correction function, $g_L(\xi)$. Accuracy and stability results reported in this chapter are taken from [55].

It is reasonable to assume that the optimal correction function is problem-dependent [55]; in this context, the flexibility of FR/CPR to produce various methods according to the choice of g should be seen as a relevant strength, as it allows tailoring the trade-off between accuracy and stability to each particular case. Huynh [55] originally put forward three kinds of correction functions. The two first ones reproduce, from within the FR/CPR framework, the discontinuous Galerkin and spectral difference schemes; the third, brings about a completely new family of methods. An additional forth kind was discovered by Vincent et al. [121], and is further explored in [122].

5.2.1. DG correction function

The first (left) correction function, denoted g_{DG}, is designed so that it exactly reproduces DGSEM; see §5.3. This happens to be the *right*[3] *Radau polynomial* of degree J, $R_R^{\mathrm{J}}(\xi)$, which can be evaluated using the Legendre ones (4.1) as:

$$g_{\mathrm{DG}} = R_R^{\mathrm{J}} \coloneqq \frac{(-1)^{\mathrm{J}}}{2}\left(\mathcal{P}_{\mathrm{J}} - \mathcal{P}_{\mathrm{J}-1}\right) . \tag{5.17}$$

Taking the gradient of (5.17), leads to:

$$g'_{\mathrm{DG}} = \frac{(-1)^{\mathrm{J}}}{2}\left(\mathcal{P}'_{\mathrm{J}} - \mathcal{P}'_{\mathrm{J}-1}\right) , \tag{5.18}$$

with the following simplified expressions for its values at the edges of $\widetilde{\Omega}$:

$$g'_{\mathrm{DG}}(-1) = -\frac{\mathrm{J}^2}{2} , \qquad g'_{\mathrm{DG}}(1) = -(-1)^{\mathrm{J}}\frac{\mathrm{J}}{2} . \tag{5.19}$$

This version of FR/CPR achieves the highest formal order of accuracy among all known alternatives (see table 5.1). Moreover, under Fourier analysis, it seems to possess favorable stability characteristics up to arbitrarily high order. Its maximum allowable time-step size, however, is relatively small [55, figures 6.3, 6.5 and 6.7].

5.2.2. SG/SD correction function

A second correction function, g_{SG}, recovers a simplified version of the *staggered grid* (SG) scheme, also known as *spectral difference* (SD). Its FR/CPR version has the advantage over SG's original form of requiring only one grid. Compared with DG, this method has a slightly larger allowable time-step size, but at the cost of losing DG's alleged super-accuracy and being mildly unstable for all degrees [55].

This correction function is defined as the J-th degree polynomial that, given a set of $\mathrm{J}+1$ *Chebyshev* quadrature points (see [66, section 3.2.3]):

$$\xi_m = -\cos\left(\frac{m}{\mathrm{J}}\pi\right), \quad m = 0, 1, \ldots, \mathrm{J} , \tag{5.20}$$

interpolates over the values:

$$g_{\mathrm{SG}}(\xi_m) = \begin{cases} 1 & \text{if } m = 0 \\ 0 & \text{otherwise} \end{cases} . \tag{5.21}$$

[3] Note that the *right* Radau polynomial is the *left* correction function, and vice-versa.

5.2.3. Huynh's correction functions

Many correction functions can be defined as interpolants over a given set of values. The sample locations of these (as in the previous example) have nothing to do with the nodes of the discretization.

In addition to the two previous examples, Huynh [55] experimented with several other correction functions, leading to new methods. Among these, one interesting example is:

$$g_2 = \frac{(\mathrm{J}-1)R_R^{\mathrm{J}} + \mathrm{J}R_R^{\mathrm{J}-1}}{2\mathrm{J}-1}, \quad \mathrm{J} > 1\,. \tag{5.22}$$

This correction function is obtained if, in addition to enforcing $g_L(-1) = 1$ and $g_L(1) = 0$, a zero *for its derivative* is also specified at $\xi = 1$: $g_L'(1) = 0$ (hence its designation as g_2).

A remarkable feature of this left correction function is that its derivative is zero at all but the first (left-most) node of the J-point Gauss-Lobatto quadrature rule. That is, for a sequence $-1 < \xi_2 < \cdots < \xi_{\mathrm{J}-1} < 1$ of J Gauss-Lobatto quadrature points:

$$g_2'(\xi_m) = \begin{cases} g_2'(-1) = (1-\mathrm{J})\frac{\mathrm{J}}{2} & \text{if } m = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{5.23}$$

Huynh [55] alternatively uses the designation $g_{\text{Lump,Lo}}$ for this function, meaning that all the effect of the correction is *lumped* (i.e. concentrated) to its edge's Lobatto node (as it is zero at all others). An FR/CPR implementation that uses this correction function in conjunction with a Gauss-Lobatto distribution of J solution nodes can, therefore, be particularly simple and economical. Additionally, such as scheme is found to be stable for all degrees and to allow a time-step size twice as large as that of DG, with its accuracy being reduced only by one order (in optimal conditions).

Analogous correction functions can be designed using other point distributions, e.g. Chebyshev ($g_{\text{Lump,Ch}}$). Extending the procedure that led to g_{SG}, new schemes of the SD/SG kind can be obtained by directly selecting the $\mathrm{J}-1$ additional conditions that its correction function should satisfy (recall that g_L and g_R are polynomials of degree J and, therefore, require $\mathrm{J}+1$ constraints to be unique—two of which have already been imposed). For example, let g_{Ga} be defined as the interpolant for which:

$$g_{\text{Ga}}(\xi) = \begin{cases} 1 & \text{if } \xi = -1 \\ 0 & \text{if } \xi = \xi_m \\ 0 & \text{if } \xi = 1 \end{cases}, \tag{5.24}$$

where $\xi_m \in \{\xi_1, \xi_2, \ldots, \xi_{\mathrm{J}-1}\}$, in this case the set of $\mathrm{J}-1$ Gauss-Legendre quadrature points extending over the reference element $\widetilde{\Omega}$. Analogously, if instead of the Gauss-Legendre points one would use Gauss-Lobatto ones, a new correction function would arise: g_{Lo}. A comparative summary of the main features of all correction functions mentioned so far is given in table 5.1.

Correction function	Max. accuracy order	Max. $\frac{\Delta t}{\Delta t_{\text{DG}}}$	Linearly stable?
g_{DG}	$2\mathrm{p}+1$	1	✓
g_{Ga}	$2\mathrm{p}$	≈ 1.5	✓
$g_2 \equiv g_{\text{Lump,Lo}}$	$2\mathrm{p}$	≈ 2	✓
$g_{\text{Lump,Ch}}$	$\mathrm{p}+1$	–	✓
g_{SG}	$\mathrm{p}+1$	≈ 1.6	✗
g_{Lo}	$\mathrm{p}+1$	–	✗

Table 5.1: Comparative summary of the properties of various FR/CPR variants when applied to the linear advection equation (2.19) with a degree p approximation, according to the correction function employed. Based on Fourier analysis results reported in [55]. Orders of accuracy were obtained from (A.65). Allowable time-step size ratios shown correspond to the RK3(3) scheme and $\mathrm{p} = 3$.

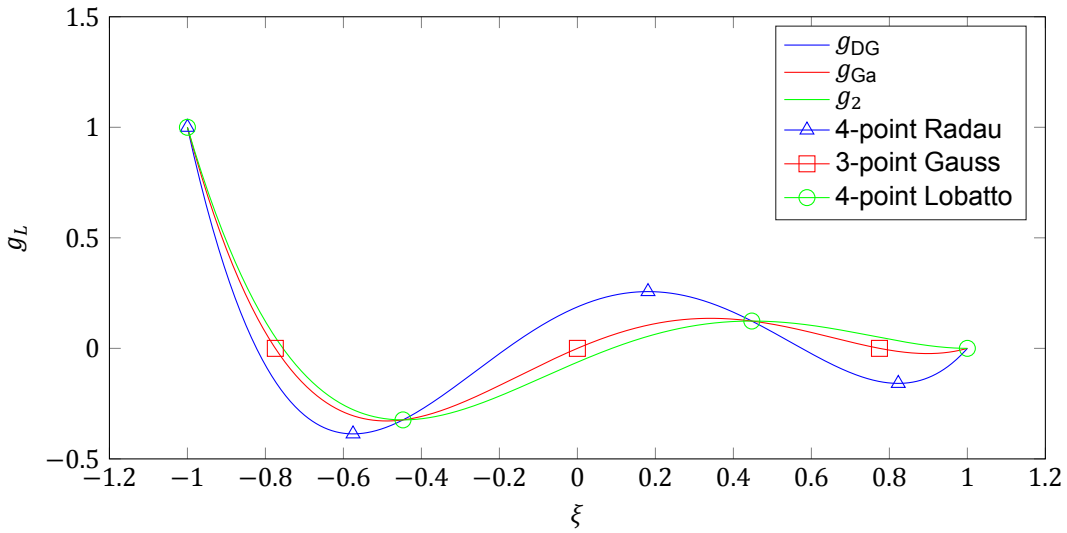


Figure 5.2: Cubic (left) correction functions—quadratic approximate solution and flux—discovered by Huynh [55]. Notice that each kind of correction function is associated to certain distribution of nodes; these are either its interior zeros (g_{Ga}) or its local extrema.

5.2.4. Energy-stable correction functions

Vincent et al. [121] introduced an entire family of correction functions parameterized with a single scalar quantity, c. These are now known as VCJH-type correction functions. By design, all g_{VCJH} functions lead to *linearly stable* schemes, provided that $c_- < c < c_\infty$ (and adequate time-step size restrictions); the lower bound of this interval, which depends on the polynomial degree of the approximate solution, is given by:

$$c_- = \frac{-2}{(2\mathrm{p}+1)\left(a_{\mathrm{p}}\mathrm{p}!\right)^2}\,, \tag{5.25}$$

where a_{p} is the coefficient of the leading monomial term of the Legendre polynomial of degree p:

$$a_{\mathrm{p}} = \frac{(2\mathrm{p})!}{2^{\mathrm{p}}\left(\mathrm{p}!\right)^2}\,. \tag{5.26}$$

There is no finite upper bound for c, i.e. $c_\infty \to \infty$. Note that $c_- < 0$ (see table 5.2). On the opposite end of its range, it turns out that c_∞ results in a valid FR/CPR scheme [121], g_∞ being equivalent to g_{DG} of degree p (i.e. lowered by one)—see figure 5.3.

For every value of c, left and right VCJH-type correction functions of degree $\mathrm{p}+1 > 1$ can be constructed from the following general definition:

$$g_L = \frac{(-1)^{\mathrm{p}}}{2}\left(\mathcal{P}_{\mathrm{p}} - \frac{\eta\mathcal{P}_{\mathrm{p}-1} + \mathcal{P}_{\mathrm{p}+1}}{1+\eta}\right), \qquad g_R = \frac{1}{2}\left(\mathcal{P}_{\mathrm{p}} + \frac{\eta\mathcal{P}_{\mathrm{p}-1} + \mathcal{P}_{\mathrm{p}+1}}{1+\eta}\right), \tag{5.27}$$

where:

$$\eta \coloneqq \frac{c\,(2\mathrm{p}+1)\left(a_{\mathrm{p}}\mathrm{p}!\right)^2}{2}\,. \tag{5.28}$$

Particular values for c that recover the most interesting of the functions studied by Huynh [55] are:

$$c_{\mathrm{DG}} \coloneqq 0\,, \qquad c_{\mathrm{Ga}} \coloneqq \frac{2\mathrm{p}}{(2\mathrm{p}+1)\,(\mathrm{p}+1)\left(a_{\mathrm{p}}\mathrm{p}!\right)^2}\,, \qquad c_2 \coloneqq \frac{2\,(\mathrm{p}+1)}{(2\mathrm{p}+1)\,\mathrm{p}\left(a_{\mathrm{p}}\mathrm{p}!\right)^2}\,; \tag{5.29}$$

or, in terms of η:

$$\eta_- = -1\,, \qquad \eta_{\mathrm{DG}} = 0\,, \qquad \eta_{\mathrm{Ga}} = \frac{\mathrm{p}}{\mathrm{p}+1}\,, \qquad \eta_2 = \frac{\mathrm{p}+1}{\mathrm{p}}\,, \qquad \eta_\infty = \infty\,. \tag{5.30}$$

This type of correction functions has later been found to be contained in an even larger set, sharing the same favorable stability properties [122]. It should be pointed out that the definition of the VCJH family of correction functions (5.27) breaks down for $p = 0$. In that case, only one correction function satisfies (5.8) (see §5.3.2): $g_L = (1 - \xi)/2$.

p	c_-	c_{DG}	c_{Ga}	c_2
1	$-2/3 \approx -0.667$	0	$1/3 \approx 0.333$	$4/3 \approx 1.333$
2	$-2/45 \approx -4.444 \times 10^{-2}$	0	$4/135 \approx 2.963 \times 10^{-2}$	$1/15 \approx 6.667 \times 10^{-2}$
3	$-2/1575 \approx -1.270 \times 10^{-3}$	0	$1/1050 \approx 9.524 \times 10^{-4}$	$8/4725 \approx 1.693 \times 10^{-3}$
4	$-2/99\,225 \approx -2.016 \times 10^{-5}$	0	$8/496\,125 \approx 1.612 \times 10^{-5}$	$1/39\,690 \approx 2.520 \times 10^{-5}$
5	$-2/9\,823\,275 \approx -2.036 \times 10^{-7}$	0	$1/5\,893\,965 \approx 1.697 \times 10^{-7}$	$4/16\,372\,125 \approx 2.443 \times 10^{-7}$

Table 5.2: Value of the c parameter for the main FR/CPR correction functions, as the degree increases.

5.3. FR/CPR and the discontinuous Galerkin method

Huynh [55] proved, as already mentioned, that the choice of g_{DG} as correction function will result in a scheme that is identical to the Lagrange polynomial-based (i.e. nodal) DG method. Such a method is identical, in turn, to DGSEM as presented in chapter 4; this is because both it and FR/CPR (in this report) employ Gauss-Legendre nodes.

The reverse is not exactly true; however, it is made manifest in §5.1.4 that—leaving out the net boundary flux term, in which correction functions play their role—the mass and discrete gradient matrices are consistent with the general formulation in 3.3. In fact, this sets some requirements on the set of test functions that would result in said FR/CPR semi-discrete operators. Specifically, for all j, φ_{nk} satisfies:

$$\int_{-1}^{1} l_j \varphi_{nk} \, \mathrm{d}\xi = l_j(\xi_n) \,, \qquad \int_{-1}^{1} l_j' \varphi_{nk} \, \mathrm{d}\xi = l_j'(\xi_n) \,. \tag{5.31}$$

These suggest that $\varphi_{nk} = \delta(\xi - \xi_n)$, *Dirac's Delta function* centered at node ξ_n, for which, by definition:

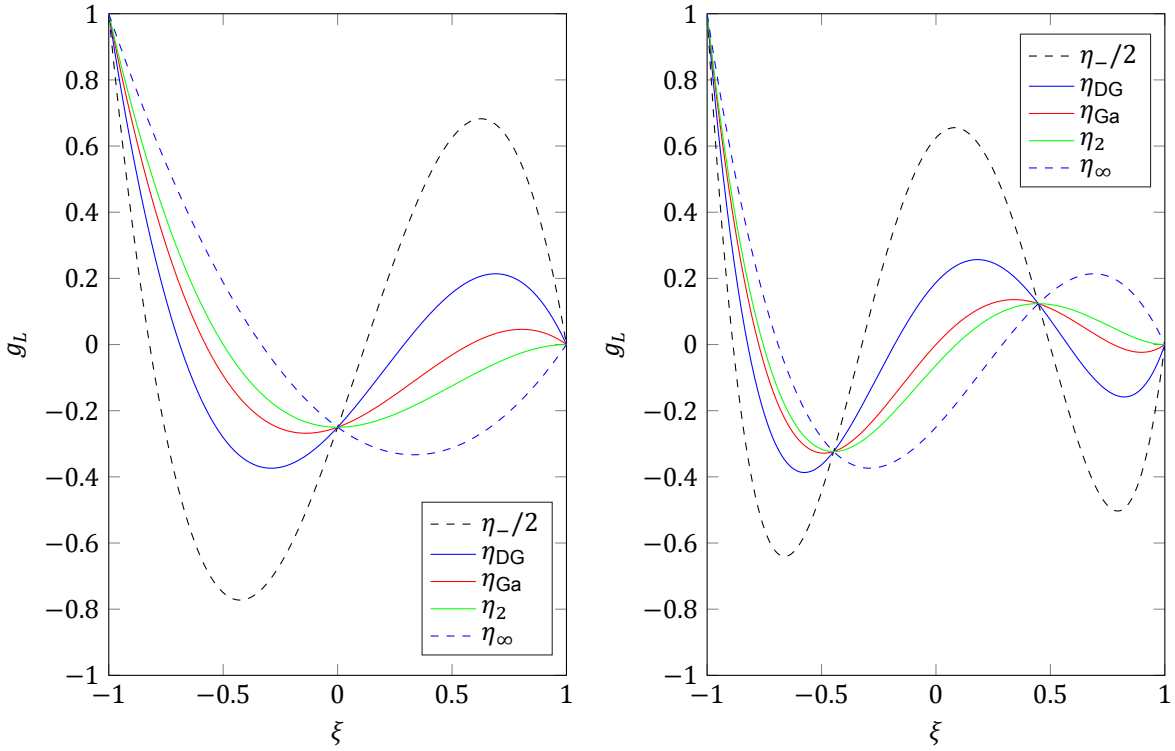
$$\delta(\xi - \xi_n) = 0, \quad \xi \neq \xi_n \,, \qquad \int_{\xi_n - \varepsilon}^{\xi_n + \varepsilon} f \delta(\xi - \xi_n) \, \mathrm{d}\xi = f(\xi_n), \quad \varepsilon > 0 \tag{5.32}$$

for *any* function $f : \mathbb{R} \to \mathbb{R}$ [6, §1.15].

My interpretation of this result is that the discrete version of the conservation law is being prescribed at each node only. However, any Lagrange interpolant is uniquely defined by its nodal values; therefore, the approximate solution is nevertheless still identical to that of DGSEM (assuming that the adequate correction function is used as well), provided that both approximate solutions are initially the same (see §5.3.1). This idea of FR/CPR seen as a non-standard application of the method of weighted residuals is briefly explored in [55].

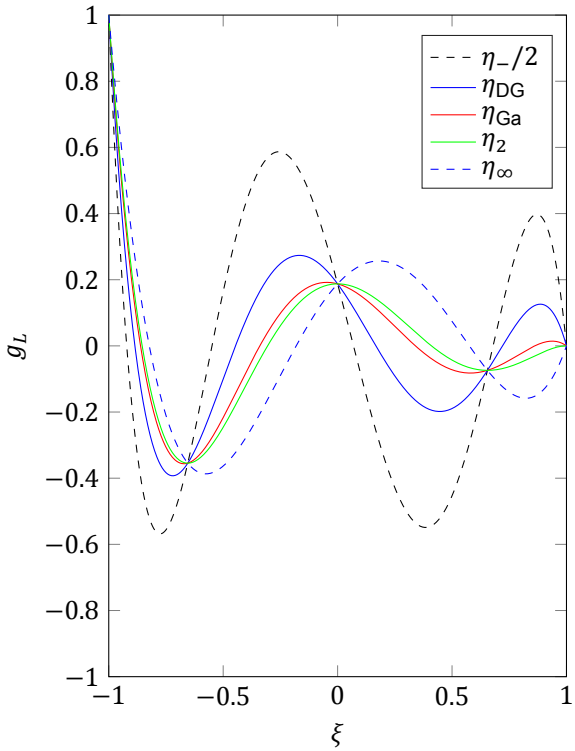
5.3.1. Initialization

As mentioned in §5.3, the semi-discrete conservation law obtained using FR/CPR's version of DG is identical to that of DGSEM. However, the approximate solution obtained with these two methods will generally be identical at a given instant $t > t_0$ only if it starts from an identical numerical initial condition. The problem is that, strictly following the general initialization procedure described in §3.5—i.e. using (3.34) with the mass matrix defined for FR/CPR by (5.15) and Dirac delta test functions (see §5.3)—

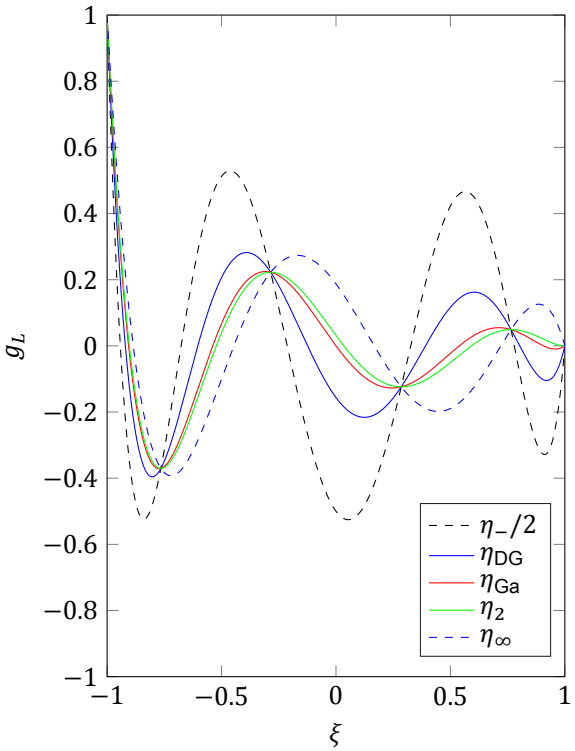


(a) Quadratic (p = 1 ⇔ J = 2)

(b) Cubic (p = 2 ⇔ J = 3)



(c) Quartic (p = 3 ⇔ J = 4)



(d) Quintic (p = 4 ⇔ J = 5)

Figure 5.3: Some energy-stable (left) correction functions.

would result in an interpolatory (rather than L^2) projection:

$$\int_{-1}^{1} q_i^0\left(\mathcal{X}_k(\xi)\right)\delta(\xi-\xi_n)\,\mathrm{d}\xi = \int_{-1}^{1}\sum_{j=1}^{\mathrm{J}}\check{q}_{ijk}l_j(\xi)\delta(\xi-\xi_n)\,\mathrm{d}\xi \implies \check{q}_{ink} = q_i^0(\mathcal{X}_k(\xi_n))\,. \tag{5.33}$$

In general, the polynomial that interpolates the exact initial solution at a given set of nodes is not the same as that which minimizes the L^2 norm of the error between it and that same exact solution. Therefore, in order to establish a fair framework in which to compare experimental results, all FR/CPR approximate initial conditions shown in this report are obtained by interpolating its DGSEM counterpart (of the same degree). This is equivalent to *momentarily* (only when projecting the initial condition) using the Lagrange polynomials as test basis functions in (3.33).

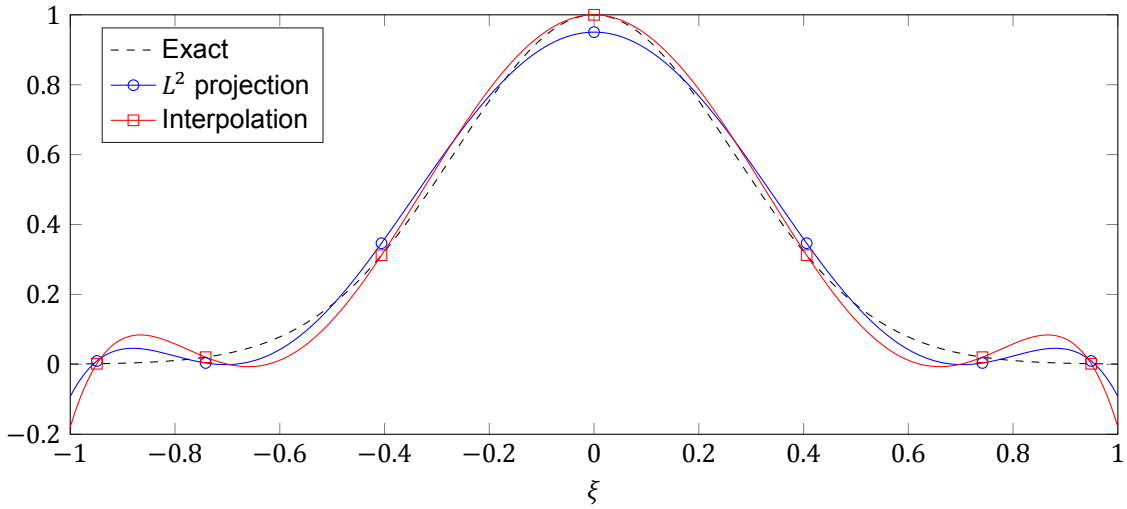


Figure 5.4: Comparison between interpolatory (red) and L^2-norm preserving (blue) approximations of a Gaussian hump, on a polynomial finite-element of degree $\mathrm{p} = 6$. Markers denote nodes (Gauss-Legendre).

5.3.2. FR/CPR and the finite volume method

It is worth considering the particular case of an FR/CPR discretization with $p = 0$, i.e. zero degree approximation. Because this translates in assuming the solution polynomials to be simply constants, one would expect this case to reduce to the first order finite volume method (as is the case with DGSEM); it can be easily proven that this is indeed true.

Proof. Since the solution is approximated as a constant, the correction functions are linear polynomials. Each of these is, therefore, uniquely defined by its two edge constraints: $g_L(-1) = 1, g_L(1) = 0$ and conversely for g_R. The uncorrected flux is approximated with the same degree as the solution; hence, it is also a constant and its derivative is zero. Therefore, the term associated with its derivative vanishes from equation 5.12:

$$\frac{\partial\tilde{\boldsymbol{f}}_k^h}{\partial\xi} = 0 \implies \frac{\Delta x_k}{2}\frac{\mathrm{d}\tilde{\boldsymbol{q}}_k^h}{\mathrm{d}t} + \Delta\check{\boldsymbol{f}}_k^L g_L' + \Delta\check{\boldsymbol{f}}_k^R g_R' = 0\,. \tag{5.34}$$

The derivatives of g_L and g_R, these being linear functions subject to (5.8), necessarily have to be:

$$g_L' = -\frac{1}{2}\,, \qquad\qquad g_R' = \frac{1}{2}\,. \tag{5.35}$$

Equation (5.14) becomes:

$$\frac{\Delta x_k}{2}\frac{\mathrm{d}\tilde{\boldsymbol{q}}_k^h}{\mathrm{d}t} + \frac{1}{2}\left(-\check{\boldsymbol{f}}_k^L + \tilde{\boldsymbol{f}}_k^h(-1) + \check{\boldsymbol{f}}_k^R - \tilde{\boldsymbol{f}}_k^h(1)\right) = 0\,, \tag{5.36}$$

but the uncorrected flux is also approximated as a constant, so $\tilde{\boldsymbol{f}}_k^h(-1) = \tilde{\boldsymbol{f}}_k^h(1)$. Therefore, (5.14) turns out to be:

$$\Delta x_k \frac{\mathrm{d}\tilde{\boldsymbol{q}}_k^h}{\mathrm{d}t} + \check{\boldsymbol{f}}_k^R - \check{\boldsymbol{f}}_k^L = 0\,, \tag{5.37}$$

which is identical to the semi-discrete form of the finite volume method. □

6

Isogeometric Analysis (IGA)

Isogeometric analysis was first proposed in [54] and further established in [28]. In essence, it consists on using the basis employed by the geometrical representation of a domain (e.g. in a CAD program) to approximate the solution fields on it. It can be seen as an evolution of the *isoparametric* concept of classical finite element analysis (FEA) [28, §3.1]. According to the literature, IGA has already demonstrated superior accuracy to traditional FEA in several problems, including (incompressible) turbulent flows [28, §9.4].

The last research object considered in the present work is a DG method employing B-splines as basis functions, to which I will refer as discontinuous galerkin isogeometric analysis (DGIGA). Isogeometric high-order methods have been applied to hyperbolic conservation laws only recently, in [57, 90] and [31, 32]. Given that the geometric advantages of an IGA formulation of DG cannot be explored in one dimension, I focus in this thesis on the effects of using NURBS—actually, and in particular, B-spline—basis functions, as opposed to the more conventional Lagrange polynomials of the previous two methods.

6.1. Basis splines (B-splines)

A B-spline is a piecewise-polynomial curve of degree p. The locations (in reference coordinates) where a polynomial segment starts or ends are called *breakpoints* [97, p. 51]. I will refer in this report to the intervals between to breakpoints as *breakpoint spans*; consequently, any B-spline is C^∞ within each of its breakpoint spans. The domain where a B-spline is defined is called *patch*. Any B-spline basis function (in 1D) will therefore be associated with at least two breakpoints: the edges of the patch.

6.1.1. Knot vector

A sequence of nondecreasing (possibly repeated) real values defines a so-called *knot vector*:

$$\Xi \coloneqq \{\xi_1, \xi_2, \dots, \xi_{\mathrm{J}+\mathrm{p}+1}\}, \tag{6.1}$$

where each $\xi_l \in \mathbb{R}$ is a *knot*, and every half-open interval $\Sigma_l \coloneqq [\xi_l, \xi_{l+1})$ is a *knot span*. The set of distinct knot values equals the set of breakpoints, as defined above; likewise, every breakpoint span corresponds to a knot span of non-zero size.

Knot vectors can be *uniform* or *nonuniform*, depending on whether knots are equidistantly spaced (in reference coordinates) or not. They can also be either *open*[1] or *closed*, the former being those for which the first and last knots have multiplicity $\mathrm{m} = \mathrm{p} + 1$; all knot vectors that appear in this report are open.

6.1.2. Basis functions

An open knot vector of length $\mathrm{J} + \mathrm{p} + 1$ defines the set $\{N_1^\mathrm{p}, \dots, N_\mathrm{J}^\mathrm{p}\}$ of linearly independent B-spline functions, each of piecewise-polynomial degree p, which constitute a basis. Evaluation of $N_j^\mathrm{p} : \mathbb{R} \to \mathbb{R}$

[1] Open knot vectors are also referred to as nonperiodic or clamped in the literature.

can be carried out via:

$$N_j^0(\xi) = \begin{cases} 1 & \text{if } \xi \in \Sigma_j \\ 0 & \text{otherwise} \end{cases}, \tag{6.2a}$$

$$N_j^{\mathrm{p}}(\xi) = \frac{\xi - \xi_j}{\xi_{j+\mathrm{p}} - \xi_j} N_j^{\mathrm{p}-1}(\xi) + \frac{\xi_{j+\mathrm{p}+1} - \xi}{\xi_{j+\mathrm{p}+1} - \xi_{j+1}} N_{j+1}^{\mathrm{p}-1}(\xi)\,, \tag{6.2b}$$

which is known as the *Cox-de Boor recursive formula* (see figure 6.1). Derivatives of B-spline basis functions can also be computed recursively. First derivatives are given by [97, §2.3]:

$$\left(N_j^{\mathrm{p}}\right)' = \frac{\mathrm{p}}{\xi_{j+\mathrm{p}} - \xi_j} N_j^{\mathrm{p}-1} - \frac{\mathrm{p}}{\xi_{j+\mathrm{p}+1} - \xi_{j+1}} N_{j+1}^{\mathrm{p}-1}\,. \tag{6.3}$$

A trivial generalization of (6.3) to an arbitrary κ-th derivative is:

$$\left(N_j^{\mathrm{p}}\right)^{(\kappa)} = \frac{\mathrm{p}}{\xi_{j+\mathrm{p}} - \xi_j} \left(N_j^{\mathrm{p}-1}\right)^{(\kappa-1)} - \frac{\mathrm{p}}{\xi_{j+\mathrm{p}+1} - \xi_{j+1}} \left(N_{j+1}^{\mathrm{p}-1}\right)^{(\kappa-1)}\,. \tag{6.4}$$

Equations (6.2), (6.3) and (6.4) can all be implemented efficiently[2] following the guidelines in [97, §2.5].

This basis has a number of interesting properties, listed here without proof (refer to [97, §§ 2.2, 2.4]). Note that orthogonality is *not* one of them.

Property 6.1 (local support). Every basis function has nonzero support over $\mathrm{p}+1$ knot spans, more precisely: $N_j^{\mathrm{p}}(\xi) = 0$ for all $\xi \notin \left[\xi_j, \xi_{j+\mathrm{p}+1}\right)$. As a consequence, each basis function shares support with, at most, $2\mathrm{p}+1$ functions (up to p on each side, plus itself), regardless of knot multiplicities. This implies that discrete IGA operators (e.g. mass matrix) have the same bandwidth as those of classical FEA [28, p. 22].

Property 6.2. Given a knot span Σ_l, only the functions $N_{l-\mathrm{p}}^{\mathrm{p}}, \ldots, N_l^{\mathrm{p}}$ (if defined) have nonzero support on it, i.e. a maximum of $\mathrm{p}+1$ basis functions. If Σ_l is a break span, exactly $\mathrm{p}+1$ basis functions are nonzero over it.

Property 6.3 (nonnegativity). $N_j^{\mathrm{p}}(\xi) \geq 0$ for all j, p, ξ.

Property 6.4 (partition of unity). $\sum_{j=l-\mathrm{p}}^{l} N_j^{\mathrm{p}}(\xi) = 1$ for all $\xi \in \Sigma_l$, i.e. the addition of all basis functions with support on a knot span is unity at any point within that knot span.

Property 6.5. All derivatives of a basis function exists within a knot span, i.e. $N_j^{\mathrm{p}}(\xi) \in C^\infty$ for all $\xi \in (\xi_l, \xi_{l+1})$. At a breakpoint, $N_j^{\mathrm{p}} \in C^{\mathrm{p}-\mathrm{m}}$ (m is the number of knots sharing that breakpoint's location); in particular, open knot vectors impose C^{-1} continuity at patch edges. For the same reason, continuity can be at most $C^{\mathrm{p}-1}$ across breakpoint spans. Increasing the degree increases continuity (smoothness), while increasing knot multiplicity reduces it.

Property 6.6. Any basis function of degree $\mathrm{p} > 0$ has exactly one maximum.

6.1.3. B-spline curves

B-splines are used in CAD software to generate complex geometric shapes (curves, surfaces and solids) in two or three-dimensional space. Their interpretation as geometric entities is quite intuitive, and so I use it here to facilitate the introduction of the main elements that will later be "repurposed" to construct an IGA discretization (§6.3). Higher-dimensional B-spline shapes, i.e. surfaces and solids, can be built via tensor products between univariate B-spline bases [28, §2.1.3]; only said univariate (i.e. single knot vector) case is considered in this report. A set of B-spline basis functions obtained from (6.2) may be used to generate parametric curves $\boldsymbol{c}\colon \mathbb{R} \to \mathbb{R}^d$ via linear combination:

$$\boldsymbol{c}(\xi) \coloneqq \sum_{j=1}^{\mathrm{J}} \boldsymbol{b}_j N_j^{\mathrm{p}}(\xi), \quad \xi_1 \leq \xi \leq \xi_{\mathrm{J}+1}\,. \tag{6.5}$$

[2] It is possible to evaluate these expressions as is, if one takes the precaution of replacing any indetermination of the type $\frac{0}{0}$ by 0; a well-designed algorithm avoids these terms altogether.

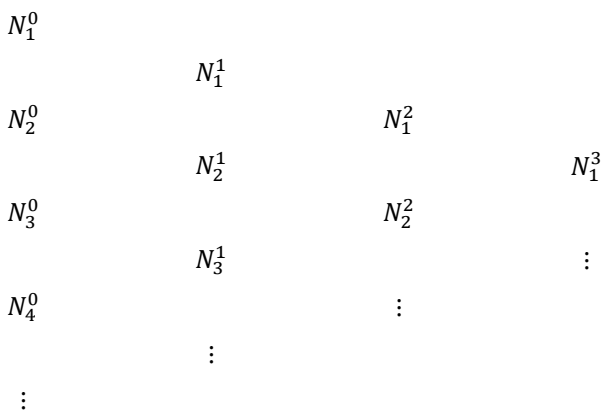


Figure 6.1: Tabular representation of (6.2): a given basis function is generated via a linear combination of its upper-left and lower-left neighbors in this arrangement; its domain of nonzero support is the union of these two's respective ones.

Thus, $\boldsymbol{c}(\xi)$ is a *one-dimensional shape* in a d*-dimensional space*, parameterized by ξ, and each of its d components exists in a J-dimensional function space spanned by the B-spline basis functions associated to some given knot vector.

Each of the J vectors $\boldsymbol{b}_j \in \mathbb{R}^d$ is given the name *control point*; the entire set of them defines the *control polygon* associated to $\boldsymbol{c}(\xi)$. These control points (in combination with the set of knots) generalize the role of nodes in a Lagrange polynomial expansion, in the sense that they represent the expansion coefficients of the basis while being associated to a particular location. With a B-spline basis, however, the B-spline curve—itself analogous to the Lagrange interpolant—does *not* interpolate over the control point positions in general; qualitatively, a control point "tends to pull" the B-spline curve towards it, but may fail to do so all the way to intersection.

A number of properties of B-spline curves follow from those of their basis functions [97, §3.2], namely:

Property 6.7. The B-spline curve $\boldsymbol{c}(\xi)$ is piecewise-polynomial of degree p; it requires J control points (one per basis function) and a knot vector of length $\mathrm{J}+\mathrm{p}+1$.

Property 6.8 (endpoint interpolation). The curve intersects its first and last control points i.e.: $\boldsymbol{c}(\xi_1) = \boldsymbol{b}_1$ and $\boldsymbol{c}\left(\xi_{\mathrm{J}+\mathrm{p}+1}\right) = \boldsymbol{b}_{\mathrm{J}}$.

Property 6.9 (affine invariance). Any affine transformation can be applied to the curve indirectly, by applying it to its control points.

Property 6.10 (strong convex hull). The curve is contained in the convex hull of its control polygon. More specifically, the portion of $\boldsymbol{c}(\xi)$ for $\xi \in \Sigma_l$ such that $\mathrm{p} < l < \mathrm{J}$, is in the convex hull of the $\boldsymbol{b}_{l-\mathrm{p}}, \ldots, \boldsymbol{b}_{l+1}$ control points.

Property 6.11 (local modification scheme). A change in $\boldsymbol{b}_j$ can only affect the shape of the curve in the interval $\xi \in \left[\xi_j, \xi_{j+\mathrm{p}+1}\right)$. This is a consequence of the local support of the basis.

Property 6.12. The B-spline curve can be regarded as an approximation to its control polygon. The higher the degree, the *worse* this approximation becomes—with $\mathrm{p} = 1$ being exact. Note, however, that the entity of interest is usually not the control polygon but the curve.

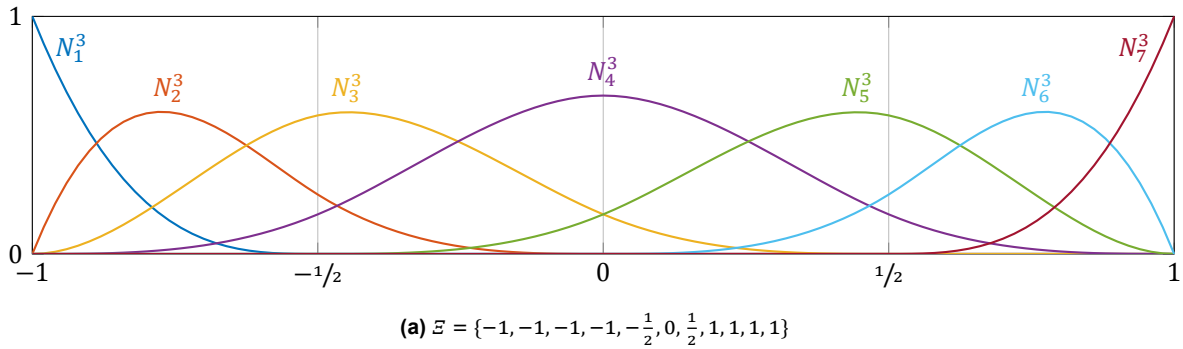
Property 6.13 (variation diminishing). No line (if $d = 2$; plane if $d = 3$) can have more intersections with a B-spline curve than with its control polygon. This is illustrated in figure 6.4.

6.2. Related bases

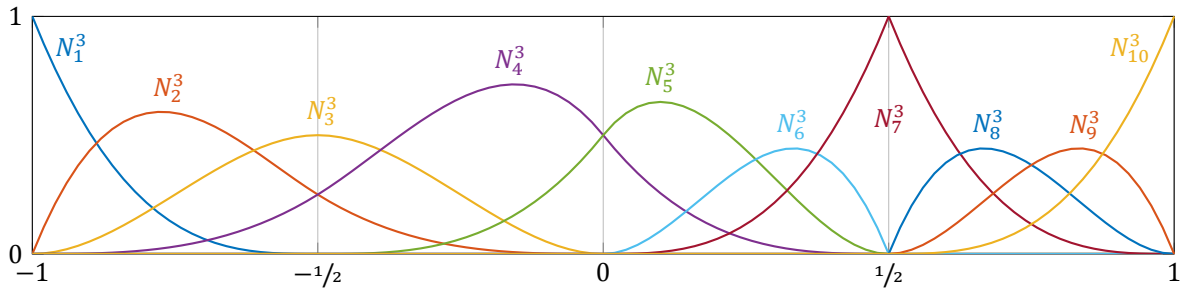
It is insightful to compare B-splines to other sets of basis functions. This might help us to identify critical differences with the previous methods, as well as possible equivalences.

6.2.1. Non-uniform rational B-splines (NURBS)

NURBS were the original basis functions of choice for IGA, due to their prevalence in geometry generation software used in the computer aided engineering (CAE) industry [28, §1.1.2]. They are obtained



(a) $\Xi = \{-1, -1, -1, -1, -\frac{1}{2}, 0, \frac{1}{2}, 1, 1, 1, 1\}$



(b) $\Xi = \{-1, -1, -1, -1, -\frac{1}{2}, 0, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1, 1\}$

Figure 6.2: Examples of cubic B-spline function bases for the reference patch.

via weighted rational combinations of B-splines, in such a way that any B-spline can be seen as a particular case of NURBS. The advantage of NURBS over non-rational B-splines is that they allow exact representation of analytic shapes beyond those amenable to polynomials, namely conics (in 2D) and quadrics (in 3D), while at the same time allowing representation of free-form shapes. It is debatable whether such a feature justifies the added complexity over non-rational B-splines, even in CAD applications [98]. More recently developed alternatives (e.g. T-splines) might prove particularly advantageous in IGA [28, §13]. In any case, rational B-splines are left outside the scope of the current study; more details on them can be found in [28, 97].

6.2.2. Bernstein polynomials and Bézier curves

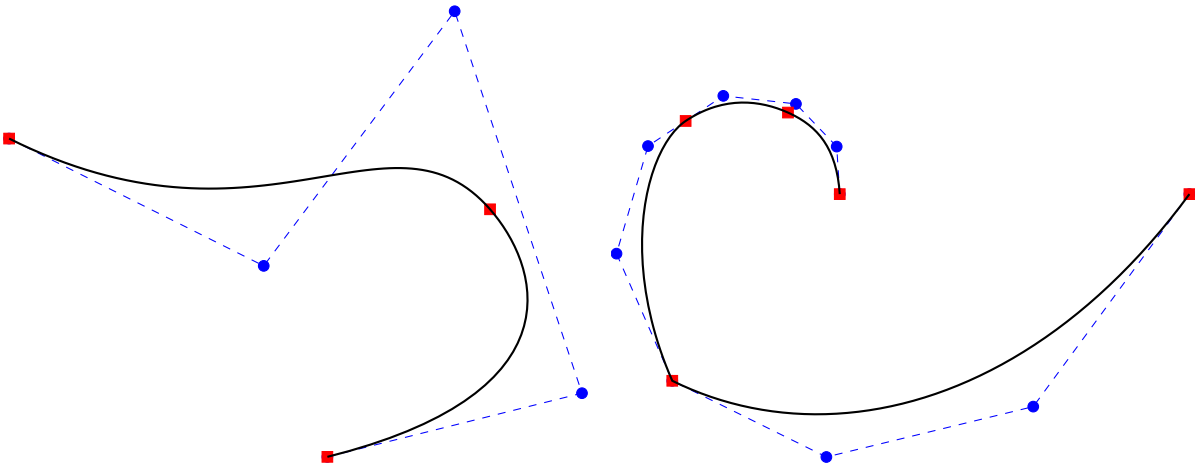
An open knot vector such as $\Xi = \{-1, \dots, -1, 1, \dots, 1\}$ (i.e. that has a single break span) generates a set of basis functions known as the Bernstein polynomials (see figure 4.1). Such a basis has the same length as the Legendre and Lagrange ones of the same degree, $\mathrm{J} = \mathrm{p} + 1$, and all three span the same function space: that of all polynomials of degree up to p. B-spline curves employing such a polynomial basis have historically been referred to as *Bézier curves*, i.e. B-splines with a single break point span.

It is possible to transform a general B-spline to/from piecewise-Bézier curves without changing its geometry via a process known as *knot insertion* [97, §5.2]. In [31, 32], for example, an isogeometric DG method is proposed exclusively for Bézier elements, taking advantage of the fact that such a conversion is available.

6.2.3. Classical FEA vs. IGA

In continuous Galerkin methods, it is typical to use Lagrange basis functions within each (continuously coupled) element, leading to C^0 smoothness for the approximate solution overall. B-spline and Lagrange bases are identical for the linear case[3] , yet the former is capable of maintaining up to $C^{\mathrm{p}-1}$ smoothness for any p while the latter is limited to piece-wise continuity only. A comparison between FEA and IGA bases of varying smoothness is shown in figure 6.5. For $\mathrm{p} > 1$, the CG and IGA bases are no longer identical, even in the C^0 case (6.5b); nevertheless, they still span the exact same function space. Consequently, keeping §6.2.2 in mind, we realize that B-spline basis functions can reproduce

[3] Provided that a Gauss-Lobatto distribution of nodes is employed in the latter—which is most convenient due to the continuity requirement of CG at element interfaces. Note, however, that this is not the nodal distribution used for DG in this work.



(a) $\Xi = \{-1,-1,-1,-1,\frac{1}{3},1,1,1,1\}$ **(b)** $\Xi = \{-1,-1,-1,-1,-\frac{1}{2},0,0,\frac{1}{2},\frac{1}{2},\frac{1}{2},1,1,1,1\}$

Figure 6.3: Examples of cubic B-spline curves (solid, black), with their control polygons (dashed, blue), control points (circles, blue) and breakpoints (squares, red).

both CG and DG methods, under particular combinations of knot vectors and number of patches.

A consequence of the increased smoothness of B-spline bases is a reduction in overall number of degrees of freedom, in the same way that CG experiences with respect to DG. Assume a DG discretization of a function space $S^h(\Omega)$ into K elements, each using degree p. Each element has its own set of $\mathrm{p}+1$ (polynomial) basis functions. This is reproduced in IGA with an open knot vector of K breakpoint spans and no continuity constraints whatsoever, which requires multiplicity $\mathrm{p}+1$ for every knot, i.e.:

$$\Xi = \{\underbrace{\xi_1,\dots,\xi_1}_{\mathrm{p}+1},\underbrace{\xi_2,\dots,\xi_2}_{\mathrm{p}+1},\dots,\underbrace{\xi_{\mathrm{K}+1},\dots,\xi_{\mathrm{K}+1}}_{\mathrm{p}+1}\}\ . \tag{6.6}$$

Both these two discretizations imply $\dim(S^h) = \mathrm{K}(\mathrm{p}+1)$. If we now increase continuity to C^0, the number of dimensions (i.e. the number of degrees of freedom) is reduced to that of CG—known to require fewer degrees of freedom than DG for a given order of accuracy. The smoother the basis functions of degree p are, the fewer distinct ones there can be in a given mesh (compare subfigures 6.5a and 6.5d); in general [97, §2.4]:

$$\dim\left(S^h\right) = \mathrm{K}(\mathrm{p}+1) - \sum_{k=1}^{\mathrm{K}+1}(\varkappa_k+1)\ , \tag{6.7}$$

where $-1 \leq \varkappa_k < \mathrm{p}$ is the number of successive derivatives of every basis function that are continuous at the ξ_k breakpoint—typically, $\varkappa_1 = \varkappa_{\mathrm{K}+1} = -1$.

6.3. Discontinous Galerkin isogeometric analysis (DGIGA)

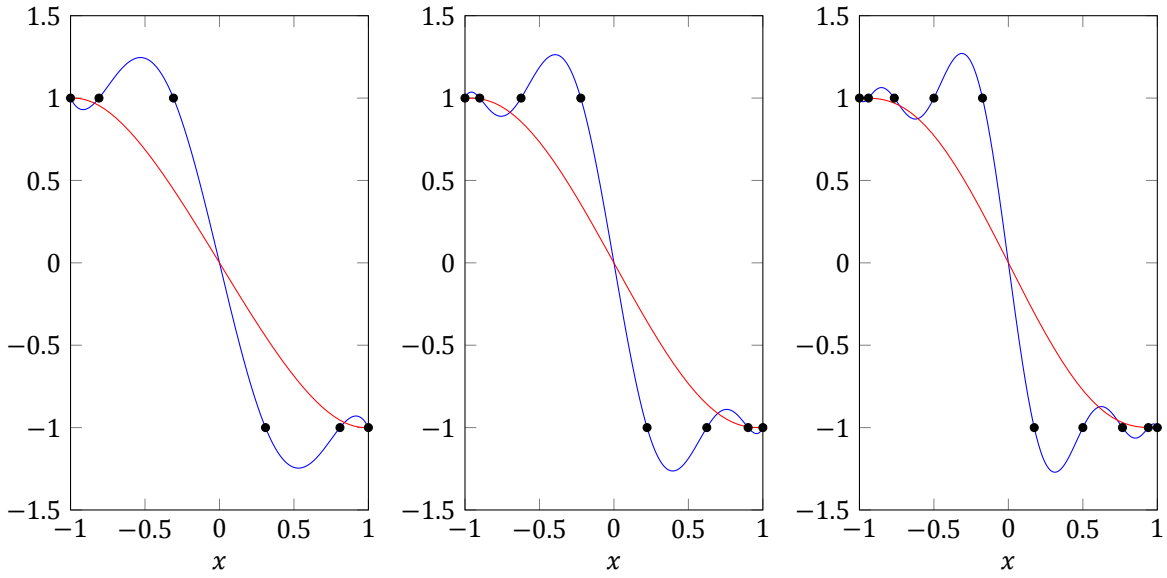
Once more, only 1D domains are under consideration in what follows. In a discontinuous Galerkin discretization, a patch is best assimilated to an element[4] (as defined in §3.1). Assume that $\mathcal{T}^h = \{\Omega_k\}_{k=1}^{\mathrm{K}}$. Let the test and basis function spaces on Ω_k be equal and defined as:

$$V_k^h \equiv S_k^h \coloneqq \mathrm{span}\{N_j^{\mathrm{p}}(\xi)\}_{j=1}^{\mathrm{J}}\ , \tag{6.8}$$

i.e. the space spanned by some set of B-spline functions, generated by any arbitrary open knot vector of $\mathrm{J}+\mathrm{p}+1$ elements. To maintain consistency with the general formulation in §3, let us assume $\xi \in \widetilde{\Omega}$, i.e.:

$$\Xi_k = \{\underbrace{-1,\dots,-1}_{\mathrm{p}+1},\xi_{\mathrm{p}+2},\dots,\xi_{\mathrm{J}},\underbrace{1,\dots,1}_{\mathrm{p}+1}\}\ , \tag{6.9}$$

[4] Some authors [28] prefer to consider the knot span as the element analogue; this is convenient when attempting a *continuous* Galerkin discretization based on B-spline basis functions (see §6.2.3).



(a) p = 5 **(b)** p = 7 **(c)** p = 9

Figure 6.4: B-spline (red) and Lagrange (blue) "interpolants" of degree p over a discontinuous set of non-equidistant points (Chebyshev nodes). The former are variation diminishing yet not actually interpolatory, while the latter experience spurious oscillations.

for $k = 1, 2, \ldots, \mathrm{K}$ (each DG element—i.e. IGA patch—may have its own values of p, J and smoothness distribution). Hence, the mappings to and from reference patch are still given by (3.4), and its B-spline basis functions $\boldsymbol{N} : \mathbb{R} \to \mathbb{R}^{\mathrm{J}}$ (sub and superscripts omitted in favor of conciseness) are defined in the same reference coordinate system as every other basis function in this report. Additionally, let us restrict interior knot multiplicity to $\mathrm{m} \leq \mathrm{p}$ to avoid the possibility of discontinuous basis functions within a patch. We may now carry on as in the general discontinuous high-order case (§3.3.4).

Each component of the approximate solution over Ω_k is a linear combination of the basis functions of S^h, i.e. :

$$\tilde{\boldsymbol{q}}_k^h(t, \xi) = \hat{\boldsymbol{Q}}_k(t) \boldsymbol{N}(\xi) \,. \tag{6.10}$$

The expansion coefficients—or degrees of freedom—play the role of "scalar control points" or *control values*. We may regard each approximate solution components as a B-spline curve in 2D state-space if we assign arbitrary locations to each degree of freedom; in this report I simply employ a uniform distribution of control point abscissae from the element's left edge to its right one. In other words: the graph of the function $q_{i,k}^h(t, x)$, at a fixed time instant, is a B-spline curve in the $(x, q_{i,k}^h)$ plane. This facilitates the analogy between control values in DGIGA and nodal values in DGSEM and FR/CPR (illustrated e.g. in figure 6.4).

6.3.1. Flux expansion coefficients

Once more, an approximation of the flux components into the same space as the solution's (3.14) is assumed for convenience:

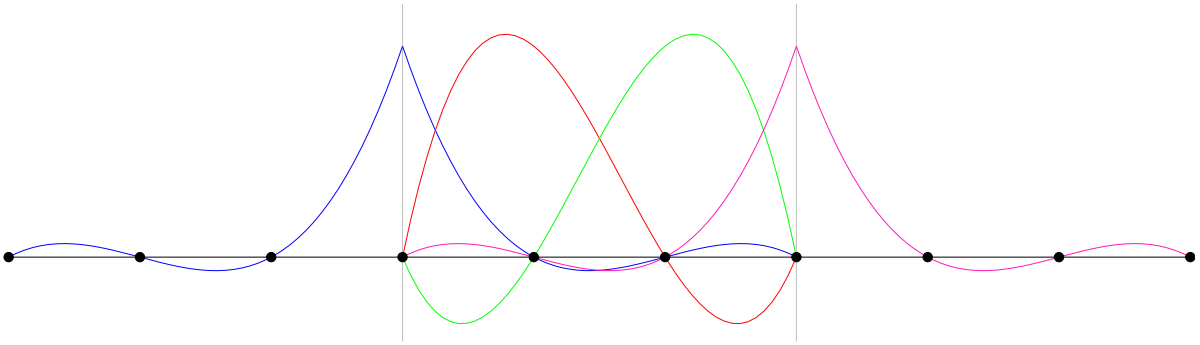
$$\tilde{\boldsymbol{f}}_k^h(t, \xi) = \hat{\boldsymbol{F}}_k(t) \boldsymbol{N}(\xi) \,, \tag{6.11}$$

at the cost of an additional error being introduced into the discretization in nonlinear cases. The alternative would be to apply higher-order quadrature to the nonlinear flux function, evaluated on the approximate solution [31, 32] (see §4.2.1).

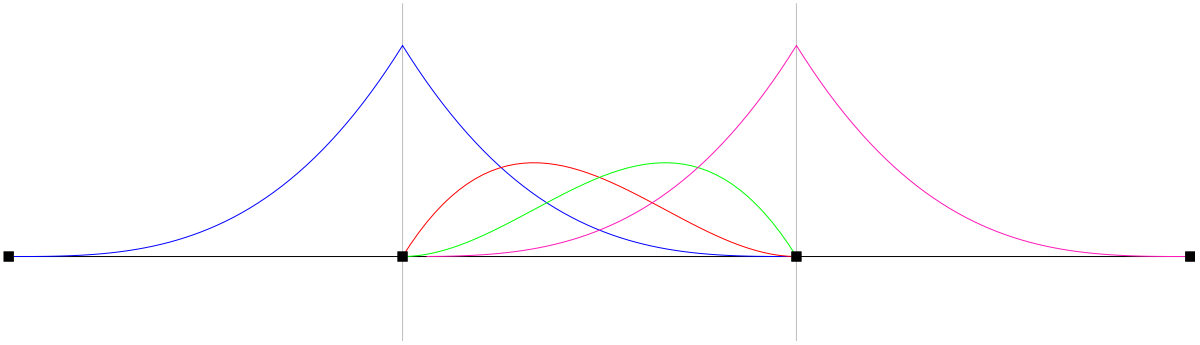
Flux expansion coefficients in (6.11) are the result of applying the flux function to the approximate solution sampled at each control location, i.e. :

$$\tilde{\boldsymbol{f}}_k^h(t, \xi_j) = \boldsymbol{f}\left(\tilde{\boldsymbol{q}}_k^h(t, \xi_j)\right) \,, \tag{6.12}$$

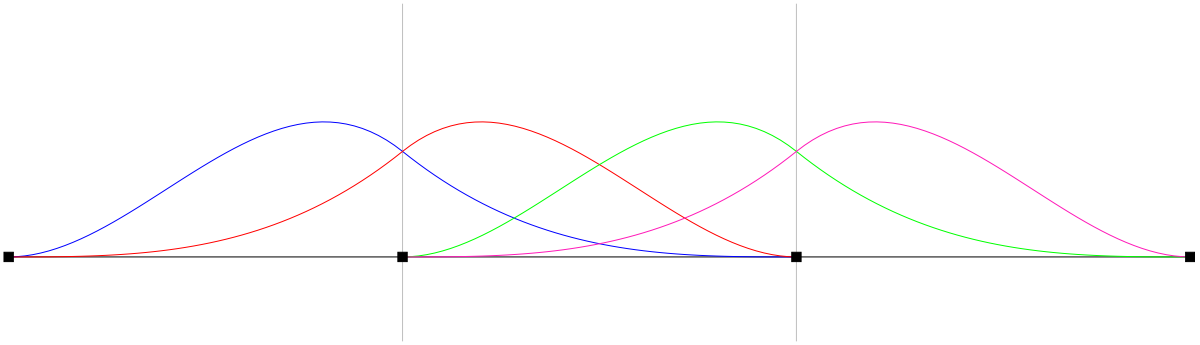
which is exact, having made the assumption beforehand that the flux can be accurately approximated as a B-spline curve in the same space as that of the solution's (as mentioned in the previous paragraph,



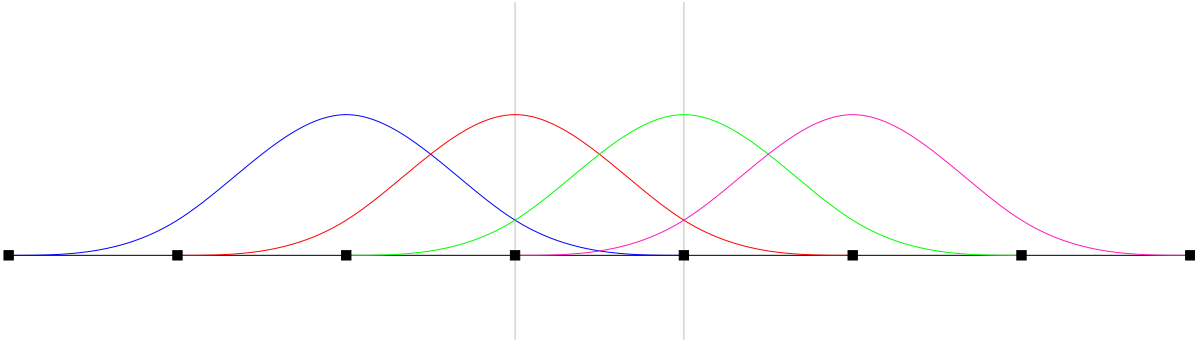
(a) C^0 piecewise Lagrange polynomials (classical FEA)



(b) C^0 B-splines



(c) C^1 B-splines



(d) C^2 B-splines (typical IGA)

Figure 6.5: Cubic basis functions with nonzero support on an arbitrary interior CG element (in 6.5a) or knot span, with uniformly distributed nodes (circles) or knots (squares). Notice that the portion of the patch over which an average B-spline basis function has nonzero support increases as they become smoother.

this is generally not the case). Defining, analogously to (4.19), the following *patch Vandermonde matrix*,

$$\boldsymbol{N}(\boldsymbol{\xi}^{\mathsf{T}}) \coloneqq \begin{bmatrix} N_1(\xi_1) & N_1(\xi_2) & \cdots & N_1(\xi_{\mathrm{J}}) \\ N_2(\xi_1) & N_2(\xi_2) & \cdots & N_2(\xi_{\mathrm{J}}) \\ \vdots & \vdots & & \vdots \\ N_{\mathrm{J}}(\xi_1) & N_{\mathrm{J}}(\xi_2) & \cdots & N_{\mathrm{J}}(\xi_{\mathrm{J}}) \end{bmatrix}, \tag{6.13}$$

the flux expansion coefficients in (6.11) satisfying (6.12) can be obtained as:

$$\hat{\boldsymbol{F}}_k = \left(\boldsymbol{f}\left(\hat{\boldsymbol{Q}}_k \boldsymbol{N}(\boldsymbol{\xi}^{\mathsf{T}})\right)\right)\left(\boldsymbol{N}(\boldsymbol{\xi}^{\mathsf{T}})\right)^{-1}, \tag{6.14}$$

where $\boldsymbol{f}$ is applied column-by-column.

An even simpler flux evaluation is proposed in [57, 90]:

$$\hat{\boldsymbol{f}}_{jk}(t) \approx \boldsymbol{f}\left(\hat{\boldsymbol{q}}_{jk}(t)\right), \tag{6.15}$$

which would be equivalent to (6.14) if the basis were nodal[5]. While this is clearly not the case for a modal DG basis in general, B-splines have "quasi-nodal" behavior in the sense of property 6.12; this prevents the approximate solution from outright diverging, but could reduce its accuracy even further.

Direct comparison between these two approaches in terms of numerical errors (figures 6.7 and 6.9) suggests that the solution obtained with (6.15) experiences additional numerical diffusion in relation to that computed using (6.14), and that this is the case (for the particular example considered) both with and without AFC-based limiting. Nevertheless, the role of (6.13) and its inverse in (6.14) warrants the question of how well-conditioned said matrix is. It turns out that, for the assumed uniform distribution of control points, (6.13) can easily become quasi-singular as the number of breakpoint spans grows, for degrees $\mathrm{p} > 2$. Moreover, non-uniform distributions of control points seem to *worsen* this problem even further (figure 6.11).

6.3.2. Semi-discrete DGIGA operators

The general semi-discrete conservation law (3.23) for DGIGA may be written as:

$$\frac{\mathrm{d}\hat{\boldsymbol{Q}}_k}{\mathrm{d}t}\int_{\Omega_k} \boldsymbol{N}\boldsymbol{N}^{\mathsf{T}}\,\mathrm{d}x = \hat{\boldsymbol{F}}_k \int_{\Omega_k} \boldsymbol{N}\frac{\mathrm{d}\boldsymbol{N}^{\mathsf{T}}}{\mathrm{d}x}\,\mathrm{d}x - \left[\check{\boldsymbol{f}}\boldsymbol{N}^{\mathsf{T}}\right]_{\partial\Omega_k}, \tag{6.16}$$

which corresponds to:

$$\boldsymbol{\mathcal{M}}_k = \frac{\Delta x_k}{2}\int_{-1}^{1} \boldsymbol{N}\boldsymbol{N}^{\mathsf{T}}\,\mathrm{d}\xi, \qquad \boldsymbol{\mathcal{C}}_k = \int_{-1}^{1} \boldsymbol{N}\left(\boldsymbol{N}'\right)^{\mathsf{T}}\,\mathrm{d}\xi. \tag{6.17}$$

One last remaining issue is the evaluation of the integral terms in the mass and discrete gradient matrices. For DGIGA, I propose a "brute force" approach based on Gauss quadrature at the knot span level, carried out as as follows. Let an additional affine mapping $\psi_l : \Sigma_l \to [-1, 1]$, analogous to (3.4), be defined as:

$$\psi_l(\sigma) \coloneqq \overline{\xi}_l + \frac{\Delta\xi_l}{2}\sigma, \tag{6.18}$$

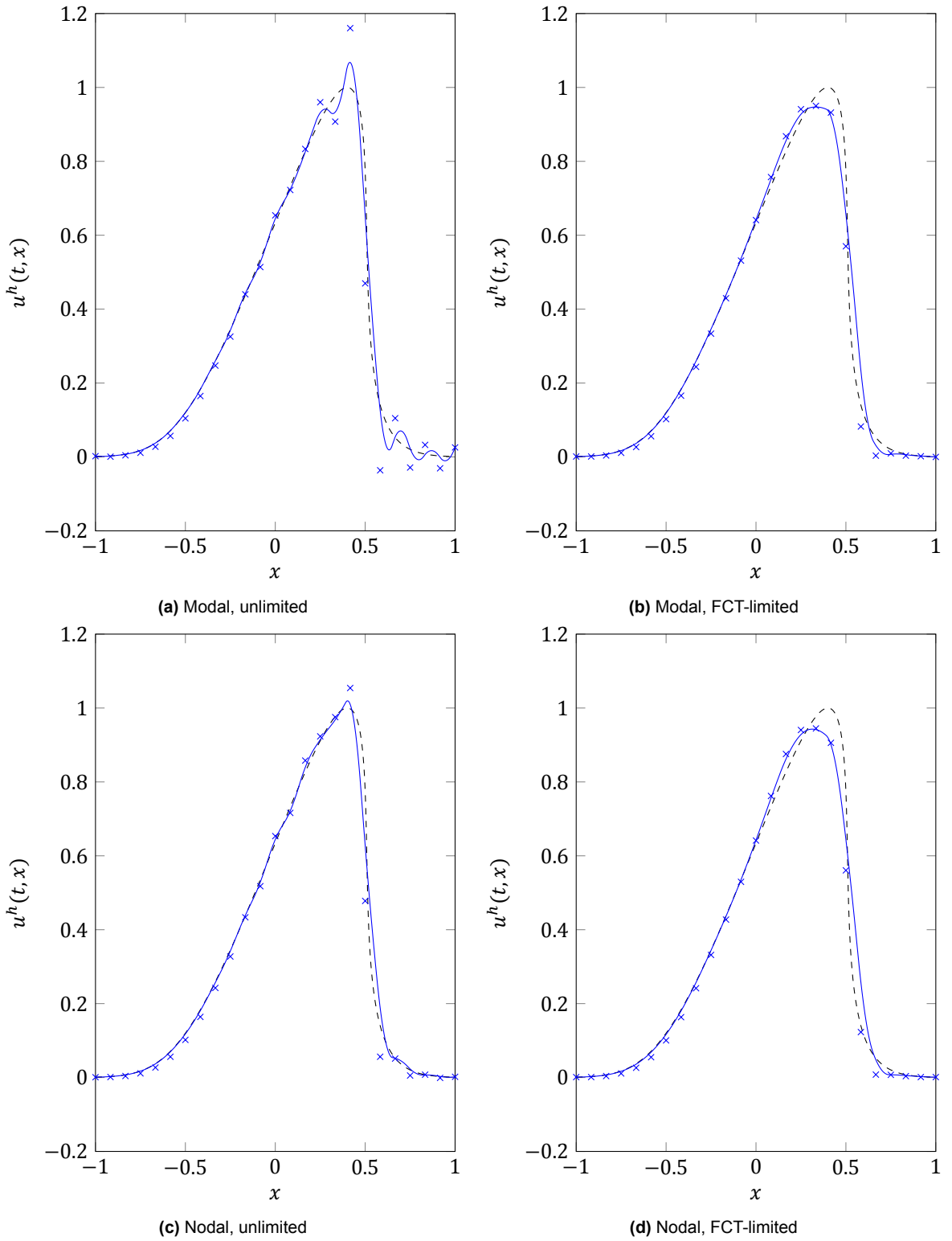
for every $\xi_l \in \Xi_k \;:\; l = 1, 2, \ldots, \mathrm{J} + \mathrm{p}$, i.e. each knot span in Ω_k. Operator entries are first evaluated in reference patch coordinates:

$$\int_{\Omega_k} N_j N_r\,\mathrm{d}x = \mathcal{X}'_k \int_{-1}^{1} N_j N_r\,\mathrm{d}\xi, \qquad \int_{\Omega_k} N_j \frac{\mathrm{d}N_r}{\mathrm{d}x}\,\mathrm{d}x = \int_{-1}^{1} N_j\left(N_r\right)'\,\mathrm{d}\xi; \tag{6.19}$$

then, each integral is split knot span-wise:

$$\int_{-1}^{1} N_j N_r\,\mathrm{d}\xi = \sum_{l=1}^{\mathrm{J}+\mathrm{p}} \int_{\xi_l}^{\xi_{l+1}} N_j N_r\,\mathrm{d}\xi, \qquad \int_{-1}^{1} N_j\left(N_r\right)'\,\mathrm{d}\xi = \sum_{l=1}^{\mathrm{J}+\mathrm{p}} \int_{\xi_l}^{\xi_{l+1}} N_j\left(N_r\right)'\,\mathrm{d}\xi; \tag{6.20}$$

[5] In addition, nodal and modal treatments are identical for the linear flux case (e.g. advection equation); see §A.3.3.



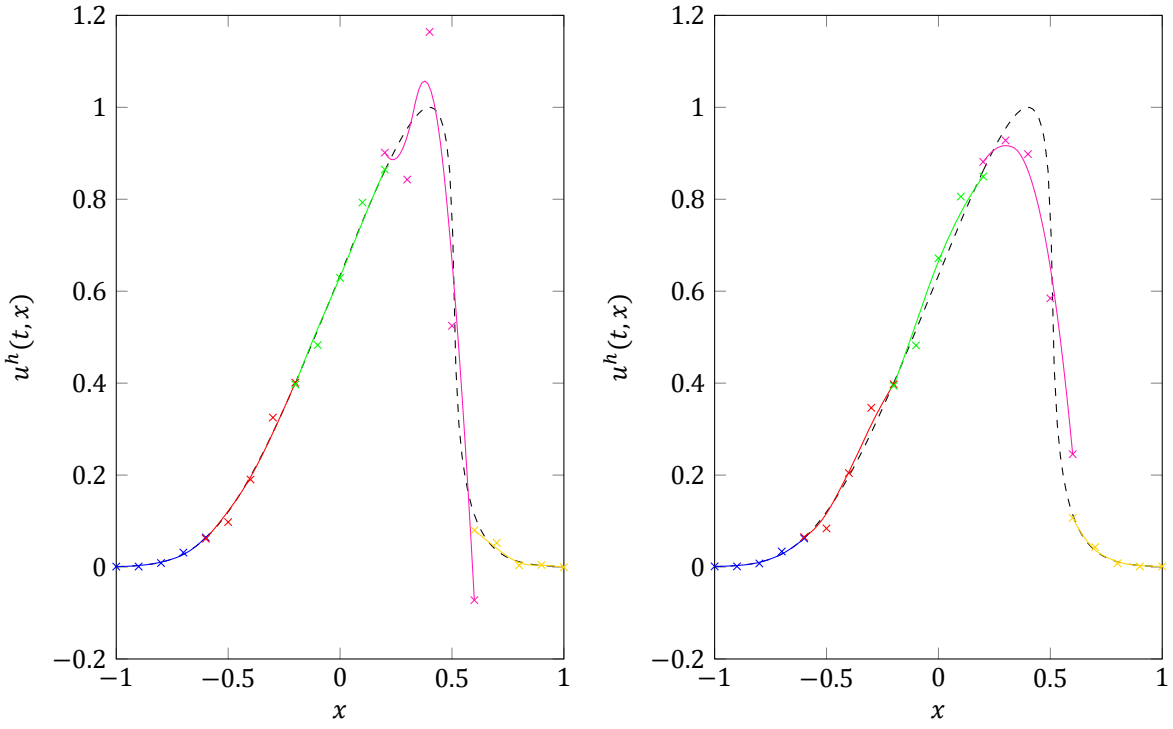
(a) Modal, unlimited

(b) Modal, FCT-limited

(c) Nodal, unlimited

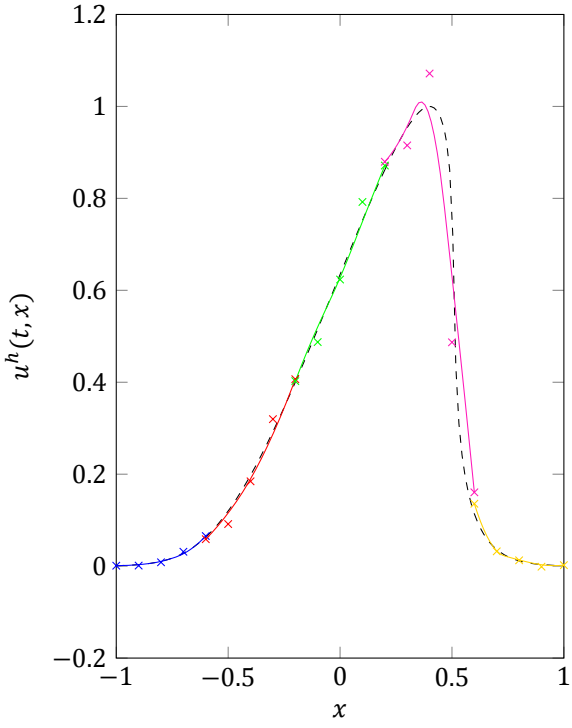
(d) Nodal, FCT-limited

Figure 6.6: Approximate solution to Burgers equation at $t = 0.4$ (periodic boundaries, Gaussian hump initial condition at $t = 0$), with (right) and without (left) AFC-based FCT limiting, for the modal (6.14) and quasi-nodal (6.15) treatment of flux coefficients in IGA (i.e. single patch). Control points are shown as cross markers and the exact solution as a black dashed curve. All cases approximate the solution as a quadratic C^1 B-spline curve, made up of 23 polynomial segments (breakpoint spans), resulting in a total degree of freedom count of 25. Time scheme is SSP-RK3, with $\Delta t = 4 \times 10^{-4}$.

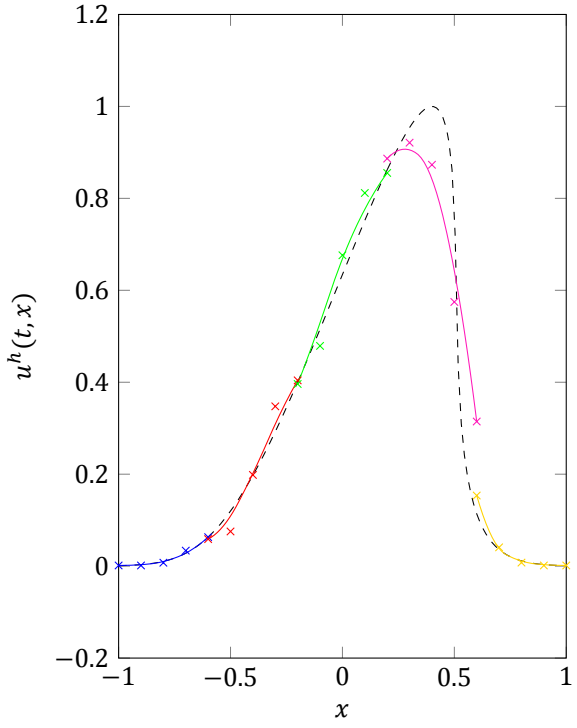


(a) Modal, unlimited

(b) Modal, FCT-limited

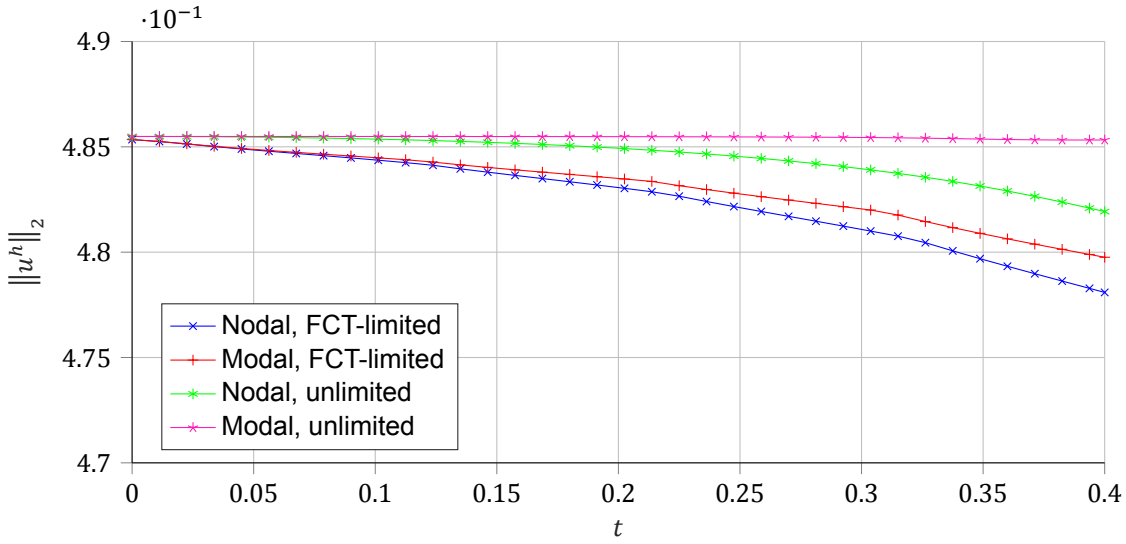


(c) Nodal, unlimited

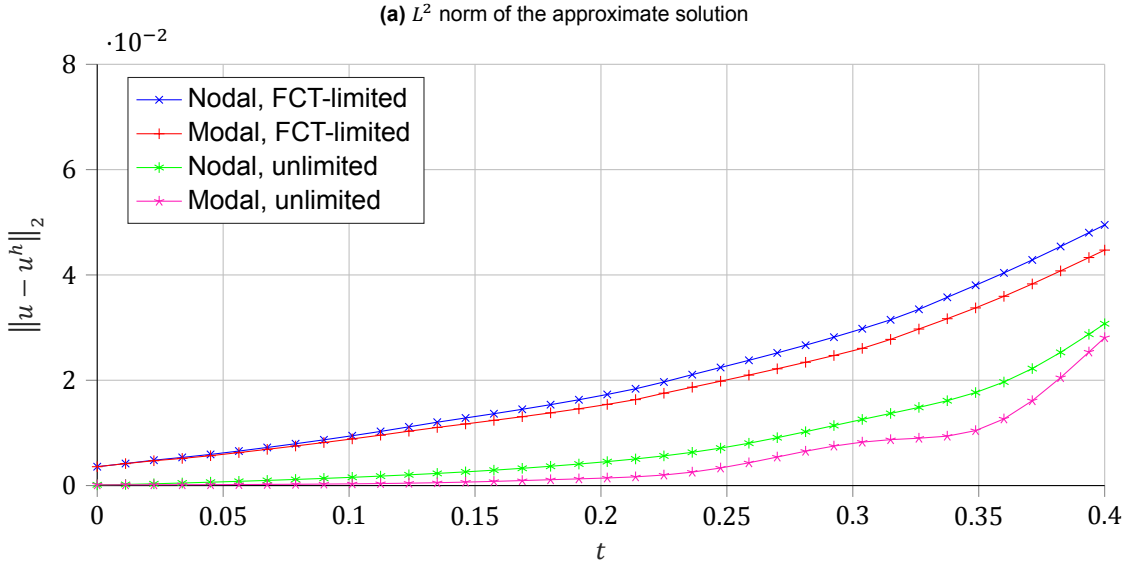


(d) Nodal, FCT-limited

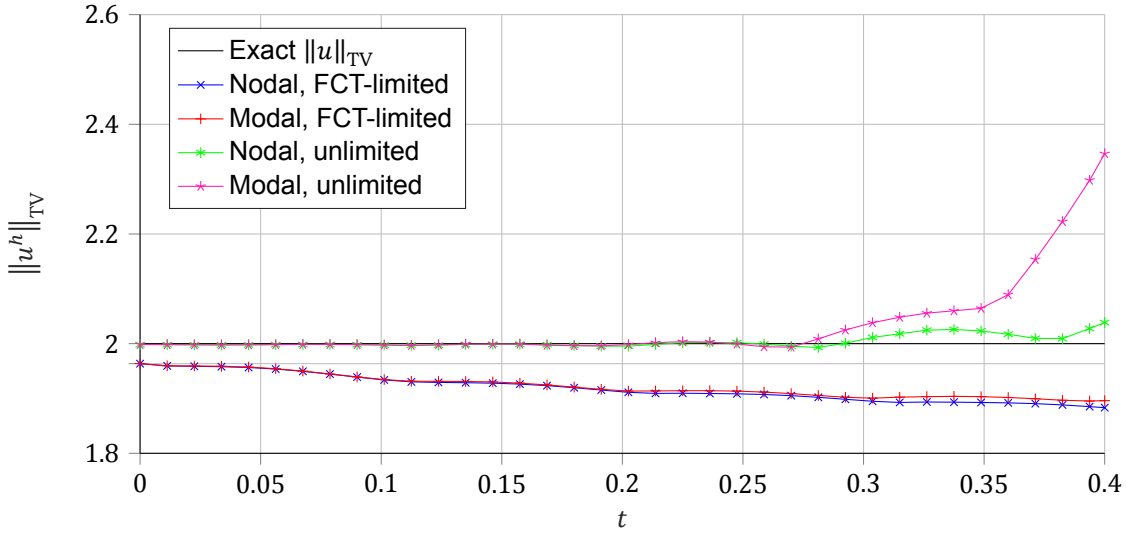
Figure 6.7: DGIGA version of the results in figure 6.6. The approximate solution is now globally C^{-1}, divided into 5 C^1 B-spline patches, each in turn made up of 3 quadratic C^∞ polynomials; the overall number of degrees of freedom remains unchanged (25). All results still employ SSP-RK3, but this time with $\varsigma = 10^{-3}$ and 5 elements (depicted in distinct colors), each with $\Xi_k = \{-1, -1, -1, -\frac{1}{3}, \frac{1}{3}, 1, 1, 1\}$.



(a) L^2 norm of the approximate solution

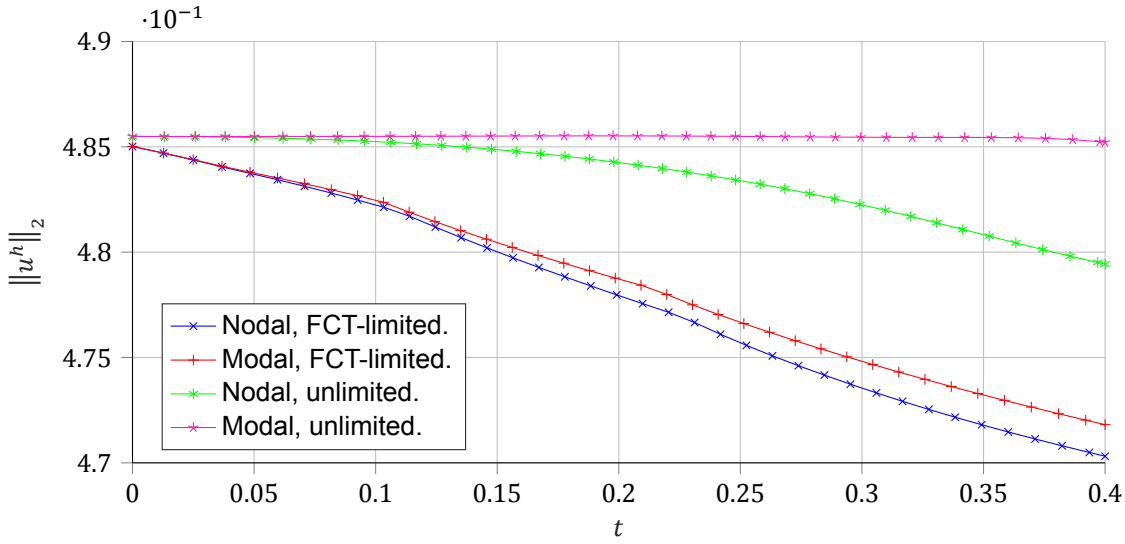


(b) L^2 norm of the approximate solution's error

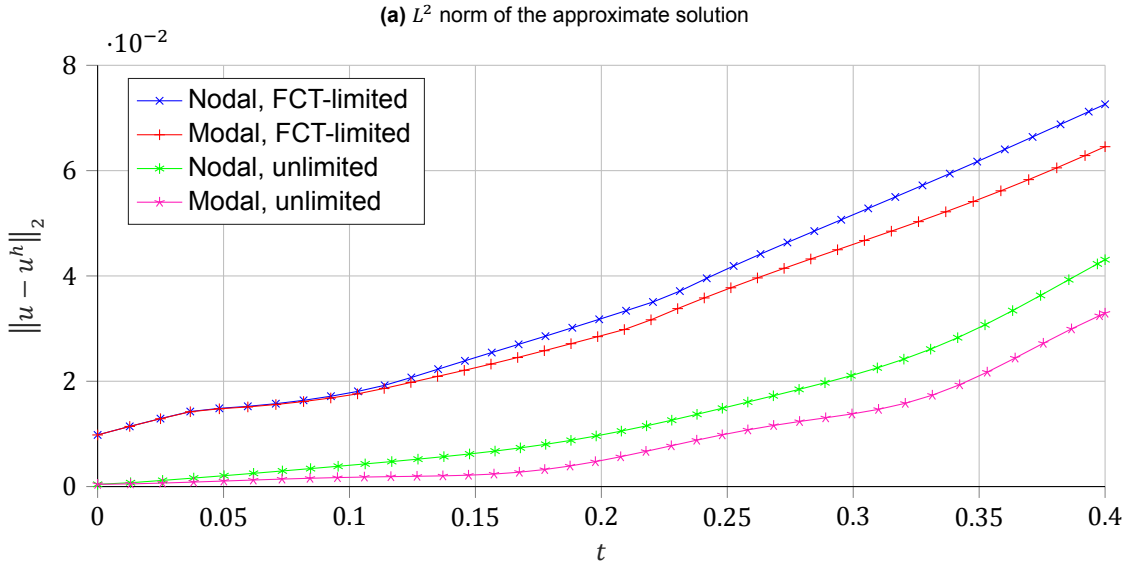


(c) Estimated total variation of the approximate solution; see §8.1

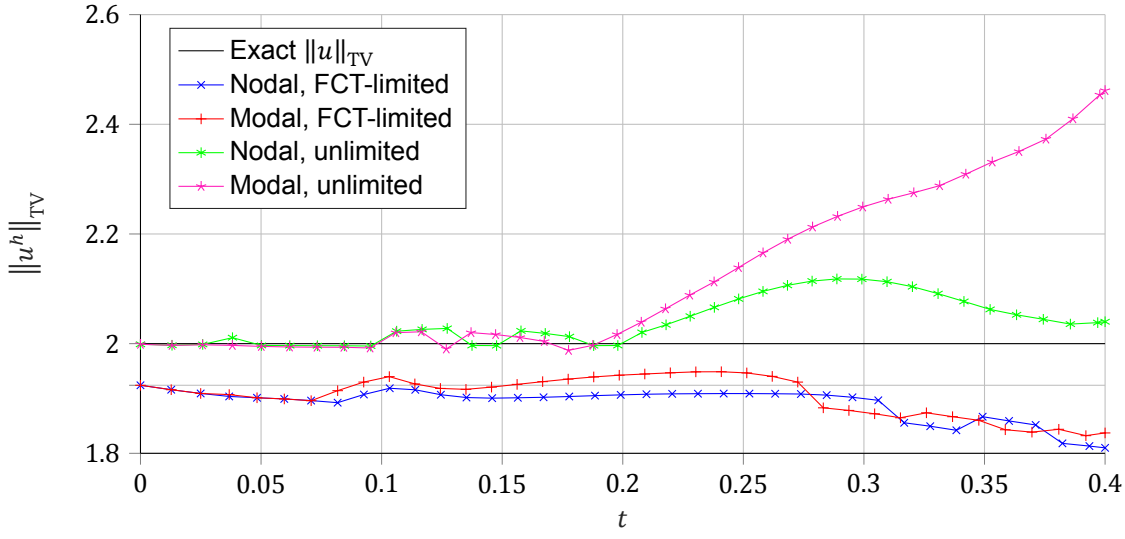
Figure 6.8: Behavior of the approximate solution in time (sampled every 25 time-steps), for all cases in figure 6.6.



(a) L^2 norm of the approximate solution



(b) L^2 norm of the approximate solution's error



(c) Estimated total variation of the approximate solution; see §8.1

Figure 6.9: Behavior of the approximate solution in time (sampled every 25 time-steps), for all cases in figure 6.7.

followed by each span being brought to reference span coordinates:

$$\int_{\xi_l}^{\xi_{l+1}} N_j N_r \,\mathrm{d}\xi = \psi_l' \int_{-1}^{1} N_j N_r \,\mathrm{d}\sigma\,, \qquad \int_{\xi_l}^{\xi_{l+1}} N_j \left(N_r\right)' \mathrm{d}\xi = \psi_l' \int_{-1}^{1} N_j \left(N_r\right)' \mathrm{d}\sigma\,; \tag{6.21}$$

and, finally, every knot span-wise integral is evaluated exactly using a unique set of $\mathrm{p}+1$ Gauss-Legendre quadrature points (σ_n) and weights (w_n) on the interval $[-1,1]$:

$$\int_{-1}^{1} N_j N_r \,\mathrm{d}\sigma = \sum_{n=1}^{\mathrm{p}+1} N_j(\psi_l(\sigma_n))\; N_r(\psi_l(\sigma_n))\; w_n\,, \tag{6.22a}$$

$$\int_{-1}^{1} N_j \left(N_r\right)' \mathrm{d}\sigma = \sum_{n=1}^{\mathrm{p}+1} N_j(\psi_l(\sigma_n))\; \left(N_r\right)'(\psi_l(\sigma_n))\; w_n\,. \tag{6.22b}$$

Note that for the single breakpoint span case, the mapping (6.18) reduces to the identity and quadrature is performed element-wise.

The mass matrix in DGIGA is known to be very badly conditioned (figure 6.10), and it is not diagonal (unlike in the previous two methods); it instead has a bandwidth of $2\mathrm{p}+1$ (property 6.1). Nevertheless, it is *symmetric positive definite*.

Proof. $\boldsymbol{\mathcal{M}}_k$ is symmetric because the product of B-spline basis functions is commutative: $N_j N_r \equiv N_r N_j$. For an arbitrary vector $\hat{\boldsymbol{q}} \in \mathbb{R}^{\mathrm{J}}$, it will be positive definite if $\hat{\boldsymbol{q}}^{\mathrm{T}} \boldsymbol{\mathcal{M}}_k \hat{\boldsymbol{q}} \geq 0$ (with the equality implying the trivial case) [16, §0.2]. This may be written as:

$$\sum_{j=1}^{\mathrm{J}} \sum_{r=1}^{\mathrm{J}} \hat{q}_j \left(\int_{\Omega_k} N_j N_r \,\mathrm{d}x \right) \hat{q}_r = \int_{\Omega_k} \left(\sum_{j=1}^{\mathrm{J}} \hat{q}_j N_j \right) \left(\sum_{r=1}^{\mathrm{J}} N_r \hat{q}_r \right) \mathrm{d}x \equiv \left\| q_k^h \right\|_{L^2}^2\,, \tag{6.23}$$

i.e. the square of the L^2 norm of some arbitrary function in S^h, which is non-negative and zero only if $\hat{\boldsymbol{q}} = 0$, as required. □

The discrete gradient operator matrix, provided that $\mathrm{p} > 0$ (otherwise the gradient operator is a scalar and trivially zero), is *quasi-skew-symmetric* in the following sense:

$$c_{jr} = \begin{cases} -\frac{1}{2} & \text{if } j = r = 1 \\ +\frac{1}{2} & \text{if } j = r = \mathrm{J} \\ -c_{rj} & \text{otherwise} \end{cases}\,. \tag{6.24}$$

Proof. It follows from the chain rule: $\int_{-1}^{1} N_j \left(N_r\right)' \mathrm{d}\xi = \left[N_j N_r\right]_{-1}^{1} - \int_{-1}^{1} N_r \left(N_j\right)' \mathrm{d}\xi$. If $j = r$, since the first and last B-splines in a patch are interpolatory at its respective edges:

$$\left[N_j N_j\right]_{-1}^{1} = \begin{cases} -1 & \text{if } j = 1 \\ +1 & \text{if } j = \mathrm{J} \\ 0 & \text{otherwise} \end{cases}\,. \tag{6.25}$$

Then:

- if $j = 1$:

$$\int_{-1}^{1} N_1 \left(N_1\right)' \mathrm{d}\xi = -1 - \int_{-1}^{1} N_1 \left(N_1\right)' \mathrm{d}\xi \implies \int_{-1}^{1} N_1 \left(N_1\right)' \mathrm{d}\xi = -\frac{1}{2} \tag{6.26}$$

- else, if $j = \mathrm{J}$:

$$\int_{-1}^{1} N_{\mathrm{J}} \left(N_{\mathrm{J}}\right)' \mathrm{d}\xi = 1 - \int_{-1}^{1} N_{\mathrm{J}} \left(N_{\mathrm{J}}\right)' \mathrm{d}\xi \implies \int_{-1}^{1} N_{\mathrm{J}} \left(N_{\mathrm{J}}\right)' \mathrm{d}\xi = \frac{1}{2} \tag{6.27}$$

- else:

$$\int_{-1}^{1} N_j \left(N_j\right)' \,\mathrm{d}\xi = -\int_{-1}^{1} N_j \left(N_j\right)' \,\mathrm{d}\xi \implies \int_{-1}^{1} N_j \left(N_j\right)' \,\mathrm{d}\xi = 0 \tag{6.28}$$

Otherwise (if $j \neq r$), at least one of the two functions is zero at any given edge; therefore: $\left[N_j N_r\right]_{-1}^{1} = 0$. This implies skew-symmetry in all non-diagonal entries: $\int_{-1}^{1} N_j \left(N_r\right)' \,\mathrm{d}\xi = -\int_{-1}^{1} N_r \left(N_j\right)' \,\mathrm{d}\xi$. □

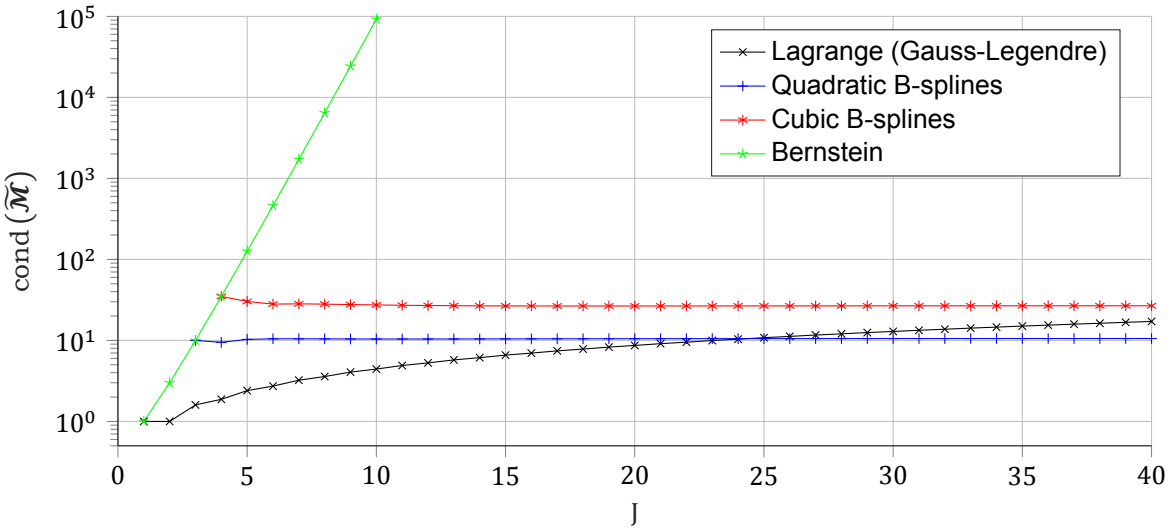


Figure 6.10: Condition number of the reference element mass matrix for various B-spline bases (also Lagrange, see 4.2), as a function of the number of degrees of freedom per patch. This gives a rough idea of relative computational cost vs. memory requirements between bases, regardless of accuracy. The condition number of the Bernstein mass matrix keeps growing exponentially until assembly breaks down (reaching $\approx 7 \times 10^{16}$ at $\mathrm{J} = 30$).

6.4. Algebraic flux correction (AFC)

In [90], a B-spline based continuous Galerkin discretization is turned into a high-resolution scheme by combining it with a limiting strategy in the context of *algebraic flux correction* [78]. This pairing is quite synergistic, since it exploits the nonnegativity property of IGA bases to guarantee that the approximate solution is *local extremum diminishing*[6] (LED) in the presence of discontinuous features, regardless of the degree of the basis. This is in contrast to Lagrange-based FEM, for which only the $\mathrm{p} = 1$ case is amenable to this methodology. Unlike typical shock capturing schemes for DG (see §8), such an approach achieves *sub-element resolution*, i.e. the discontinuity is modeled across knot spans rather than patches (see figure 6.12).

6.4.1. A predictor-corrector approach to high-resolution

Algebraic flux correction is based on a predictor-corrector strategy. Equation (6.16) defines a matrix of *uncorrected* high-order residuals, namely:

$$\hat{\boldsymbol{\mathcal{R}}}_k = \left(\hat{\boldsymbol{F}}_k \boldsymbol{\mathcal{C}}_k - \left[\check{\boldsymbol{f}} \boldsymbol{N}^{\intercal}\right]_{\partial\Omega_k}\right) \boldsymbol{\mathcal{M}}_k^{-1} . \tag{6.29}$$

In combination with a B-spline basis, AFC is able to (algebraically) construct a low-order version of these residuals such that, when used in conjunction with a strong stability preserving time-integration scheme (see §7), no spurious oscillation are generated in the end-of-step solution [90]. These so-called *predictor residuals*, are defined as:

$$\hat{\boldsymbol{\mathcal{R}}}_k^L \coloneqq \left(\hat{\boldsymbol{F}}_k \boldsymbol{\mathcal{C}}_k - \left[\check{\boldsymbol{f}} \boldsymbol{N}^{\intercal}\right]_{\partial\Omega_k} + \boldsymbol{F}_k^D\right) \left(\boldsymbol{\mathcal{M}}_k^L\right)^{-1} , \tag{6.30}$$

[6] Local extremum diminishing (LED) and total variation diminishing (TVD) criteria are equivalent in one dimension [59]

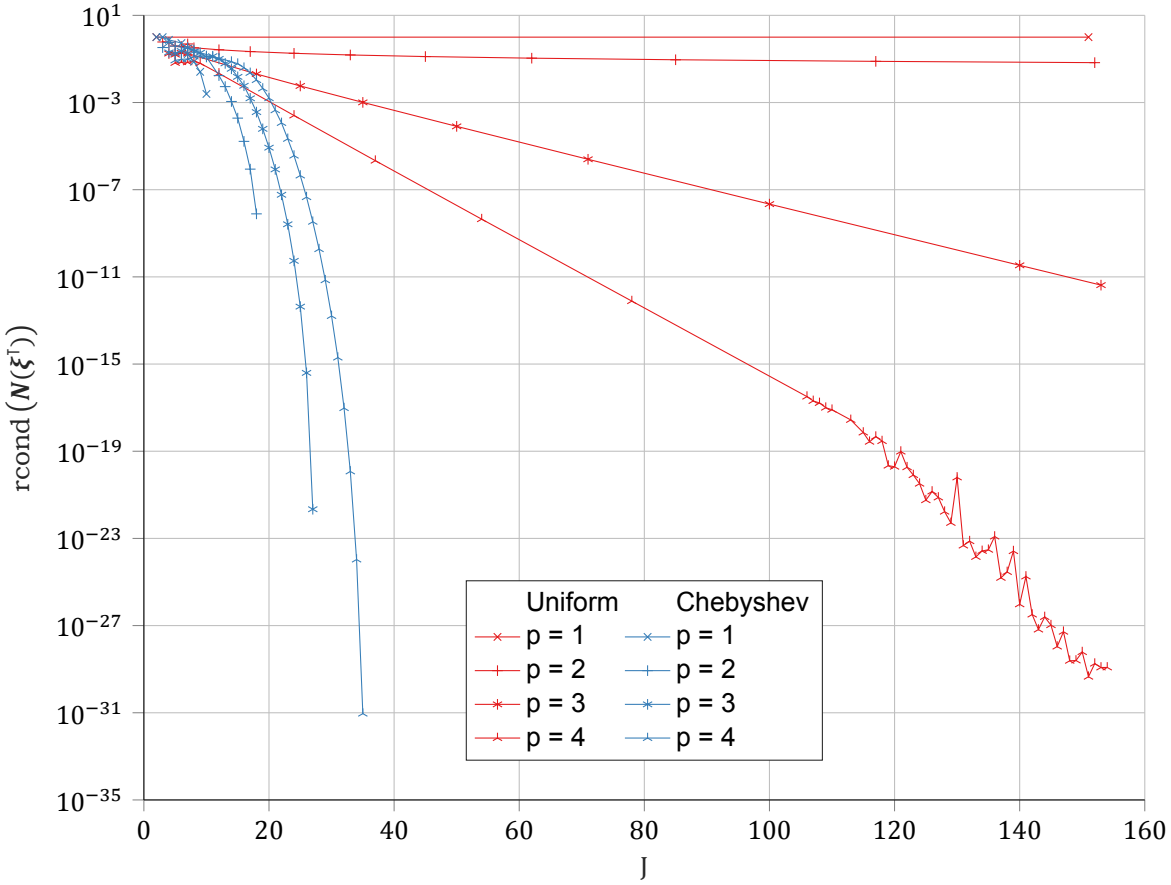


Figure 6.11: Reciprocal condition number of (6.13), for various degrees and control point distributions, as a function of the number of basis functions per patch. Smoothness class is $C^{\mathrm{p}-1}$ in all cases. Higher is better.

which differs from (6.29) in two aspects: the mass matrix is replaced by its *lumped* counterpart, and an explicit *numerical diffusion* term is added.

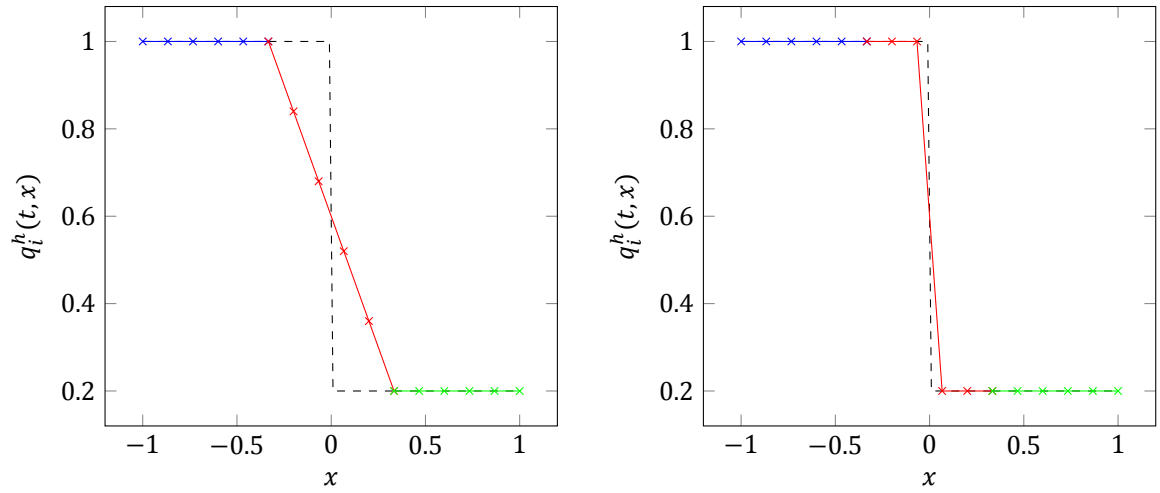
From an implementation perspective, this can be seen as a replacement of the DGIGA operator with a low-order counterpart—say, a DGIGA-AFC operator—which ensures *a priori* that no spurious oscillations will appear in the solution as it is advanced a number of time-stages, until the next time-step. All other routines remain unchanged (e.g. evaluation of internal and numerical fluxes, time-integration; all the terms of (6.30) are re-evaluated using latest available predictor control variables at every intermediate stage). Once all "transported and diffused"—advanced from t to $t + \Delta t$ using (6.30) instead of (6.29)—sets of control values $\hat{\boldsymbol{Q}}_k^L(t + \Delta t)$ in $\mathcal{T}^h$ are available, a correction procedure takes care of recovering high-order accuracy. This second step can be treated as an "a posteriori" limiter (or, more precisely, an "anti-limiter"), and is described as such in §8.7. The process can then start again, and so on.

6.4.2. Mass matrix lumping

The lumped mass matrix for DGIGA is:

$$\boldsymbol{\mathcal{M}}_k^L \coloneqq \begin{bmatrix} m_{1k}^L & & 0 \\ & \ddots & \\ 0 & & m_{\mathrm{J}k}^L \end{bmatrix}, \qquad m_{jk}^L \coloneqq \sum_{r=1}^{\mathrm{J}} m_{jrk} \equiv \frac{\Delta x_k}{2} \int_{-1}^{1} N_j \, \mathrm{d}\xi \,. \tag{6.31}$$

Unlike high-order Lagrange polynomials in general, B-spline basis functions guarantee that all off-diagonal entries of $\boldsymbol{\mathcal{M}}^L$ are strictly positive [90]. Note that an added benefit of employing the lumped mass matrix is that we recover a decoupled left-hand-side in (3.23), circumventing one of the disadvantages of the B-spline basis not being orthogonal.



(a) DGIGA + TVB limiter (§8.4)

(b) DGIGA-AFC + FCT limiter (§8.7)

Figure 6.12: Inter-element vs. sub-element resolution of a shock (dashed line) using 2nd order DGIGA. Markers denote control points; the two meshes are identical (each color corresponds to a different patch). In both examples, the shock has been *projected* onto the mesh; both limiters perform optimally in this situation.

6.4.3. Artificial viscosities

The term $\boldsymbol{F}_k^D$ in (6.30) represents an array of net diffusive fluxes "going into" each given control point, i.e. :

$$\boldsymbol{F}_k^D \coloneqq \begin{bmatrix} \boldsymbol{f}_{1k}^D & \boldsymbol{f}_{2k}^D & \cdots & \boldsymbol{f}_{\mathrm{J}k}^D \end{bmatrix} . \tag{6.32}$$

In AFC, an *edge-based representation* is common; in this sense, a so-called edge is a pair of distinct control points with shared nonzero support. A simple general way to identify such edges is through the *sparsity graph* of the (consistent) mass matrix: any component $m_{jrk} \neq 0$ implies that there is at least one basis function that the pair of control points j and r falls within nonzero support of. Diffusive fluxes are defined such that:

$$\boldsymbol{f}_{rk}^D \coloneqq \sum_{j \neq r} \boldsymbol{D}_{rjk} \left(\hat{\boldsymbol{q}}_{jk} - \hat{\boldsymbol{q}}_{rk} \right) , \tag{6.33}$$

to ensure a suitable predictor [90]. The matrix $\boldsymbol{D}_{rjk} \in \mathbb{R}^{\mathrm{I}\times\mathrm{I}}$ determines the contribution due to control point j to the net diffusion added to control point r. Various definitions for it are given in [78, §5]. I propose the following as a compromise between robustness and accuracy.

Let $\boldsymbol{A}^r$, $\boldsymbol{A}^j$ and $\widetilde{\boldsymbol{A}}$ be the Jacobian matrices of the system of PDEs, evaluated at $\hat{\boldsymbol{q}}_{rk}$, $\hat{\boldsymbol{q}}_{jk}$ and the Roe-averaged state (2.45) between the previous two (arithmetic average in Burgers), respectively. As in [90]:

$$\boldsymbol{D}_{rjk} \coloneqq \left| e_{rjk} \right| \widetilde{\boldsymbol{R}} \left| \widetilde{\boldsymbol{\Lambda}} \right| \widetilde{\boldsymbol{L}} , \qquad e_{rjk} \coloneqq \frac{c_{rj} - c_{jr}}{2} , \tag{6.34}$$

but, in advance, apply an entropy fix[7] [64] to every entry of $\widetilde{\boldsymbol{\Lambda}}$:

$$\tilde{\lambda}_i \leftarrow \frac{\tilde{\lambda}_i^2 + \varepsilon^2}{2\varepsilon} \quad \text{if } \left| \tilde{\lambda}_i \right| < \varepsilon , \qquad \varepsilon \coloneqq \max\left\{ 0, \tilde{\lambda}_i - \lambda_i^r, \lambda_i^j - \tilde{\lambda}_i \right\} . \tag{6.35}$$

Should (6.34) fail to prevent oscillations in the predicted solution, the following may be used instead [78, equation 46]:

$$\boldsymbol{D}_{rjk} \coloneqq \left| e_{rjk} \right| d_{rjk} \boldsymbol{I} , \qquad d_{rjk} \coloneqq \max\left\{ \left|\lambda_1^r\right|, \left|\lambda_2^r\right|, \ldots, \left|\lambda_{\mathrm{I}}^r\right|, \left|\lambda_1^j\right|, \left|\lambda_2^j\right|, \ldots, \left|\lambda_{\mathrm{I}}^j\right| \right\} , \tag{6.36}$$

which, according to Kuzmin et al. [78], is the most robust (but also most diffusive) of the alternatives listed therein. Yet another option, more economical and comparable in accuracy to (6.34), is to employ an arithmetic average (instead of Roe's) [78, equation 44].

[7] Harten and Hyman's first entropy fix; any other of those listed in [64] (or similar) could have been used instead.

7

Time Discretization

In the present thesis, the discretization of time is of no particular interest in itself. It merely has to satisfy the following two basic requirements:

- Produce a consistent fully discrete conservation law; in particular, the spatial discretization error should be dominant for sufficiently small time-step sizes (Δt).
- Result in a stable numerical method when in combination with the spatial discretizations reviewed.

Temporal and spatial semi-discretizations are coupled together via the *method of lines* approach, previously brushed over in §3.3.1, now properly addressed in §7.1. Prime candidates for such a task are the so-called *strong stability preserving* (SSP) time schemes (§7.2); relevant schemes of this type are detailed in §7.3. Finally, a brief mention of alternatives present in the literature is made in §7.4.

7.1. The method of lines

Consider (2.1), the general conservation law in differential form. Up to this point, time has been a continuous variable; let its discrete counterpart be henceforth defined as the sequence $t_0 \leq t_1 \leq \cdots \leq t_N$ of $N+1$ discrete *time levels* (the first of which is assumed to correspond to a known initial state), such that $\Delta T \coloneqq t_N - t_0$ represents the total *simulated time span*—i.e. domain in the temporal dimension in which (2.1) is to be solved. ΔT is divided into N *time-steps*, not necessarily equal, such that $\Delta T = \sum_{n=1}^{N} \Delta t_n$. For ease of notation, I will omit the explicit indication that the time-step size is time level–dependent: $\Delta t_n \equiv \Delta t$.

Consider now the integral of (2.1) over an arbitrary time-step:

$$\int_{t_n}^{t_{n+1}} \left(\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{f}}{\partial x} \right) \mathrm{d}t = 0 \,, \tag{7.1}$$

by the fundamental theorem of calculus[1], it is equivalent to:

$$\boldsymbol{q}(t_{n+1}, x) - \boldsymbol{q}(t_n, x) = -\int_{t_n}^{t_{n+1}} \frac{\partial \boldsymbol{f}}{\partial x} \, \mathrm{d}t \,. \tag{7.2}$$

Thus, the change in the solution across a discrete time-step is equal to the time-integral of the spatial residuals (right-hand-side). This strategy is known as the *method of lines* [50, §8.3.2]. If we now introduce approximate solution and residual vectors into (7.2),

$$\boldsymbol{q}^h(t_{n+1}, x) = \boldsymbol{q}^h(t_n, x) + \int_{t_n}^{t_{n+1}} \boldsymbol{r}(t, x) \, \mathrm{d}t \,, \tag{7.3}$$

[1] If t and x are independent from each other, the following holds: $\int \frac{\partial q(t,x)}{\partial t} \, \mathrm{d}t = q(t,x) + C(x)$. For a definite integral, the x-dependent integration "constant" vanishes.

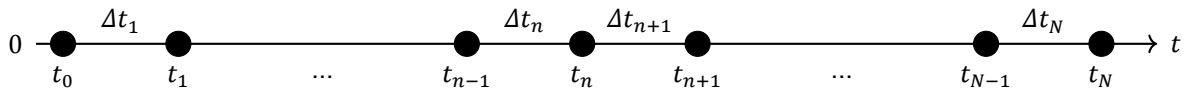


Figure 7.1: Time discretization and indexing convention used for it in this report.

it becomes clear that the discretization of time is completely independent from that of space.

All that is left is to evaluate the right-hand-side integral in (7.3) numerically—this is the job of the *time-integration scheme*. As an example, consider the *right* Riemann sum approximation:

$$\boldsymbol{q}^h(t_{n+1}, x) = \boldsymbol{q}^h(t_n, x) + \boldsymbol{r}(t_{n+1}, x)\Delta t\,. \tag{7.4}$$

This is the backward Euler method, a 1st order implicit time scheme. Only explicit schemes, in particular those of the Runge-Kutta family—see §7.3—, are used in this thesis' numerical experiments (part II of this report). These are methods which, unlike (7.4), require only the solution (and residuals) at the current time level (t_n) to approximate the solution at the next (t_{n+1}).

7.1.1. Courant number

The Courant number represents a dimensionless time-step size, equal to the ratio between physical and numerical information propagation speeds. In linear cases, the velocity of physical propagation of information is unique and unambiguous (e.g. c in the wave equation):

$$\varsigma \coloneqq \frac{c\Delta t}{\min\{\Delta x_k\}_{k=1}^{\mathrm{K}}}\,, \tag{7.5}$$

where the smallest Δx_k is chosen so that the resulting Courant number is largest (i.e. most restrictive[2]).

For the general case, consider an arbitrary element $\Omega_k \in \mathcal{T}^h$; the solution of the Riemann problem at its edges, $\partial\Omega_k^L$ and $\partial\Omega_k^R$, produces two sets of eigenvalues: $\{\lambda_{ik}^L\}_{i=1}^{\mathrm{I}}$ and $\{\lambda_{ik}^L\}_{i=1}^{\mathrm{I}}$ (respectively). The subsets $\lambda_{ik}^L > 0$ and $\lambda_{ik}^R < 0$ of the previous represent characteristic waves traveling *into* element Ω_k (from the left and right, respectively). The most restrictive of these two (left and right) defines the element's component-wise Courant number; similarly, the largest among components is selected as element-wise Courant number and, finally, the same is done mesh-wide. In other words:

$$\varsigma \coloneqq \max\{\varsigma_k\}_{k=1}^{\mathrm{K}}\,, \qquad \varsigma_k \coloneqq \max\{\varsigma_{ik}\}_{i=1}^{\mathrm{I}}\,, \qquad \varsigma_{ik} \coloneqq \frac{\Delta t}{\Delta x_k}\max\{\lambda_{ik}^+, -\lambda_{ik}^-\}\,; \tag{7.6}$$

with:

$$\lambda_{ik}^+ \coloneqq \max\{0, \lambda_{ik}^L\}\,, \qquad \lambda_{ik}^- \coloneqq \min\{0, \lambda_{ik}^R\}\,. \tag{7.7}$$

All results in this report use either a constant Δt, or a constant ς (or both, in linear cases).

7.1.2. Amplification factor

Some characteristics of numerical methods (such as stability) can be studied analytically for the linear case (see §A.4). Relevant for our current discussion of time schemes is that said methodology involves the linear spatial discretization being encoded into a linear operator (i.e. a matrix). The eigenvalues of the discrete spatial operator describe its influence on the numerical error. The temporal scheme acts on an arbitrary $z \in \mathbb{C}$—one of such spatial operator eigenvalues, associated with some component of the spatial discretization error. The actual values that z takes depend entirely on the spatial discretization and, as a scaling factor, the Courant number. Because of the independent treatment of time and space in the method of lines, z does *not* depend on the time scheme; the latter simply *amplifies or dampens* the error encoded in each z provided by the spatial scheme.

If the temporal discretization is itself linear, it is possible to define an *amplification factor function* $G : \mathbb{C} \to \mathbb{C}$ that quantifies the former's influence on an arbitrary error component in the complex plane (see §A.4.1 for details). Among the isocontours of $|G(z)|$ that join together the values of z that result

[2] If, for example, a method is stable for $\varsigma < \varsigma_{\max}$, this definition ensures that *all* elements are within stability bounds.

in certain amplification, the specific set of all z for which $|G(z)| = 1$ defines the boundary of the time scheme's *stability region*, i.e. the set of all z for which $|G(z)| < 1$. If the spatial discretization can be shown to restrict the values of z (possibly under some conditions on the time-step) so that they never lie outside of the time scheme's stability region, the numerical error is damped in every time step; the method is, therefore, *linearly stable*. The amplification factor of (7.4), for instance, is:

$$G(z) = \frac{1}{1-z}\,. \tag{7.8}$$

Amplification factor functions of the time schemes used in this report are included with their definitions in §7.3. Other examples can be found e.g. in [116, §B.3.2].

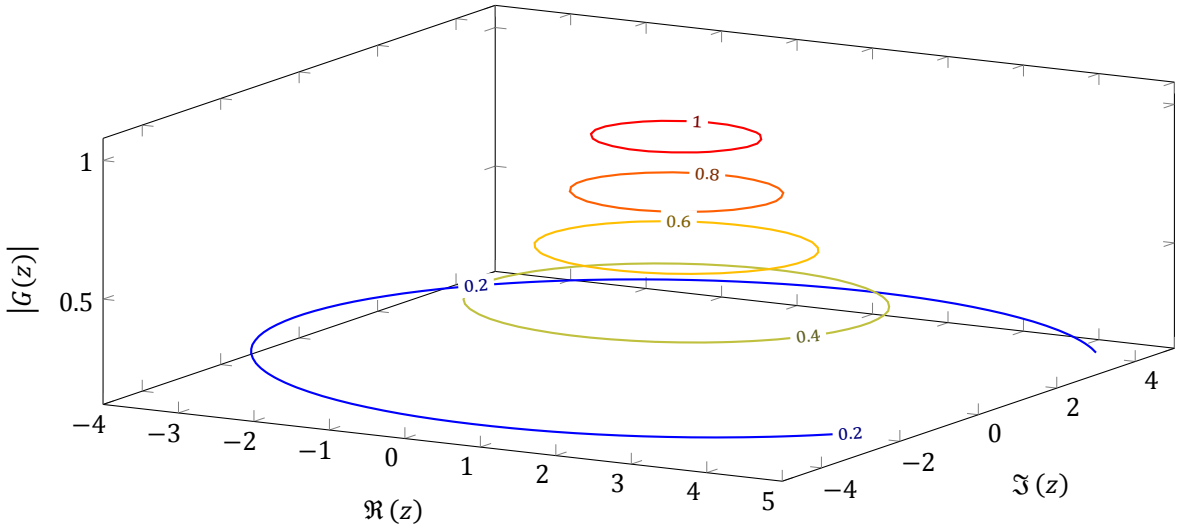


Figure 7.2: Iso-contours of $|G(z)|$ (i.e. magnitude of the amplification factor) of the implicit Euler scheme (7.4). Its stability region includes the entire complex plane except for the circle centered around $z = 1$ and radius 1. This scheme is unconditionally stable in combination with any spatial discretization for which $\Re(z) < 0$.

7.2. Strong stability preserving (SSP) time discretization

The method of lines leads to a separate characterization of spatial and temporal operators. Gottlieb and Shu [42] numerically showed that, even when a spatial discretization is proven to be stable in a total variation sense (see §8.1), the method resulting from its combination with a time discretization may turn out to be unstable in general. This is particularly important for discontinuous solutions of hyperbolic conservation laws, for which a combination of time and space discretizations which is linearly stable may become *unstable* [41]. Stability of a numerical method is addressed in §8. Ideas initially explored by Shu and Osher [109] eventually culminated in a class of so-called *strong stability preserving* time schemes[3].

Temporal discretizations of the SSP kind are high-order generalizations of the forward Euler method, which guarantee that "*the nonlinear stability properties satisfied by the spatial discretization when coupled with the forward Euler integration will be preserved when the same spatial discretization is coupled with these higher order methods*" [41, §1.1], provided that the time-step size is suitably restricted. In more precise terms, assuming that there exists a $\Delta t_{\max}$ such that:

$$\left\|\boldsymbol{q}^h(t_n,x) + \Delta t \boldsymbol{r}(t_n,x)\right\| \leq \left\|\boldsymbol{q}^h(t_n,x)\right\| \quad \text{for } 0 \leq \Delta t \leq \Delta t_{\max}\,, \tag{7.9}$$

where $\|\bullet\|$ is some convex functional (e.g. a norm or semi-norm), a given (single step) temporal scheme is said to be SSP if it produces a $\boldsymbol{q}^h(t_{n+1},x)$ such that:

$$\left\|\boldsymbol{q}^h(t_{n+1},x)\right\| \leq \left\|\boldsymbol{q}^h(t_n,x)\right\| \quad \text{whenever } \Delta t \leq c_{\text{SSP}}\Delta t_{\max}\,, \tag{7.10}$$

[3] Originally, these were known as TVD time schemes (by analogy with TVD spatial methods). Over time, this designation has been phased out in favor of the the broader (and more accurate) term, SSP.

for some $c_{\mathrm{SSP}} > 0$—its *SSP coefficient*. In the linear and constant coefficient case, the SSP coefficient is replaced in the previous by a so-called *threshold factor*; the latter is an upper bound on c_{SSP}, and is at most equal to the number of stages of the method [41, §1.4.1].

7.3. Explicit SSP Runge-Kutta methods (SSP-RK)

Most popular among explicit SSP time-integration methods is the Runge-Kutta family of schemes. These methods generalize Euler's method (1st order, explicit) to higher order through the addition of intermediate sub-steps between every two discrete time instants, such that the solution is advanced from an arbitrary time level t_n to the next, $t_{n+1} = t_n + \Delta t$, in S *stages*:

$$t_{n+\frac{s}{S}} \coloneqq t_n + \Delta t \frac{s}{S}, \quad \text{for } s = 1, 2, \dots, S \tag{7.11}$$

There is quite some flexibility in the way how each of these stages' results is combined to give the solution at the future time instant. Specific combinations are possible that ensure the SSP property; in fact, the first SSP schemes discovered (§7.2) were of this kind. One prominent usage of these time schemes is in the RKDG method of Cockburn and Shu (see §4). A selection of optimal[4] SSP-RK variants currently known is given next for completeness. Refer to [41, §2.1] for details.

Time scheme	Order of accuracy	Relative CPU cost/step	Relative memory cost/step	Effective SSP coefficient
SSP-RK1(1)	1	1	1	1
SSP-RK2(2)	2	2	2	0.5
SSP-RK3(3)	3	3	2	0.333...
SSP-RK4(5)	4	5	3	≈0.30
SSP-RK4(10)	4	10	2	0.6

Table 7.1: Comparison between various explicit SSP-RK methods considered optimal. The CPU cost of each method is proportional to the number of stages per time-step, S, while the memory cost is proportional to the total number of the degrees of freedom values stored simultaneously within a time-step. The effective SSP coefficient is $\frac{c_{\mathrm{SSP}}}{S}$. Optimal explicit SSP-RK methods beyond 4th order do not exist [41, §2.1].

7.3.1. SSP-RK1(1) or Euler's method: 1st order, 1 stage

Possibly the simplest way to approximate the integral in (7.2) is via a *left* Riemann sum:

$$\int_{t_n}^{t_{n+1}} \boldsymbol{r}(t, x)\, \mathrm{d}t \approx \boldsymbol{r}(t_n, x)\Delta t\,; \tag{7.12}$$

equation (7.3) then becomes:

$$\boldsymbol{q}^h(t_{n+1}, x) = \boldsymbol{q}^h(t_n, x) + \Delta t \boldsymbol{r}(t_n, x)\,, \tag{7.13}$$

which, written directly for the expansion coefficients of the solution and residual, is:

$$\hat{\boldsymbol{Q}}^1 = \hat{\boldsymbol{Q}}^0 + \Delta t \hat{\boldsymbol{\mathcal{R}}}^0\,, \tag{7.14}$$

where, for economy of notation, the following definitions are employed:

$$\hat{\boldsymbol{Q}}^s \coloneqq \hat{\boldsymbol{Q}}\left(t_{n+\frac{s}{S}}\right), \qquad \hat{\boldsymbol{\mathcal{R}}}^s \coloneqq \hat{\boldsymbol{\mathcal{R}}}\left(t_{n+\frac{s}{S}}\right). \tag{7.15}$$

Its amplification factor is simply:

$$G(z) = z + 1\,. \tag{7.16}$$

[4] Optimal here refers to possessing the largest SSP coefficient among all explicit SSP-RK methods of a given order and number of stages; see table 7.1.

This approach is known as *forward Euler method*, to which the particular case of a Runge-Kutta scheme with a single stage reduces. While attractive because of its simplicity, this method is only first order accurate. Moreover, its stability region is relatively small and includes no purely imaginary numbers. The latter implies that its combination with any non-diffusive spatial discretization (e.g. 2nd order central finite differences) is guaranteed to be *unconditionally unstable* (see figure 7.3 and e.g. [82, §4.5]).

7.3.2. SSP-RK2(2): 2nd order, 2 stages

Re-using (7.15), this scheme can be written as:

$$\hat{\boldsymbol{Q}}^1 = \hat{\boldsymbol{Q}}^0 + \Delta t \hat{\boldsymbol{\mathcal{R}}}^0 \,, \tag{7.17a}$$

$$\hat{\boldsymbol{Q}}^2 = \frac{1}{2}\hat{\boldsymbol{Q}}^0 + \frac{1}{2}\hat{\boldsymbol{Q}}^1 + \frac{1}{2}\Delta t \hat{\boldsymbol{\mathcal{R}}}^1 \,, \tag{7.17b}$$

and its amplification factor is:

$$G(z) = \frac{z^2}{2} + z + 1 \,. \tag{7.18}$$

Despite having second order accuracy, the fact that this scheme uses two stages means that the number of operations within a time step is twice that of SSP-RK1(1). Moreover, its SSP coefficient remains the same, $c_{\mathrm{SSP}} = 1$ (reducing the effective SSP coefficient to one half), and it requires storing all the degrees of freedom of the discretization at two time levels simultaneously. On the positive side, the stability domain of this method is such that a finite portion of the imaginary axis is in its boundary of marginal stability.

7.3.3. SSP-RK3(3): 3rd order, 3 stages

This is one of the most popular time schemes of this type, perhaps due to it being an "optimum among optima" in the sense of it achieving 3rd order with only 3 stages (the number of stages starts growing beyond the scheme's order, from here on) and being the lowest order method to include part of the imaginary axis *inside* its stability domain.

$$\hat{\boldsymbol{Q}}^1 = \hat{\boldsymbol{Q}}^0 + \Delta t \hat{\boldsymbol{\mathcal{R}}}^0 \,, \tag{7.19a}$$

$$\hat{\boldsymbol{Q}}^2 = \frac{3}{4}\hat{\boldsymbol{Q}}^0 + \frac{1}{4}\hat{\boldsymbol{Q}}^1 + \frac{1}{4}\Delta t \hat{\boldsymbol{\mathcal{R}}}^1 \,, \tag{7.19b}$$

$$\hat{\boldsymbol{Q}}^3 = \frac{1}{3}\hat{\boldsymbol{Q}}^0 + \frac{2}{3}\hat{\boldsymbol{Q}}^2 + \frac{2}{3}\Delta t \hat{\boldsymbol{\mathcal{R}}}^2 \,. \tag{7.19c}$$

Its amplification factor is:

$$G(z) = \frac{z^3}{6} + \frac{z^2}{2} + z + 1 \,. \tag{7.20}$$

The cost per time-step is now three times that of forward Euler. Nevertheless, the theoretical memory cost is still only twice that of the aforementioned (only two different time levels appear in each stage update).

7.3.4. SSP-RK4(5): 4th order, 5 stages

Unfortunately, there is no optimal SSP-RK method with 4th order and only 4 stages [41, p. 271]; instead, at least $\mathrm{S} = 5$ are required. The optimal SSP-RK scheme of 5 stages and 4th order is:

$$\hat{\boldsymbol{Q}}^1 = \hat{\boldsymbol{Q}}^0 + 0.391752226571890\Delta t \hat{\boldsymbol{\mathcal{R}}}^0 \,, \tag{7.21a}$$

$$\hat{\boldsymbol{Q}}^2 = 0.444370493651235\hat{\boldsymbol{Q}}^0 + 0.555629506348765\hat{\boldsymbol{Q}}^1 + 0.368410593050371\Delta t \hat{\boldsymbol{\mathcal{R}}}^1 \,, \tag{7.21b}$$

$$\hat{\boldsymbol{Q}}^3 = 0.620101851488403\hat{\boldsymbol{Q}}^0 + 0.379898148511597\hat{\boldsymbol{Q}}^2 + 0.251891774271694\Delta t \hat{\boldsymbol{\mathcal{R}}}^2 \,, \tag{7.21c}$$

$$\hat{\boldsymbol{Q}}^4 = 0.178079954393132\hat{\boldsymbol{Q}}^0 + 0.821920045606868\hat{\boldsymbol{Q}}^3 + 0.544974750228521\Delta t \hat{\boldsymbol{\mathcal{R}}}^3 \,, \tag{7.21d}$$

$$\begin{aligned}\hat{\boldsymbol{Q}}^5 = {} & 0.517231671970585\hat{\boldsymbol{Q}}^2 + 0.096059710526147\hat{\boldsymbol{Q}}^3 + 0.063692468666290\Delta t \hat{\boldsymbol{\mathcal{R}}}^3 \\ & + 0.386708617503269\hat{\boldsymbol{Q}}^4 + 0.226007483236906\Delta t \hat{\boldsymbol{\mathcal{R}}}^4 \,,\end{aligned} \tag{7.21e}$$

with the following amplification factor:

$$G(z) \approx 0.00448\, z^5 + 0.0417\, z^4 + 0.167\, z^3 + 0.5\, z^2 + z + 1\,. \tag{7.22}$$

This scheme has a larger $c_{\mathrm{SSP}} \approx 1.508$, making it only slightly more CPU-intensive than SSP-RK3 per time-step (but an entire order more accurate). It requires at least one additional memory register, however, and has irrational coefficients.

7.3.5. SSP-RK4(10): 4th order, 10 stages

Lastly, this is an alternative to (7.21) which goes up to $\mathrm{S} = 10$, reaching $c_{\mathrm{SSP}} = 6$. It can be proven optimal analytically (its SSP coefficient is equal to its threshold factor, thus reaching its theoretical upper bound), and recovers the simple rational coefficients of the first three schemes in the family. Also, it can be implemented such that each time step requires no more memory than SSP-RK2(2) and SSP-RK3(3) [41, p. 274].

$$\hat{\boldsymbol{Q}}^s = \hat{\boldsymbol{Q}}^{s-1} + \frac{1}{6}\Delta t \hat{\boldsymbol{\mathcal{R}}}^{s-1}, \quad s = 1, 2, 3, 4, 6, 7, 8, 9\,, \tag{7.23a}$$

$$\hat{\boldsymbol{Q}}^5 = \frac{3}{5}\hat{\boldsymbol{Q}}^0 + \frac{2}{5}\hat{\boldsymbol{Q}}^4 + \frac{1}{15}\Delta t \hat{\boldsymbol{\mathcal{R}}}^4\,, \tag{7.23b}$$

$$\hat{\boldsymbol{Q}}^{10} = \frac{1}{25}\hat{\boldsymbol{Q}}^0 + \frac{9}{25}\hat{\boldsymbol{Q}}^4 + \frac{3}{5}\hat{\boldsymbol{Q}}^9 + \frac{3}{50}\Delta t \hat{\boldsymbol{\mathcal{R}}}^4 + \frac{1}{10}\Delta t \hat{\boldsymbol{\mathcal{R}}}^9\,. \tag{7.23c}$$

Its amplification factor function is:

$$G(z) = \frac{z^{10}}{251942400} + \frac{z^9}{4199040} + \frac{z^8}{155520} + \frac{z^7}{9720} + \frac{7\,z^6}{6480} + \frac{17\,z^5}{2160} + \frac{z^4}{24} + \frac{z^3}{6} + \frac{z^2}{2} + z + 1\,. \tag{7.24}$$

7.4. Alternative time discretization schemes

Strong stability preserving Runge-Kutta methods are not, by any means, the only option when performing the integration of a semi-discrete conservation law. Other popular Runge-Kutta variants include low-storage (LSRK) [95] and low dissipation–low dispersion (LDDRK) [52]. Moreover, an entirely independent class of high-order time integration schemes are the *linear multistep methods*, some variants of which can also be SSP [41]. All these are left outside of the scope of this thesis. The same applies to *implicit* time schemes of any kind.

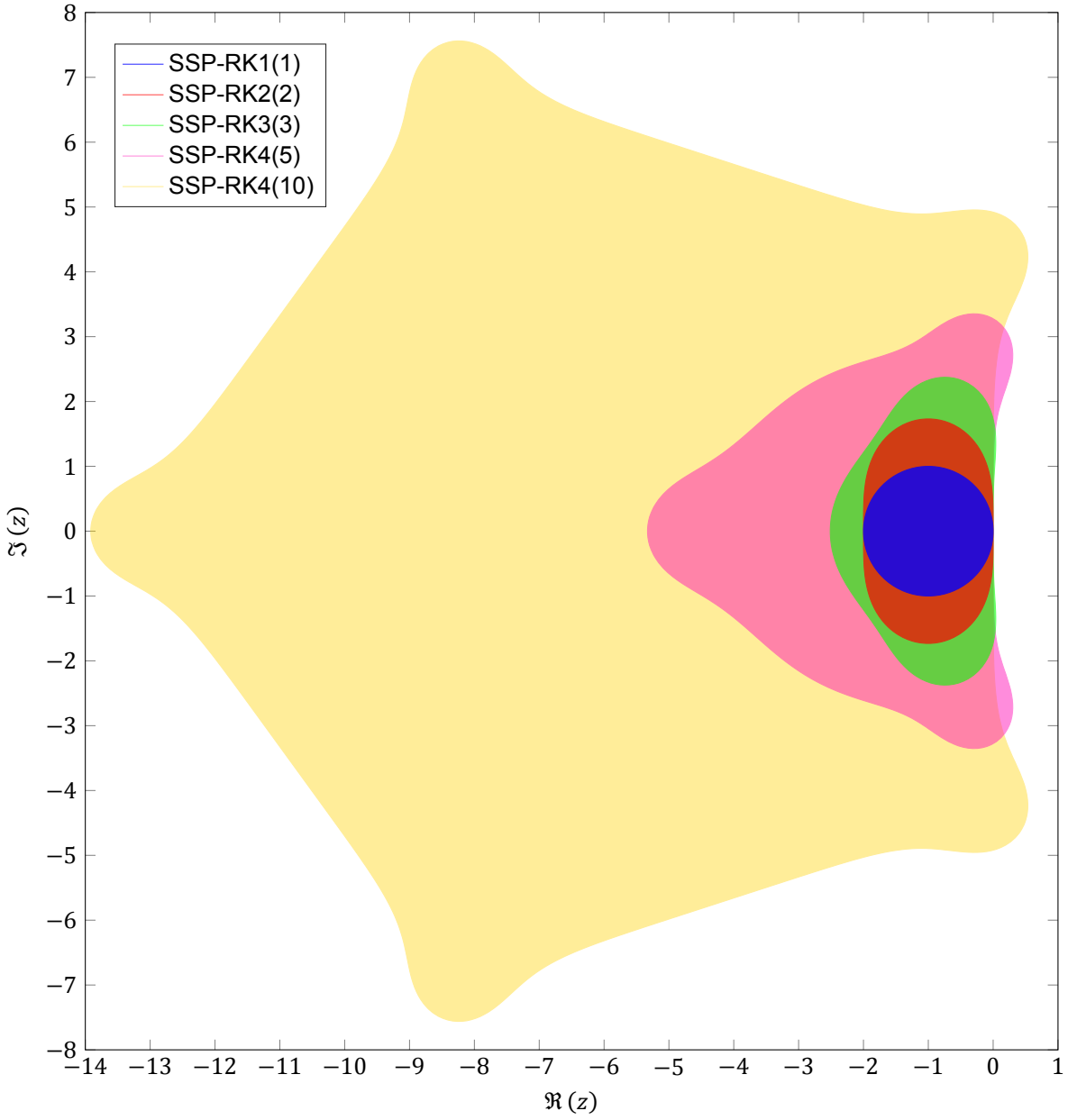


Figure 7.3: Domain of stability, i.e. set of all $z \in \mathbb{C}$ where $\left|G(z)\right| \leq 1$, of each time scheme in §7.3.

8

Nonlinear Stabilization

The approximate solution in discontinuous methods is allowed to have jump discontinuities at element edges. The larger these jumps are, the larger the error associated with the numerical flux across them becomes. This error typically[1] has a diffusive nature, and so the discretization will implicitly add more numerical diffusion the less resolved the solution is—a convenient negative feedback loop. This is the main stabilization mechanism of all discontinuous high-order schemes, and it is sufficient to guarantee linear stability (when combined with an appropriate time discretization). Nevertheless, it is not enough to ensure stability for the high-order case in general.

Godunov showed with his *barrier theorem* of 1959 (see [50, §8.3.3]) that only first-order linear schemes for the advection equation can be monotone. Consequently, any linear high-order discretization (DGSEM, FR/CPR and DGIGA included) will result in nonphysical oscillations appearing in the approximate solution whenever its exact counterpart has a discontinuity—by the Gibbs phenomenon. These wiggles will prevent the discretization from retaining its formal order of accuracy. Moreover, for PDEs in which certain state quantities are physically bounded (e.g. density and total energy being non-negative in the Euler equations), they may cause the method to diverge. It is for this reason that a discontinuity capturing strategy is required in every of the methods addressed in this work.

Shock capturing consists on modeling a discontinuous feature of the solution (e.g. a shock wave, which in an inviscid context could only be represented accurately with an infinitely fine discretization) via a discrete solution feature of finite thickness that approximates it reasonably well, in a way that does not hinder accuracy elsewhere in the domain. Seen at the higher level, shock capturing adds nonlinear steps to the high-order discretization so that the premise of Godunov's theorem no longer applies. Said nonlinearity may introduce considerable complexity to the discretization.

The literature on stabilization methods for hyperbolic conservation laws is vast (figure 8.1). In this chapter, I describe a selection of six limiters and two sensors; the performance of these is later studied in part II. The goal here is to provide all necessary details to facilitate an eventual reproduction of any such results. This choice of sensors and limiters is motivated, to a large extent, by the comparisons in [102, 124, 135].

8.1. Total variation stability

Linear stability[2] of a numerical method may be defined as the requirement that, for some simulated time span $\Delta T \equiv t_N - t_0$ (see §7.1) [82, §8.3.2]:

$$\|\boldsymbol{S}^n\| \leq C, \text{ for all } n \leq \frac{\Delta T}{\Delta t}, \tag{8.1}$$

where C is a constant and $\boldsymbol{S}$ is the matrix (i.e. linear operator) that advances the approximate solution from $\boldsymbol{q}^h(t,x)$ to $\boldsymbol{q}^h(t+\Delta t,x)$—i.e. some norm of the nth power of $\boldsymbol{S}$ is uniformly bounded from an initial time t_0 up to a final time t_N. The basics of linear stability analysis via the Von Neumann (or

[1] This is the case for all upwind numerical fluxes, such as those given by any of the Riemann solvers used in this report.
[2] Sometimes referred to as Lax-Richtmyer stability.

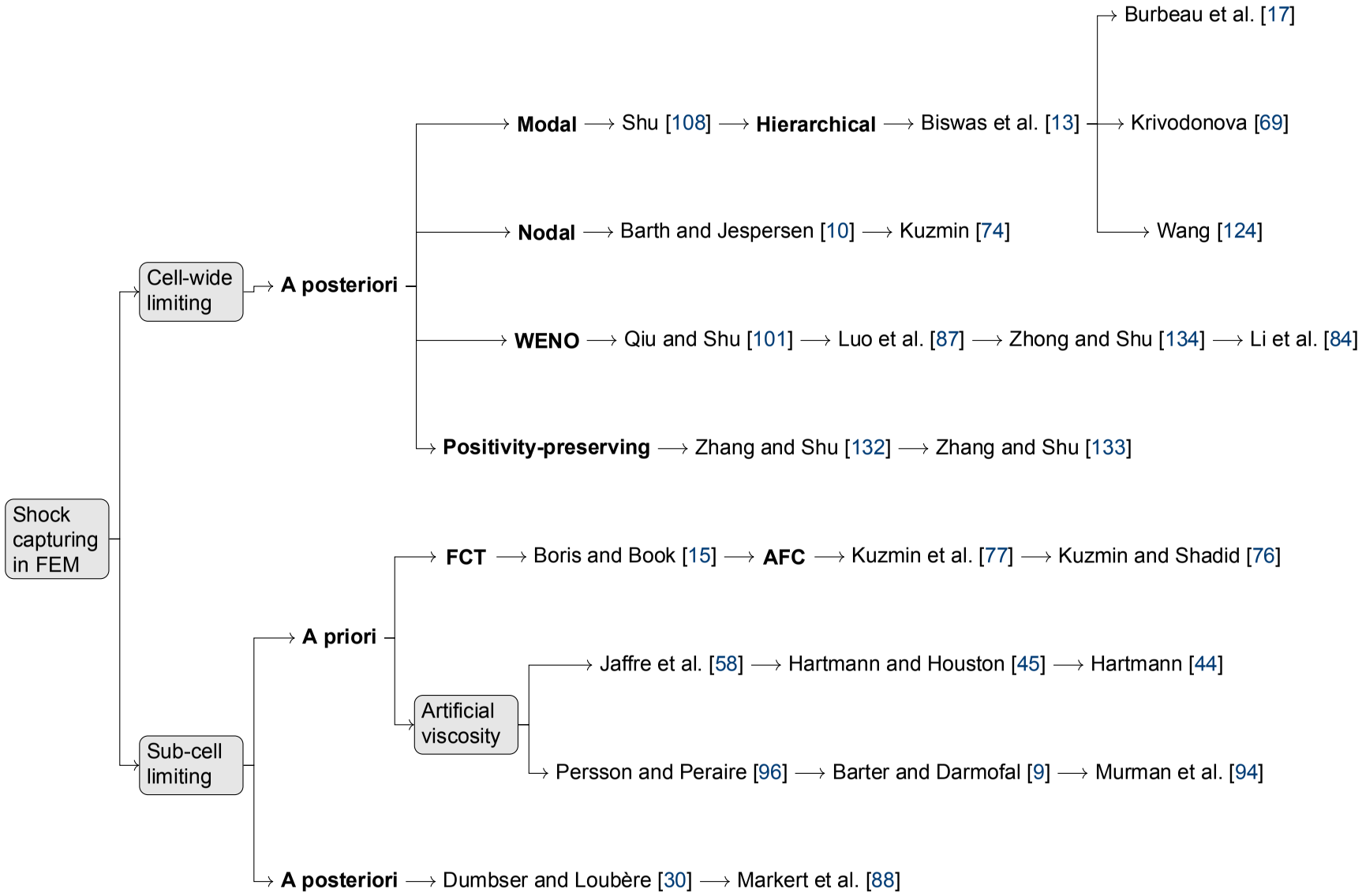


Figure 8.1: Overview of some of the main nonlinear stabilization families that have been applied to high-order methods in the literature.

Fourier) method for compact high-order spatial discretizations and explicit Runge-Kutta time schemes are covered in appendix A.

Nonlinear stability refers instead to a nonlinear numerical method, regardless of the continuous PDE that it tries to approximate being linear or not (e.g. if limiting is employed, the method itself may be nonlinear even for a linear PDE). The previous notion of stability is lost, since $\boldsymbol{S}$ no longer exists as a matrix; in fact, no linear stability criterion is enough to ensure convergence in such cases [82, §8.3.5]. Intuitively, it is clear that even if the approximate solution remains bounded e.g. in terms of energy (L^2 norm), it can still experience nonphysical oscillations. Such spurious wiggles may be quantified through the so-called *total variation* of the approximate solution, which can be used to define sufficient criteria for nonlinear stability [82, §12.12].

The total variation acts similarly to a norm—it is a scalar associated with the (exact or approximate) solution over the entire domain at a given time instant—measuring *how oscillatory* a given function is. It is defined, for a scalar function $q(t,x)$ at some arbitrary time instant, as [82, §6.7]:

$$\left\|q(t,x)\right\|_{\mathrm{TV}} \coloneqq \sup \sum_{n=1}^{N} \left|q(t,x_n) - q(t,x_{n-1})\right| , \tag{8.2}$$

where $\sup$ indicates the *supremum*[3] operator, taken over *all* partitions of the real line. This quantity can only be finite if $q(t,x)$ approaches constant values as $x \to \pm\infty$ (even so, in some cases—e.g. if the solution is periodic—it may be meaningful to define the total variation over a subset of the real line instead). If $q(t,x)$ is differentiable[4], (8.2) is equivalent to:

$$\left\|q(t,x)\right\|_{\mathrm{TV}} = \int_{-\infty}^{\infty} \left|\frac{\partial q}{\partial x}(t,x)\right| \mathrm{d}x \, . \tag{8.3}$$

In the vector case, (8.2) and (8.3) can be evaluated either by replacing the absolute value with a vector norm, or, alternatively, by measuring the total variation in characteristic variables [82, §15.8].

In all results shown in this report, I compute the approximate total variation of a given function (be it continuous or not) by evaluating (8.2) for increasing numbers of sample points within each $\Omega_k \in \mathcal{T}^h$. For each N, a total variation estimate is obtained; N is doubled (and the new sample points obtained via bisection) iteratively, until the sequence of resulting estimates converges (to an absolute tolerance of 1×10^{-10}). To speed up this series convergence process, I employ Richardson extrapolation [12, p. 375]. In non-scalar conservation laws, all norm-like quantities—TV included—are applied conserved variable–wise to the solution (or error) vectors.

8.1.1. Total variation diminishing (TVD)

It can be shown that, even in the nonlinear *scalar* case, if the exact initial condition $q(t_0,x) = q^0(x)$ has finite total variation, the total variation of the exact solution does not increase in time [114, §13.6.1]:

$$\left\|q(t,x)\right\|_{\mathrm{TV}} \leq \left\|q(t_0,x)\right\|_{\mathrm{TV}} \text{ for all } t > t_0 \, , \tag{8.4}$$

the previous is a condition known as *total variation diminishing* (TVD); it should be pointed out that the total variation of such a function is actually *non-increasing* (rather than strictly diminishing) as time progresses. It is possible to show that in linear systems of conservation laws, under a definition of the total variation based on characteristic variables, the total variation of the exact solution remains constant [82, §15.8.1]. In the nonlinear system case, the exact solution is not TVD even in such a definition—even in one dimension [82, §15.8.2].

Typically, the total variation of the (scalar) exact solution remains constant until (in the Burgers case) said solution develops into a shockwave—it then starts diminishing (see, for example, the results in [83, §3]). This property of the exact solution becomes one additional aspect to be mimicked by the discrete model. A numerical method is said to be TVD if the approximate solution it produces satisfies (8.4).

[3] The supremum of a set is the least upper bound of all its elements; it generalizes the concept of maximum. For example, the set of all negative real numbers has no maximum—there is no element within the set (i.e. a negative real number) larger than any other—yet, it has a supremum: zero.

[4] If $\frac{\partial q}{\partial x}$ is interpreted as the distribution derivative (i.e. constructed using Dirac Delta functions at each discontinuity point), (8.3) applies even for non-differentiable functions [82, §6.7].

The 1st order upwind finite volume spatial discretization (or, equivalently, any of the three high-order methods reviewed in this report with $\mathrm{p} = 0$), combined with the explicit 1st order Euler time scheme, is the simplest example of a TVD method.

8.1.2. Monotonicity

The onset of Gibbs oscillations in an approximate solution will be accompanied by an increase in its total variation. In fact, any approximate solution obtained by a fully discrete conservation law that does *not increase* the total variation to the function it is applied to is necessarily *monotonicity-preserving* [82, p. 110]. This, in turn, means that for *all* $x_n < x_{n+1} \in \Omega$ (two arbitrary sample locations in the domain) and $t_1 > t_0$:

$$q^h(t_0, x_n) \geq q^h(t_0, x_{n+1}) \implies q^h(t_1, x_n) \geq q^h(t_1, x_{n+1})\,. \tag{8.5}$$

In a monotonicity-preserving solution, no new local extrema can form as time advances. Moreover, in the case of non-increasing total variation, any existing local extrema can only become less pronounced—i.e. minima cannot decrease and maxima cannot increase. A TVD method, therefore, guarantees that the approximate solution it provides will not achieve invalid values that could cause the numerical solver routine to crash (e.g. negative densities or imaginary speeds of sound). There is, however, an important *disadvantage* associated with a scheme being TVD: its accuracy in local extrema, even smooth ones, can be at most 2nd order—regardless of its order in monotone smooth regions [108].

8.1.3. Total variation bounded (TVB)

An approximate solution component is said to be *total variation bounded* (TVB) [108] in $[t_0, t_N]$ if it satisfies:

$$\left\|q_i^h(t_n, x)\right\|_{\mathrm{TV}} \leq B \text{ for all } t_0 \leq t_n \leq t_N\,, \tag{8.6}$$

where $B > 0$ is a constant that *only* depends on $\left\|q_i^h(t_0, x)\right\|_{\mathrm{TV}}$—i.e. that is unique for all possible n and Δt such that $t_n = t_0 + n\Delta t$. Comparing (8.6) and (8.4), it is clear that TVD implies TVB. Total variation stability of a method is guaranteed by it being TVB [108], [82, p. 250]; even in the nonlinear system case—if a suitable TV definition is employed, see [26, p. 95].

8.1.4. Other nonlinear stability criteria

Even TVB is often too restrictive for nonlinear high-order discretizations—in the sense that, although a given method may seem to be stable and convergent in practice, no actual TVB proof is known for it. Such a qualitative notion of nonlinear stability is paradigmatic of *essentially non-oscillatory* (ENO) methods; see §8.6.

It is typical for the total variation of a high resolution–high order approximate solution to oscillate around a TV value slightly lower that of its exact counterpart, and converge towards it (from below) as the number of degrees of freedom increases [83, §3]. Such behavior is suggestive of TV boundedness as long as $\left\|q^h\right\|_{\mathrm{TV}} \leq \|q\|_{\mathrm{TV}}$ for all t, and can be attributed to a combination of:

- The total variation $\left\|q^h\right\|_{\mathrm{TV}}$ being inexact, as it is evaluated at a finite number of sample points in a subset of the real line.
- The discrete samples of the approximate solution, used in the TV estimation, not being exact themselves.

Stability criterion	Monotonicity-preserving	High-order (in smooth regions)	Non-linearly stable
TVD	✓	✗	✓
TVB	✗	✓	✓
ENO	✗	✓	✗
Linearly stable	✗	✓	✗

Table 8.1: Summary of the various stability criteria and the properties of the approximate solution that they guarantee (scalar one-dimensional case). Cross markers (✗) indicate that a given stability criterion is *insufficient* to ensure the corresponding property.

8.2. Legendre-based limiting

Most limiters incorporated in the present thesis (all except §8.7), as well as the two shock sensors in §8.3, are defined for a modal DG discretization (§4.2.1). This means that the expansion coefficients of the approximate solution and flux are assumed to be associated with a Legendre polynomial basis, spanning each element's trial solution space $S_k^h\left(\widetilde{\Omega}\right)$. These are obtained via L^2 projection from the actual J-dimensional trial function space—be it Lagrange polynomials (DGSEM and FR/CPR) or B-splines (DGIGA)—onto a Legendre-based destination one with the same number of dimensions, as follows. Once the limiting/sensing procedure is complete, the approximate solution is projected back onto its original basis.

Due to the orthogonality of the Legendre basis, any higher-order expansion coefficients that may seem undefined (e.g. because an element is using a lower approximation degree than its neighbors) can be assigned values of zero without modifying the approximate solution. This facilitates limiting of discretizations employing p-refinement.

Within this chapter, $\hat{\boldsymbol{Q}}_k$ and $\check{\boldsymbol{Q}}_k$ are redefined as the matrices of Legendre and "original" (Lagrange or B-spline) expansion coefficients, respectively. Assume that destination ($\mathcal{P}_{j-1}$, Legendre) and source (ϕ_j, Lagrange or B-spline) basis functions are arranged into vectors, as in (3.9). Then, the state coefficients of the two approximate solution representations are related by:

$$\hat{\boldsymbol{Q}}_k(t)\int_{-1}^{1}\boldsymbol{\mathcal{P}}\boldsymbol{\mathcal{P}}^{\intercal}\,\mathrm{d}\xi=\check{\boldsymbol{Q}}_k(t)\int_{-1}^{1}\boldsymbol{\phi}\boldsymbol{\mathcal{P}}^{\intercal}\,\mathrm{d}\xi\,. \tag{8.7}$$

For DGSEM and FR/CPR, the previous reduces to (4.19), i.e. :

$$\hat{\boldsymbol{Q}}_k(t)=\check{\boldsymbol{Q}}_k(t)\boldsymbol{\mathcal{V}}^{-1}\,, \tag{8.8}$$

while, for DGIGA—using (4.7):

$$\hat{\boldsymbol{Q}}_k(t)=\check{\boldsymbol{Q}}_k(t)\left(\int_{-1}^{1}\boldsymbol{N}\boldsymbol{\mathcal{P}}^{\intercal}\,\mathrm{d}\xi\right)\begin{bmatrix}\frac{1}{2} & & & 0\\ & \frac{3}{2} & & \\ & & \ddots & \\ 0 & & & \frac{2\mathrm{J}-1}{2}\end{bmatrix}, \tag{8.9}$$

with each entry in the $\mathbb{R}^{\mathrm{J}\times\mathrm{J}}$ mixed inner product matrix being computed exactly via Gauss-Legendre quadrature of $\mathrm{J}+1$ points at the breakpoint span level (see §6.3.2).

DGSEM and FR/CPR employ a polynomial trial function space. Hence, for these two methods, the projection to and from Legendre is a mere change of basis; the approximate solution is unaffected. In DGIGA, however, the smoothness of the approximate solution will generally not be preserved. Furthermore, the degree of the Legendre-based version of the solution will be equal or higher than that of its B-spline-based version (the overall number of degrees of freedom is kept the same).

8.3. Discontinuity sensing

Limiting adds non-negligible computational overhead to each residual evaluation; in addition, the accuracy in limited elements is generally reduced. It would therefore be desirable to apply any given limiter only to the smallest set of elements which require it. Shock sensors (also known as troubled cell detectors) try to determine *a priori* which elements contain discontinuities and/or will result in undesirable spurious oscillations unless limited. Two sensors have been selected to complement the limiters reviewed in the present study.

8.3.1. KXRCF sensor

Krivodonova et al. [68] propose a sensor that estimates the local order of accuracy of a DG discretization at the inflow edge(s) of each element, and uses this information to determine whether or not to mark it for limiting. It exploits the fact that the DG approximate solution is $\mathcal{O}(\Delta x^{2\mathrm{p}+1})$ and $\mathcal{O}(\Delta x^{\mathrm{p}+2})$ superaccurate at smooth outflow and inflow element boundaries, respectively [4]. This shock detector is both simple and free of user-defined parameters, and seems to be well-regarded in the literature [102].

The implementation I use in this thesis, slightly adapted from [68], is as follows:

1. Consider an arbitrary $\Omega_k \in \mathcal{T}^h$. Evaluate a PDE-dependent, velocity-like quantity[5] at each of its edges; let us denote these here as u_k^L and u_k^R. Then, use them to compute the indicator variable I_k as follows:

   (a) Initialize it to $I_k = 0$.

   (b) If $u_k^L > 0$ (the left edge is an inflow boundary):

   $$I_k \leftarrow I_k + \left|\widetilde{q}^h_{1\,k}(-1) - \widetilde{q}^h_{1\,k-1}(1)\right| . \tag{8.10}$$

   (c) If $u_k^R < 0$, i.e. the right edge is (also) an inflow boundary:

   $$I_k \leftarrow I_k + \left|\widetilde{q}^h_{1\,k}(1) - \widetilde{q}^h_{1\,k+1}(-1)\right| . \tag{8.11}$$

   For vector PDEs, I only consider the first state vector component (e.g. density for Euler).

2. Normalize it by a baseline convergence rate:

   $$I_k \leftarrow \frac{I_k}{|\hat{q}_{1\,1\,k}|} \Delta x_k^{-\frac{\mathrm{p}+1}{2}} . \tag{8.12}$$

3. It is shown in [68] that $I_k \to 0$ as either $\Delta x_k \to 0$ or $\mathrm{p} \to \infty$ if $q_1(x) \in C^\infty$ (locally), while $I \to \infty$ near regions where $q_1(x) \notin C^0$. Therefore, if $I_k > 1$ (once normalized), assume that Ω_k is *troubled* (i.e. it contains a discontinuity and/or needs to be limited).

8.3.2. AP-TVD sensor

This second sensor is due to Wang [124]. He argues that KXRCF has too much of a tendency to result in false positives, a problem that this alternative marker is designed to correct. It is based on the TVD generalized slope limiter of Cockburn and Shu (§8.4), but tries to avoid mistaking smooth extrema for spurious oscillations caused by discontinuities without requiring user-defined parameters (similarly to §8.5), hence its "accuracy-preserving" designation.

I use it in a slightly modified way from [124], which can be summarized as follows:

1. Consider $\Omega_k \in \mathcal{T}^h$ and the first, second and third Legendre coefficients of its approximate solution vector ($\hat{\boldsymbol{q}}_{1\,k}$, $\hat{\boldsymbol{q}}_{2\,k}$ and $\hat{\boldsymbol{q}}_{3\,k}$, respectively associated with $\mathcal{P}_0$, $\mathcal{P}_1$ and $\mathcal{P}_2$). This element is preemptively assumed to require limiting if:

   $$\widetilde{q}^h_{ijk}(\xi_n) > 1.001 \max\{\hat{\boldsymbol{q}}_{1\,k-1}, \hat{\boldsymbol{q}}_{1\,k}, \hat{\boldsymbol{q}}_{1\,k+1}\} \quad \text{or} \quad \widetilde{q}^h_{ijk}(\xi_n) < 0.999 \min\{\hat{\boldsymbol{q}}_{1\,k-1}, \hat{\boldsymbol{q}}_{1\,k}, \hat{\boldsymbol{q}}_{1\,k+1}\} \tag{8.13}$$

   for any $i \in [1, \mathrm{I}]$, $j \in [1, \mathrm{J}]$ and $n \in [1, \mathrm{J}+2]$, where $\xi_n \in \{-1, \xi_2, \xi_3, \ldots, \xi_{\mathrm{J}+1}, 1\}$, the set of coordinates associated with the degrees of freedom (nodes or control points), augmented with the left and right edge locations.

2. If Ω_k was marked as troubled according to the previous step, compute:

   $$\hat{\boldsymbol{q}}^*_{3\,k} \coloneqq \mathrm{minmod}\left(\hat{\boldsymbol{q}}_{3\,k}, \frac{\hat{\boldsymbol{q}}_{2\,k} - \hat{\boldsymbol{q}}_{2\,k-1}}{3}, \frac{\hat{\boldsymbol{q}}_{2\,k+1} - \hat{\boldsymbol{q}}_{2\,k}}{3}\right) \tag{8.14}$$

   and, overriding step 1, Ω_k is considered free of troubles if $\hat{\boldsymbol{q}}_{3\,k} = \hat{\boldsymbol{q}}^*_{3\,k}$.

The second step, in which smooth extrema are unmarked, is done comparing element-averaged derivatives of the approximate solution in [124]. My alternative formulation—using Legendre coefficients instead—is justified solely on its similarity with hierarchical limiters (§8.5); there seems to be no reason why, it being effective in that situation, should no longer be so in this context.

[5] Advection: a; wave: c; Burgers: $\frac{u}{2}$; Euler: u.

8.4. Generalized slope limiting

Cockburn and Shu [23], aware of the limitations of their novel method in regards to nonlinear stability, already included a simple (yet very effective) limiter in their original proposal of the Runge-Kutta Discontinuous Galerkin scheme. This limiter is applied to each troubled element (see §8.3) at the end of every Runge-Kutta stage. Its role is to modify its Legendre coefficients $\hat{\boldsymbol{Q}}_k(t)$ (in fact, it only limits the columns for which $j > 1$) in such a way that any spurious oscillations are removed from $\boldsymbol{q}_k^h(t,x)$.

The DG slope limiter of Cockburn and Shu borrows a minmod-based[6] TVD slope limiter from MUSCL-type high-resolution finite volume schemes [82, §6.9] to instead enforce the weaker TVB criterion on the limited approximate solution.

For a piece-wise constant approximate solution, (8.2) reduces to being evaluated at each element's mean value. This generalizes into a weaker stability indicator for high-order methods—TVM rather than TV—but one that is much simpler (and economical) to evaluate [22, p. 424]:

$$\left\| q^h(t_1,x) \right\|_{\mathrm{TVM}} \coloneqq \sum_{k=1}^{\mathrm{K}+1} \left| \hat{q}_{1\,k}(t_1) - \hat{q}_{1\,k-1}(t_1) \right| , \tag{8.15}$$

where $\hat{q}_{1\,k}$ is the first Legendre coefficient of the approximate solution on Ω_k (see §8.2). The slope-limited DG method is proven in [22] to satisfy the corresponding TVDM and TVBM criteria (when $M = 0$ and $M > 0$, respectively; see §8.4), when combined with a *strong stability–preserving Runge-Kutta* (SSP-RK) time scheme (see §7) for a small enough[7] Courant number. Both TVDM and TVBM imply TVB [26, p. 95]—this is one of the very few general stability results known for high-order methods.

8.4.1. Modified minmod function

Shu's element-wise *modified minmod* function is defined as:

$$\operatorname{mod}\underset{k}{\operatorname{minmod}}\left(q_1,q_2,q_3\right) \coloneqq \begin{cases} q_1 & \text{if } |q_1| \leq M\Delta x_k^2 \\ \operatorname{minmod}\left(q_1,q_2,q_3\right) & \text{otherwise} \end{cases} , \tag{8.16}$$

with the conventional minmod function being:

$$\operatorname{minmod}\left(q_1,q_2,q_3\right) \coloneqq \begin{cases} \operatorname{sign}(q_1)\min\{q_1,q_2,q_3\} & \text{if } \operatorname{sign}(q_1) = \operatorname{sign}(q_2) = \operatorname{sign}(q_3) \\ 0 & \text{otherwise} \end{cases} . \tag{8.17}$$

It is convenient to *vectorize* these functions, such that:

$$\operatorname{mod}\underset{k}{\operatorname{minmod}}\left(\boldsymbol{q}_1,\boldsymbol{q}_2,\boldsymbol{q}_3\right) = \begin{bmatrix} \operatorname{mod}\operatorname{minmod}_k\left(q_{1\,1},q_{1\,2},q_{1\,3}\right) \\ \operatorname{mod}\operatorname{minmod}_k\left(q_{2\,1},q_{2\,2},q_{2\,3}\right) \\ \vdots \\ \operatorname{mod}\operatorname{minmod}_k\left(q_{\mathrm{I}\,1},q_{\mathrm{I}\,2},q_{\mathrm{I}\,3}\right) \end{bmatrix} . \tag{8.18}$$

The scalar $M \geq 0$ is a user-specified parameter[8]. For $M = 0$, the limiter enforces TVD slopes on any element which contains a *critical point* (i.e. location where the first derivative of the solution either is zero or does not exist). This makes $\boldsymbol{q}^*(t,x)$ TVDM (which implies TVB) but has the drawback of greatly diffusing local maxima or minima, even if they are smooth, needlessly reducing accuracy to low order in their neighborhood. By gradually increasing M, it is possible to reduce the activation threshold of the limiter so that smooth extrema are excluded, and the limiter only diffuses actual discontinuities (as intended). The resulting solution is TVBM (and, still, TVB) for $M > 0$,. As $M \to \infty$, limiting stops.

[6] The minmod slope limiter can be regarded as the baseline TVD limiter, since it results in the most conservative slopes—i.e. introduces the most numerical diffusion—among similar alternatives (e.g. superbee, MC, van Leer; see [82, §6.12]). Cockburn and Shu's generalized slope limiter is actually built upon a slightly less restrictive one due to Osher (that also uses the minmod function) [25, §2.4].

[7] The Courant number (ς) required for TVBM-stability is much larger than that necessary for linear stability when the order of the SSP-RK time scheme is equal or smaller than that of the discretization [25, p. 198].

[8] In scalar conservation laws, the optimal M corresponds to the supremum of the set of absolute values of the second derivative of the solution at local extrema [25, p. 195].

8.4.2. Local characteristic variables

In systems of conservation laws, generalized slope (and also moment, see §8.5) limiters are typically applied to *local* characteristic variables, as this is necessary to ensure TVB behavior [9].

A given state vector $\boldsymbol{q}_k^h$ may be projected to characteristic variables by left-multiplying it with some left eigenvector matrix (as defined in §2). Inter-element limiters act on Legendre coefficients of the solution on a compact stencil encompassing three elements, $\{\hat{\boldsymbol{Q}}_{k-1}, \hat{\boldsymbol{Q}}_k, \hat{\boldsymbol{Q}}_{k+1}\}$. It is thus convenient to employ such a local characteristic decomposition directly on the matrices of Legendre coefficients, which simply means that any eigenvectors or eigenvalues are evaluated at the mean state of the middle element in the stencil, i.e. $\{\boldsymbol{L}_k\hat{\boldsymbol{Q}}_{k-1}, \boldsymbol{L}_k\hat{\boldsymbol{Q}}_k, \boldsymbol{L}_k\hat{\boldsymbol{Q}}_{k+1}\}$ with:

$$\boldsymbol{L}_k \coloneqq \boldsymbol{L}\left(\begin{bmatrix}\hat{q}_{1\,1\,k} & \hat{q}_{2\,1\,k} & \cdots & \hat{q}_{\mathrm{I}\,1\,k}\end{bmatrix}^{\mathsf{T}}\right), \qquad \boldsymbol{R}_k \coloneqq \boldsymbol{R}\left(\begin{bmatrix}\hat{q}_{1\,1\,k} & \hat{q}_{2\,1\,k} & \cdots & \hat{q}_{\mathrm{I}\,1\,k}\end{bmatrix}^{\mathsf{T}}\right). \tag{8.19}$$

For further details and generalizations (e.g. to triangular elements), refer to [24].

8.4.3. TVB limiter

My implementation of the TVB slope limiter used in this thesis is based on [25, §2.4]. It consists on the following steps:

1. Let us assume that, after applying stage s of an S-stage Runge-Kutta time-scheme, $\widetilde{\boldsymbol{q}}_k^h(t_{n+s/\mathrm{S}}, \xi)$ and $\hat{\boldsymbol{Q}}_k\left(t_{n+s/\mathrm{S}}\right)$ have just been computed at $t_{n+s/\mathrm{S}} = t_n + (s/S)\Delta t$, for $\Omega_k \in \mathcal{T}^h \cup \mathcal{G}^h$ (ghost elements included). Evaluate the following for every (troubled) element in $\mathcal{T}^h$:

$$\Delta\boldsymbol{q}_k^L \coloneqq \hat{\boldsymbol{q}}_{1\,k}\left(t_{n+\frac{s}{\mathrm{S}}}\right) - \widetilde{\boldsymbol{q}}_k^h\left(t_{n+\frac{s}{\mathrm{S}}}, -1\right), \qquad \Delta\boldsymbol{q}_k^R \coloneqq \widetilde{\boldsymbol{q}}_k^h\left(t_{n+\frac{s}{\mathrm{S}}}, 1\right) - \hat{\boldsymbol{q}}_{1\,k}\left(t_{n+\frac{s}{\mathrm{S}}}\right), \tag{8.20a}$$

$$\Delta\bar{\boldsymbol{q}}_k^L \coloneqq \hat{\boldsymbol{q}}_{1\,k}\left(t_{n+\frac{s}{\mathrm{S}}}\right) - \hat{\boldsymbol{q}}_{1\,k-1}\left(t_{n+\frac{s}{\mathrm{S}}}\right), \qquad \Delta\bar{\boldsymbol{q}}_k^R \coloneqq \hat{\boldsymbol{q}}_{1\,k+1}\left(t_{n+\frac{s}{\mathrm{S}}}\right) - \hat{\boldsymbol{q}}_{1\,k}\left(t_{n+\frac{s}{\mathrm{S}}}\right). \tag{8.20b}$$

2. Compute corresponding left and right–sided, TVB-limited slopes (see §§ 8.4.1 and 8.4.2):

$$\left(\Delta\boldsymbol{q}_k^L\right)^* \coloneqq \boldsymbol{R}_k \bmod \operatorname*{minmod}_k\left(\boldsymbol{L}_k\Delta\boldsymbol{q}_k^L, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^L, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^R\right), \tag{8.21a}$$

$$\left(\Delta\boldsymbol{q}_k^R\right)^* \coloneqq \boldsymbol{R}_k \bmod \operatorname*{minmod}_k\left(\boldsymbol{L}_k\Delta\boldsymbol{q}_k^R, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^L, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^R\right). \tag{8.21b}$$

3. Compute safe slopes for the L^2 projection of the approximate solution on a $\mathrm{p} = 1$ space:

$$\hat{\boldsymbol{q}}_{2\,k}^* \equiv \boldsymbol{R}_k \bmod \operatorname*{minmod}_k\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{2\,k}, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^L, \boldsymbol{L}_k\Delta\bar{\boldsymbol{q}}_k^R\right). \tag{8.22}$$

4. Any component i of the approximate solution for which $\left(\Delta q_{ik}^L\right)^* \neq \Delta q_{ik}^L$ or $\left(\Delta q_{ik}^R\right)^* \neq \Delta q_{ik}^R$ has unsafe edge values. To correct this, replace it with its limited linear version by setting:

$$\hat{\boldsymbol{q}}_{i\,2\,k} \leftarrow \hat{\boldsymbol{q}}_{i\,2\,k}^*, \qquad \hat{\boldsymbol{q}}_{ijk} \leftarrow 0 \quad \text{for } j > 2. \tag{8.23}$$

Despite its robustness, simplicity and effectiveness (for a proper choice of M), this method has two important shortcomings:

- It relies on a problem-dependent and generally unknown parameter to function effectively.
- When a solution component is limited, all information apart from its mean value is discarded.

8.5. Generalized moment limiting

Biswas et al. [13] pioneered an approach aimed at overcoming both weaknesses of the TVB slope limiter (§8.4.3) at once. Their idea was to extend the slope limiting idea to higher-order *moments* of the approximate solution. This was a breakthrough achievement, making it possible for high-order accuracy to be retained—not only at smooth extrema, but also across linearly degenerate discontinuities and

[9] This is *not* enough, however, to guarantee that certain physical bounds will always be preserved; see §8.8.

all the way up to the very close neighborhood of genuine nonlinear shocks—without any problem-dependent constant to adjust. Over time, this method has developed into its own family of DG limiters [17, 69, 124].

In the context of Legendre-based DG (see §4.2.1), the mth moment of a polynomial approximant $\widetilde{q}^h_{ik}(t,\xi)$ is:

$$\int_{-1}^{1} \widetilde{q}^h_{ik} \mathcal{P}_m \, \mathrm{d}\xi \equiv \frac{2}{2m+1} \hat{q}_{i\,m+1\,k} \,. \tag{8.24}$$

If applied to a linear approximation $\widetilde{q}^h_{ik}(\xi)$ (so that its two non-zero moments are mean value and slope), generalized moment limiting reduces to generalized slope limiting. It can be shown [69] that the mth moment of a C^∞ function defined over Ω_k—and therefore, in turn, the $\hat{q}_{i\,m+1\,k}$ Legendre coefficient of its polynomial approximation—is proportional to its mth derivative sampled at any point of the element (i.e. the zeroth coefficient is the mean value, the first one is associated with the slope, the second with the curvature, and so on).

8.5.1. BDF limiter

This is the original generalized moment limiter due to Biswas, Devine, and Flaherty [13] (hence its designation), the first of the three such limiters studied in this report. My implementation of it is as follows; starting with $i = 1$:

1. Let $\mathcal{T}^* \subseteq \mathcal{T}^h$ be the set of troubled $\Omega_k \in \mathcal{T}^h$ (§8.3); if no sensor is used, $\mathcal{T}^* = \mathcal{T}^h$.
2. For every j from $\mathrm{J}-1$ to 1:
   (a) For every $\Omega_k \in \mathcal{T}^*$, use (8.17) to compute a limited version of the jth Legendre coefficient (in characteristic variables) associated with the ith state vector component of the approximate solution:
   $$\left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{j+1\,k}\right)^*_i \coloneqq \mathrm{minmod}\left(\left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{j+1\,k}\right)_i, \frac{\left(\boldsymbol{L}_k \left(\hat{\boldsymbol{q}}_{j\,k} - \hat{\boldsymbol{q}}_{j\,k-1}\right)\right)_i}{2j-1}, \frac{\left(\boldsymbol{L}_k \left(\hat{\boldsymbol{q}}_{j\,k+1} - \hat{\boldsymbol{q}}_{j\,k}\right)\right)_i}{2j-1} \right) . \tag{8.25}$$
   This generalizes (8.22) to modes $j \geq 1$.
   (b) Exclude from $\mathcal{T}^*$ any Ω_k for which $\left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{j+1\,k}\right)^*_i = \left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{j+1\,k}\right)_i$ or $\left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{j+1\,k}\right)_i = 0$, and assume that their remaining modes are safe—i.e.: $\left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{r\,k}\right)^*_i \coloneqq \left(\boldsymbol{L}_k \hat{\boldsymbol{q}}_{r\,k}\right)_i$ for $r = 2, 3, \ldots, j$. The limiting process is thus stopped for the ith characteristic component of all elements which (are assumed to) no longer require it.

   This step gives rise to the notion of *hierarchical limiting*, in which higher-order modes are limited using lower-order ones until no more limiting is required. Looping over j from highest to lowest ensures that only unlimited modes of each element's left and right neighbors are used in each iteration.
3. Advance to the next PDE component, $i \leftarrow i+1$. If $i \leq \mathrm{I}$, go back to step 1 (i.e. repeat the process for the next characteristic component of *all* troubled elements); otherwise, move on to the last step.
4. For every (troubled) Ω_k, replace its unlimited expansion coefficients with limited ones (only unsafe ones will actually be modified[10]):
   $$\hat{\boldsymbol{Q}}_k \leftarrow \boldsymbol{R}_k \left(\boldsymbol{L}_k \hat{\boldsymbol{Q}}_k\right)^* . \tag{8.26}$$

8.5.2. BSB limiter

Burbeau, Sagaut, and Bruneau [17] proposed this limiter as a general improvement over BDF in terms of overall accuracy. In essence, it consists on performing the following after step 2b in §8.5.1:

(c) Compute the following quantities:

$$\Delta\hat{\boldsymbol{q}}^L_{jk} \coloneqq \hat{\boldsymbol{q}}_{j\,k-1} + (2j-1)\,\hat{\boldsymbol{q}}_{j+1\,k-1} \,, \qquad \Delta\hat{\boldsymbol{q}}^R_{jk} \coloneqq \hat{\boldsymbol{q}}_{j\,k+1} - (2j-1)\,\hat{\boldsymbol{q}}_{j+1\,k+1} \,. \tag{8.27}$$

[10] In the DGIGA case, special care needs to be taken to make sure that, for any conservative variable that is not affected by the limiter, the original B-spline expansion coefficients are maintained (instead of needlessly converting them back and forth).

(d) Using the previous, evaluate:

$$\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{\max} \coloneqq \operatorname{minmod}\left(\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i, \frac{\left(\boldsymbol{L}_k\left(\hat{\boldsymbol{q}}_{jk}-\Delta\hat{\boldsymbol{q}}_{jk}^{L}\right)\right)_i}{2j-1}, \frac{\left(\boldsymbol{L}_k\left(\Delta\hat{\boldsymbol{q}}_{jk}^{R}-\hat{\boldsymbol{q}}_{jk}\right)\right)_i}{2j-1}\right). \tag{8.28}$$

(e) And, in turn, compute:

$$\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{**} \coloneqq \operatorname{maxmod}\left(\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{*}, \left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{\max}\right), \tag{8.29}$$

where:

$$\operatorname{maxmod}(q_1,q_2) \coloneqq \begin{cases} \operatorname{sign}(q_1)\max\{|q_1|,|q_2|\} & \text{if } \operatorname{sign}(q_1)=\operatorname{sign}(q_2) \\ 0 & \text{otherwise} \end{cases}, \tag{8.30}$$

which is applied in vectorized fashion, in the same manner as (8.17) earlier.

(f) Repeat step 2b in 8.5.1, this time searching for elements such that $\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{**} = \left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i$.

In step 4, $\left(\boldsymbol{L}_k\hat{\boldsymbol{Q}}_k\right)^{**}$ is used instead of the more conservative $\left(\boldsymbol{L}_k\hat{\boldsymbol{Q}}_k\right)^{*}$. Numerical experiments in [17] suggest that this modification does indeed lower numerical diffusion, while still maintaining nonlinear stability in practice.

8.5.3. Krivodonova's limiter

Yet another variant of the generalized moment limiter for DG was proposed by Krivodonova [69]. She claims that her recipe leads to reduced numerical diffusion in the limited solution without compromising stability (in most practical situations, at least). While this is similar to BSB, her approach is substantially simpler.

The reasoning behind this limiter is as follows. When generalizing (8.22) from slope to any arbitrary higher order moment, consider limiting solution derivatives directly (instead of inner products between solution and each Legendre basis function). Assuming that $q_{ik}^h(t,x) \in C^\infty$, comparison between its Taylor series and Legendre polynomial expansion reveals that [69]:

$$\hat{q}_{ijk}(t) \approx C\Delta x_k^j \frac{\partial^j q_{ik}^h}{\partial x^j}(t,\zeta), \quad \zeta \in \Omega_k\,. \tag{8.31}$$

i.e. each Legendre coefficient is an estimate (up to a scaling factor, $C \in \mathbb{R}^+$) of the spatial derivative of matching order of the approximate solution, sampled at any point of the element. This eventually leads to [69]:

$$\hat{q}_{i\,j+1\,k} = \frac{\hat{q}_{i\,j\,k}-\hat{q}_{i\,j\,k-1}}{2\,(2j-1)} + \mathcal{O}\left(\Delta x^{j+2}\right) = \frac{\hat{q}_{i\,j\,k+1}-\hat{q}_{i\,j\,k}}{2\,(2j-1)} + \mathcal{O}\left(\Delta x^{j+2}\right), \quad j=1,2,3,\ldots,\mathrm{J}\,, \tag{8.32}$$

which is smaller than the approximation employed by Biswas et al. [13] in the BDF limiter (§8.5.1) by a factor of $1/2$ in the leading term of every mode. As an entirely ad hoc modification, Krivodonova [69] then proposes to replace (8.25), in step 2a of §8.5.1, with the following:

$$\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i^{*} \coloneqq \operatorname{minmod}\left(\left(\boldsymbol{L}_k\hat{\boldsymbol{q}}_{j+1\,k}\right)_i, \frac{\left(\boldsymbol{L}_k\left(\hat{\boldsymbol{q}}_{j\,k}-\hat{\boldsymbol{q}}_{j\,k-1}\right)\right)_i}{\alpha_j}, \frac{\left(\boldsymbol{L}_k\left(\hat{\boldsymbol{q}}_{j\,k+1}-\hat{\boldsymbol{q}}_{j\,k}\right)\right)_i}{\alpha_j}\right). \tag{8.33}$$

For $\alpha_j = 2j-1$, Krivodonova's limiter becomes identical to BDF. However, if $\alpha_j = 2\,(2j-1)$ (i.e. strictly as derived), it turns out to be too conservative for practical purposes (it becomes unnecessarily diffusive). This suggests that, actually, BDF's overestimation of the modal expansion coefficients is itself an ad hoc compromise between stability and accuracy, made implicitly through the design decision of limiting Legendre moments (rather than solution derivatives). Krivodonova's contribution is the realization that this can be taken one step further by using an *even more aggressive* set of α_j values in (8.33). More specifically, any values in the range:

$$1 \leq \alpha_j \leq 2\,(2j-1)\,, \tag{8.34}$$

are claimed to maintain stability[11] in all numerical tests performed in [69]. Consequently, the simplest case $\alpha_j = 1$—corresponding to the mildest possible limiting—seems to be optimal. Only this value is employed in the present report.

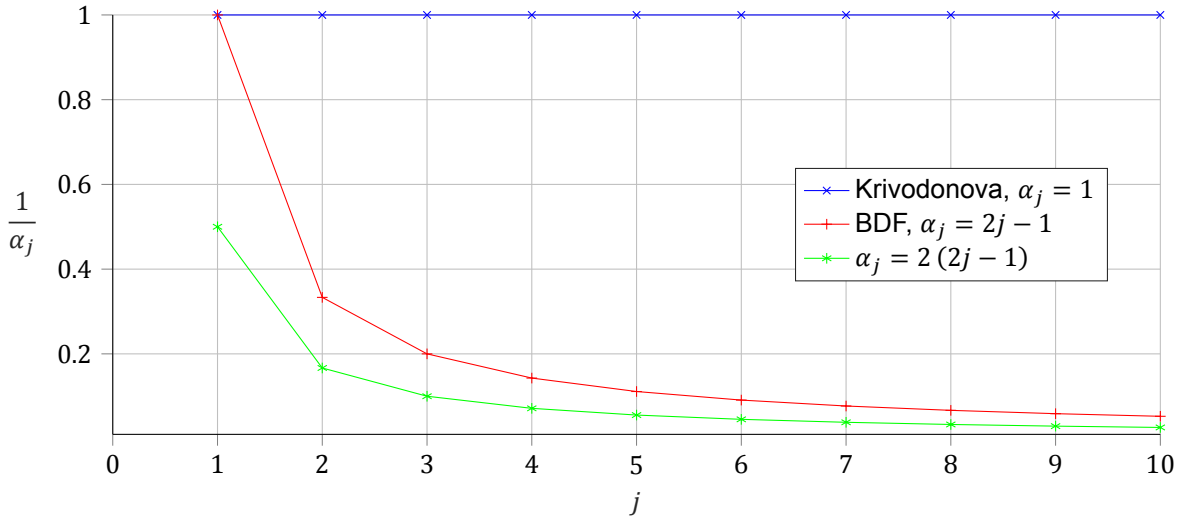


Figure 8.2: Krivodonova's range of $\frac{1}{\alpha_j}$ scaling factors as functions of j, in comparison with BDF's ones. The reduction in limiting intensity of $\alpha_j = 1$, in relation to (8.32), becomes more pronounced at high modes, possibly leading to some amount of small-wavelength oscillations not being removed by the limiter.

8.6. Weighted essentially non-oscillatory (WENO) limiting

A more recent trend exists in the literature trying to bring the ENO and/or WENO machinery, well-established in the contexts of finite differences and finite volumes, to discontinuous Galerkin. For details on this kind of high resolution schemes, see [43, 62, 110] and references therein.

This "(W)ENO for DG" initiative seems to have started with the series of papers [101, 103, 136], in which the so-called *Hermite WENO* (HWENO) approach is developed. HWENO schemes have a more compact stencil than traditional variants for a given order, making them better suited to DG. Other examples of HWENO limiters for DG are [84, 87, 134]. The main downsides of WENO limiters are their relative complexity and associated computational cost. The latter, however, can be mitigated with the use of sensors (§8.3).

8.6.1. HWENO limiter

In the present thesis, I consider a particularly simple HWENO limiter originally due to Zhong and Shu [134], later extended to FR/CPR in [29] and improved in [137]. I apply it in the following way, which incorporates the improvements of Zhu et al. [137] by exploiting the fact that, when undergoing limiting, all solution components exist in a Legendre-based space (see §8.2).

Consider, once again, the set of troubled elements $\mathcal{T}^* \subseteq \mathcal{T}^h$. Now, for every $\Omega_k \in \mathcal{T}^*$:

1. The local ENO/WENO stencil centered on Ω_k contains three polynomial vectors, namely: $\boldsymbol{q}^h_{k-1}$, $\boldsymbol{q}^h_k$ and $\boldsymbol{q}^h_{k+1}$. For ease of notation, let us define their respective projections to local characteristic variables as:

$$\boldsymbol{q}_0 \coloneqq \boldsymbol{L}_k \widetilde{\boldsymbol{q}}^h_{k-1}\,, \qquad \boldsymbol{q}_1 \coloneqq \boldsymbol{L}_k \widetilde{\boldsymbol{q}}^h_{k}\,, \qquad \boldsymbol{q}_2 \coloneqq \boldsymbol{L}_k \widetilde{\boldsymbol{q}}^h_{k+1}\,. \tag{8.35}$$

   Replace the mean values (i.e. 1st Legendre coefficient) of the left and right approximate solution polynomials in characteristic variables by those of the approximate solution polynomial, in

[11] As with the rest of moment limiters, there is no known TVB guarantee of nonlinear stability linked with this range. Krivodonova [69] herself admits that the choice $\alpha_j = 1$ allows (minor) oscillations to appear in highest order derivatives. This is similar to the essentially non-oscillatory (ENO) requirement, under which some degree of non-monotonicity is considered acceptable.

characteristic variables, of Ω_k:

$$\hat{\boldsymbol{q}}_{1\,0} \leftarrow \hat{\boldsymbol{q}}_{1\,1}\,, \qquad\qquad \hat{\boldsymbol{q}}_{1\,2} \leftarrow \hat{\boldsymbol{q}}_{1\,1}\,. \tag{8.36}$$

This ensures that any *convex combination*[12] of these three vectors' components will preserve the unlimited element-wise solution component averages.

2. Construct a limited version of $q_{i\,1}$,

$$\left(\boldsymbol{L}_k \widetilde{\boldsymbol{q}}_k^h\right)_i^* \coloneqq \omega_0 q_{i\,0} + \omega_1 q_{i\,1} + \omega_2 q_{i\,2}\,, \tag{8.37}$$

using *normalized nonlinear weights*, $\{\omega_l\}_{l=0}^2$, obtained as follows. For $i = 1, 2, \ldots, \mathrm{I}$:

(a) Estimate the smoothness of q_{il}, for $l = 0, 1, 2$. Since it is a Legendre-based polynomial of degree p, the *smoothness indicator* suggested in [134]—which, in turn, was first proposed in [62]—becomes:

$$\beta_l \coloneqq \sum_{\kappa=1}^{\mathrm{p}} 2^{2\kappa-1} \int_{-1}^{1} \left(\sum_{j=1}^{\mathrm{J}} \hat{q}_{ijl} \frac{\mathrm{d}^\kappa \mathcal{P}_{j-1}}{\mathrm{d}\xi^\kappa} \right)^2 \mathrm{d}\xi\,. \tag{8.38}$$

The integrand in (8.38) is a product of two polynomials, each of degree $\leq \mathrm{p} - 1$; it is hence also a polynomial, but of degree $\leq 2\mathrm{p} - 2$. Its integral can therefore be computed exactly using e.g. $\geq \mathrm{p}$–point Gauss-Legendre quadrature (see table 4.2).

(b) Use the previous to compute corresponding *non-normalized nonlinear weights*. For $l = 0, 1, 2$:

$$\bar{\omega}_l \coloneqq \frac{\gamma_l}{(\varepsilon + \beta_l)^r}\,, \tag{8.39}$$

where γ_l is an element of the triplet of *linear weights* subject to the condition $\sum_{l=0}^2 \gamma_l = 1$. In order to bias the stencil toward the middle element, it is typical to choose $\gamma_1 \gg \gamma_0$, $\gamma_1 \gg \gamma_2$—in fact, the larger the ratios $\frac{\gamma_1}{\gamma_0}$ and $\frac{\gamma_1}{\gamma_2}$ are, the more accurate the limited solution polynomial is in smooth regions but, also, the more oscillatory it remains near discontinuities. Imitating [134], I employ $\varepsilon = 10^{-6}$, $r = 2$ and the following linear weights:

$$\gamma_0 = 0.001\,, \qquad \gamma_1 = 0.998\,, \qquad \gamma_2 = 0.001\,. \tag{8.40}$$

(c) Normalize the nonlinear weights obtained in the previous step; for $l = 0, 1, 2$:

$$\omega_l \coloneqq \frac{\bar{\omega}_l}{\sum_{l=0}^2 \bar{\omega}_l}\,. \tag{8.41}$$

Once the previous steps have been completed, i.e. $\exists \left(\boldsymbol{L}_k \widetilde{\boldsymbol{q}}_k^h\right)^* \forall k : \Omega_k \in \mathcal{T}^*$, the unlimited approximate solution in conservative variables is overridden by its limited counterpart by setting, for every $\Omega_k \in \mathcal{T}^*$:

$$\widetilde{\boldsymbol{q}}_k^h \leftarrow \boldsymbol{R}_k \left(\boldsymbol{L}_k \widetilde{\boldsymbol{q}}_k^h\right)^*\,. \tag{8.42}$$

8.7. Flux corrected transport (FCT) limiting

As explained in §6.4, it may be advantageous for DGIGA to apply limiting in a predictor-correction approach, where AFC is used to generate a suitable predictor. In what follows, I propose a modified version of the FCT limiter in [77] that corrects said predicted solution, recovering a compromise between high-order and high-resolution. This method assumes the discretization to be of the type (6.30); it is not applicable to the other methods, (6.16) included. Also, unlike the rest of limiters in this chapter, it is used only once per time-step (and not after every time-stage)[13].

[12] A convex combination is a sum of weighted terms in which the sum of said weights is exactly 1. In the HWENO limiter case, this means that $\sum_{l=0}^2 \omega_{il} = 1$ for every i.

[13] Only the predictor is advanced stage by stage, and is corrected, only once, at the end of the step.

8.7.1. Linearized antidiffusive fluxes

Having constructed (6.30)—a low-order predictor—via AFC, the theory of flux corrected transport (FCT) ensures that, for every control point r, there must exist a vector of *high-order net antidiffusive fluxes*, $\boldsymbol{f}^H_{rk}$, that relates the degrees of freedom of the low-order predictor with those of its unlimited high-order (possibly oscillatory) counterpart, such that:

$$\left(\hat{\boldsymbol{Q}}^H_k - \hat{\boldsymbol{Q}}^L_k\right)\boldsymbol{\mathcal{M}}^L_k = \boldsymbol{F}^H_k\,, \qquad \boldsymbol{F}^H_k \coloneqq \begin{bmatrix}\boldsymbol{f}^H_{1k} & \boldsymbol{f}^H_{2k} & \cdots & \boldsymbol{f}^H_{\mathrm{J}k}\end{bmatrix}. \tag{8.43}$$

In an analogous manner to (6.33), each net flux vector is associated to a control point and includes contributions from all other ones within nonzero basis function support, each being referred to as a *raw* antidiffusive flux vector, $\boldsymbol{f}_{rjk}$. These are defined such that:

$$\boldsymbol{f}^H_{rk} \coloneqq \sum_{j\neq r}\boldsymbol{f}_{rjk}\,, \qquad \boldsymbol{f}_{rjk} \equiv \boldsymbol{f}_{jrk}\,, \tag{8.44}$$

as in (6.33), the summation can be made over all control points, but the only nonzero contributions will be due to those N_j with nonzero support overlapping that of N_r (itself excluded).

Explicit time-marching solvers such as those considered in this report (§7) are best suited to a *linearized* variant of FCT. Each raw antidiffusive flux is approximated as [90]:

$$\boldsymbol{f}_{rjk} \approx \Delta t\left(\left(\hat{\boldsymbol{r}}^L_{rk} - \hat{\boldsymbol{r}}jk^L\right)m_{jrk} + \boldsymbol{D}_{rjk}\left(\hat{\boldsymbol{q}}^L_{rk} - \hat{\boldsymbol{q}}^L_{jk}\right)\right), \tag{8.45}$$

in which every term on the right-hand-side is evaluated using the low-order approximate solution at $t + \Delta t$—including, in particular, the residual vectors. This implies one additional evaluation of the predictor residuals in (6.30), done *after* the time-scheme has finished advancing the discretization to the next time-step (if the limiter is being applied in an initialization step, see 8.7.3).

8.7.2. Synchronized sequential FCT limiter

In the context of FCT, limiting consists on reducing the magnitude of each antidiffusive flux by a factor $0 \leq \alpha_{rjk} \leq 1$, such that replacing them in (8.44) and propagating the result via (8.43), leads to a set of matrices of *limited* high-order control values which encode the sought-after high-resolution approximate solution:

$$\hat{\boldsymbol{Q}}_k \coloneqq \hat{\boldsymbol{Q}}^L_k + \boldsymbol{F}^L_k\left(\boldsymbol{\mathcal{M}}^L_k\right)^{-1}, \qquad \boldsymbol{f}^L_{rk} \coloneqq \sum_{j\neq r}\alpha_{rjk}\boldsymbol{f}_{rjk}\,, \qquad \alpha_{rjk} \equiv \alpha_{jrk}\,. \tag{8.46}$$

These limiting coefficients can be obtained in a generalized version of Zalesak's algorithm [77], [78, §9.2]. Said limiter was designed for low-order Lagrange basis functions, but can be applied in IGA by essentially treating control points as as if they were nodes [90]. The limiting procedure used for DGIGA-AFC in this report consists on the following steps:

1. Prelimit raw antidiffusive fluxes as in [75, §6.4.1] (in conservative variables), i.e. set $f_{irjk} = 0$ for every i, r, j, k for which $f_{irjk}\left(\hat{q}^L_{ijk} - \hat{q}^L_{irk}\right) > 0$ (if this product is positive, the associated raw antidiffusive flux is actually diffusive).

2. Determine upper and lower bounds for each control value in the high-resolution approximation. For DGIGA-AFC, I propose to do so as follows:

   (a) First, determine element-local extrema (ghost elements included) in the control polygon of each *primary variable's low-order predictor*:

   $$\hat{v}^{\max}_{irk} \coloneqq \max\left\{\hat{v}^L_{i\,r-\mathrm{p}\,k}, \ldots, \hat{v}^L_{i\,r+\mathrm{p}\,k}\right\}, \qquad \hat{v}^{\min}_{irk} \coloneqq \min\left\{\hat{v}^L_{i\,r-\mathrm{p}\,k}, \ldots, \hat{v}^L_{i\,r+\mathrm{p}\,k}\right\}. \tag{8.47}$$

   Control values in primary variables are obtained from conservative ones via left-multiplication with $\boldsymbol{T}(\hat{\boldsymbol{q}}^L_{rk})$, defined by (2.44) for Euler (or the identity matrix otherwise).

   (b) Then, communicate extrema across element edges (for $k = 0, \ldots \mathrm{K}$), i.e. :

   $$\hat{v}^{\max}_{i\mathrm{J}k} = \hat{v}^{\max}_{i\,1\,k+1} = \max\left\{\hat{v}^{\max}_{i\,\mathrm{J}\,k}, \hat{v}^{\max}_{i\,1\,k+1}\right\}, \quad \hat{v}^{\min}_{i\mathrm{J}k} = \hat{v}^{\min}_{i\,1\,k+1} = \min\left\{\hat{v}^{\min}_{i\,\mathrm{J}\,k}, \hat{v}^{\min}_{i\,1\,k+1}\right\}. \tag{8.48}$$

(c) Finally, propagate them inwards of each $\Omega_k \in \mathcal{T}^h$; starting e.g. with $r = 2, 3, \dots, \mathrm{p}+1$ (basis functions that have shared nonzero support with the left-most one):

$$\hat{v}^{\max}_{irk} = \max\left\{\hat{v}^{\max}_{i1k}, \hat{v}^{\max}_{irk}\right\}, \qquad \hat{v}^{\min}_{irk} = \min\left\{\hat{v}^{\min}_{i1k}, \hat{v}^{\min}_{irk}\right\}. \tag{8.49}$$

and for $r = \mathrm{J}-1, \mathrm{J}-2, \dots, \mathrm{J}-\mathrm{p}$ (analogously, with the right-most one):

$$\hat{v}^{\max}_{irk} = \max\left\{\hat{v}^{\max}_{i\mathrm{J}k}, \hat{v}^{\max}_{irk}\right\}, \qquad \hat{v}^{\min}_{irk} = \min\left\{\hat{v}^{\min}_{i\mathrm{J}k}, \hat{v}^{\min}_{irk}\right\}. \tag{8.50}$$

This recipe reduces to sharing local maxima and minima among all control points within shared nonzero basis function support of each other, treating the two nonzero basis functions at every patch interface as if they where one. Preliminary numerical results (figure 8.3) suggest that this approach is at least not worse than no coupling at all.

3. Compute synchronized limiting coefficients for each raw antidiffusive flux. This requires casting the raw antidiffusive fluxes to primitive variables, which for the Euler equations (2.39) can be done via control-point-based transformations as:

$$f^{\rho}_{rjk} \coloneqq f_{1rjk}, \tag{8.51a}$$

$$f^{u}_{rjk} \coloneqq \frac{f_{2rjk} - \hat{u}_{rk} f^{\rho}_{rjk}}{\hat{\rho}_{rk}}, \tag{8.51b}$$

$$f^{p}_{rjk} \coloneqq (\gamma - 1)\left(f_{3rjk} + \frac{1}{2}\hat{u}^2_{rk} f^{\rho}_{rjk} - \hat{u}_{rk} f_{2rjk}\right). \tag{8.51c}$$

For the other conservation laws, no such conversion is necessary: there is no distinction between primary and conservative variables.

In synchronized FCT, a single limiting coefficient acts on all components of $\boldsymbol{f}_{rjk}$. Yet, it is possible to apply multiple such coefficients sequentially, each corresponding to one loop over the following algorithm; starting with $i = 1$ and assuming, without loss of generality, that all primary variables are to be limited:

(a) Use (8.51) to compute f^{v}_{rjk}, the raw antidiffusive flux corresponding to v_{irjk}.

(b) Compute positive/negative contributions to the net antidiffusive flux at control point r:

$$f^{+}_{rk} \coloneqq \sum_{j \neq r} \max\left\{0, f^{v}_{rjk}\right\}, \qquad f^{-}_{rk} \coloneqq \sum_{j \neq r} \min\left\{0, f^{v}_{rjk}\right\}. \tag{8.52}$$

(c) Compute distances to the local maxima/minima determined earlier:

$$\Delta\hat{v}^{+}_{rk} \coloneqq \hat{v}^{\max}_{irk} - \hat{v}^{L}_{irk}, \qquad \Delta\hat{v}^{-}_{rk} \coloneqq \hat{v}^{\min}_{irk} - \hat{v}^{L}_{irk}. \tag{8.53}$$

(d) Compute positive/negative control value correction factors:

$$\alpha^{+}_{rk} \coloneqq \min\left\{1, m^{L}_{r}\frac{\Delta\hat{v}^{+}_{rk}}{f^{+}_{rk}}\right\}, \qquad \alpha^{-}_{rk} \coloneqq \min\left\{1, m^{L}_{r}\frac{\Delta\hat{v}^{-}_{rk}}{f^{-}_{rk}}\right\}. \tag{8.54}$$

(e) Use the previous to compute $\alpha^{v}_{rjk} \equiv \alpha^{v}_{jrk}$, as:

$$\alpha^{v}_{rjk} \coloneqq \min\left\{\alpha^{*}_{rjk}, \alpha^{*}_{jrk}\right\}, \qquad \alpha^{*}_{rjk} \coloneqq \begin{cases} \alpha^{+}_{rk} & \text{if } f^{v}_{rjk} \geq 0 \\ \alpha^{-}_{rk} & \text{if } f^{v}_{rjk} < 0 \end{cases}. \tag{8.55}$$

(f) Apply α^{v}_{rjk} to the antidiffusive flux vector in conservative variables:

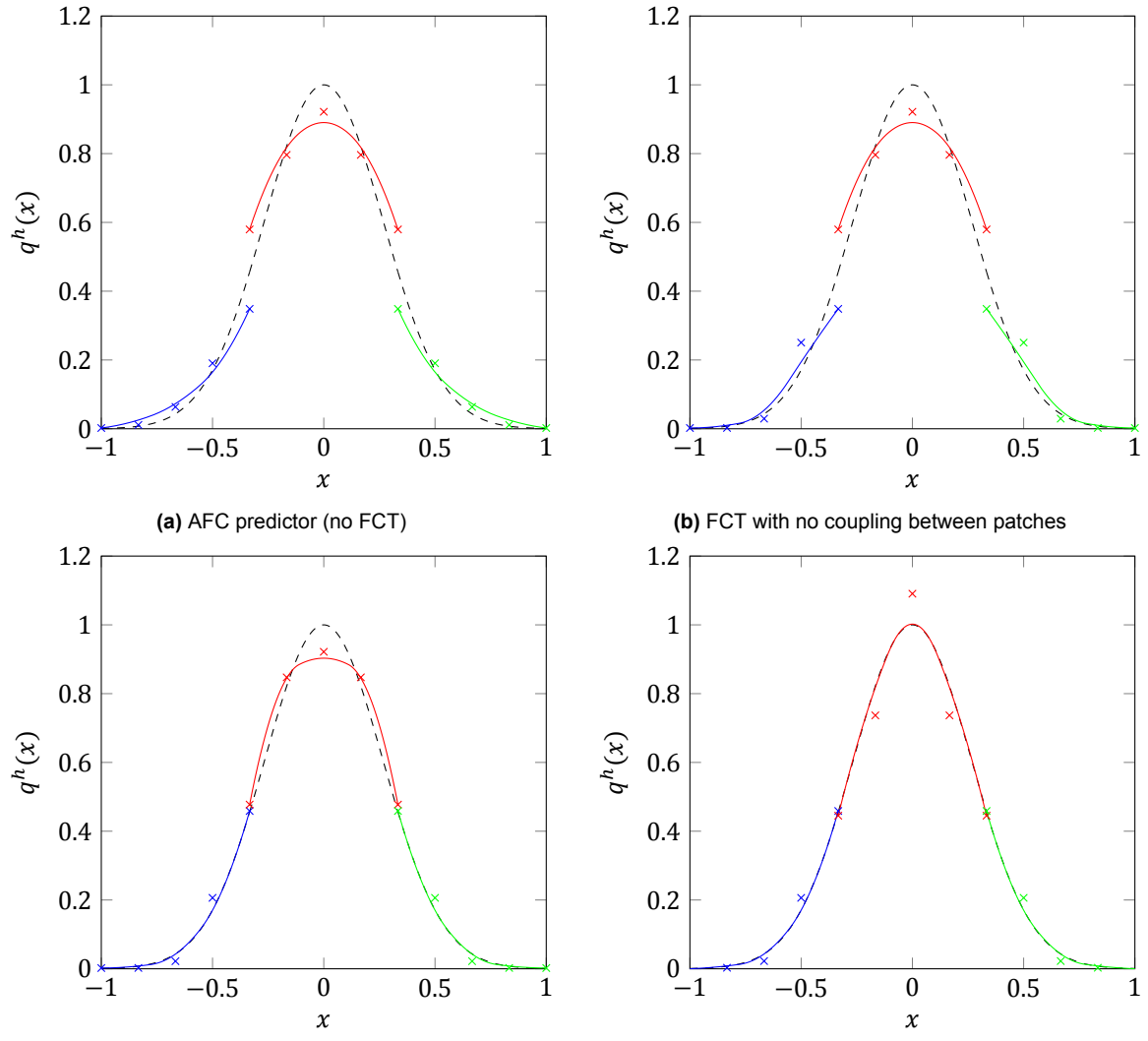
$$\boldsymbol{f}^{L}_{rjk} \coloneqq \alpha^{v}_{rjk}\boldsymbol{f}_{rjk}. \tag{8.56}$$

(g) If $i < \mathrm{I}$, set $i \leftarrow i + 1$ and go back to step (a) but replace $\boldsymbol{f}_{rjk}$ with $\boldsymbol{f}^{L}_{rjk}$, i.e. use the *partially* limited $\boldsymbol{f}^{L}_{rjk}$ to compute the next f^{v}_{rjk} and, from it, limit $\boldsymbol{f}^{L}_{rjk}$ further. Otherwise, stop: $\boldsymbol{f}^{L}_{rjk}$ is fully limited.

Limited net antidiffusive fluxes can finally be obtained using (8.46) by setting, e.g. for the Euler equations, $\alpha_{rjk} = \alpha^p_{rjk}\alpha^u_{rjk}\alpha^\rho_{rjk}$. Or, more conveniently, simply by replacing $\boldsymbol{f}_{rjk}$ with $\boldsymbol{f}^L_{rjk}$ in (8.44), *after* the limiter has been fully applied [77]:

$$\boldsymbol{f}^L_{rk} = \sum_{j \neq r} \boldsymbol{f}^L_{rjk} \,. \tag{8.57}$$

With the Euler equations, it is critical to ensure that neither density nor pressure attain negative values, making these two a sensible choice of primary variables to limit. It should also be pointed out that, in this sequential approach, the order in which the selected primary variables are limited will generally influence the final value of α_{rjk}. Unless specified otherwise, this limiter is applied in this report to all PDE components, in their natural order (i.e. as defined in §2).



(a) AFC predictor (no FCT)

(b) FCT with no coupling between patches

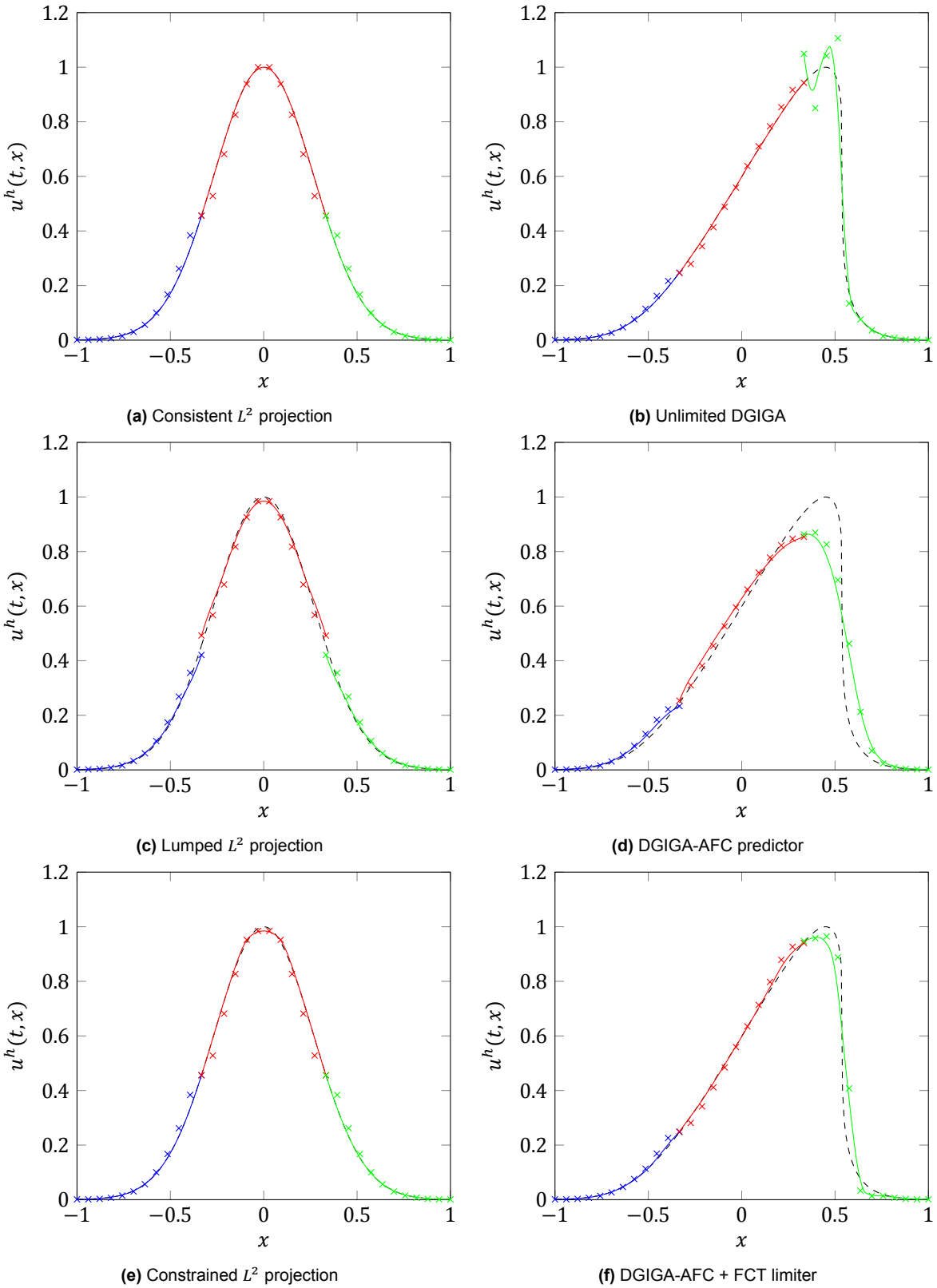
(c) FCT with proposed inter-patch coupling

(d) Consistent L^2 projection

Figure 8.3: Comparison between four ways of projecting a Gaussian hump, $q(x) = \exp\left(-\frac{9\pi x^2}{4}\right)$, into a DGIGA discretization with 15 degrees of freedom, $\mathrm{K} = 3$, $\mathrm{p} = 2$ and $\mathrm{J} = 5$ (3 breakpoint spans per patch). Markers are control points of $q^h(x)$, and the dashed line is the exact $q(x)$.

8.7.3. Constrained initialization

The FCT limiter presented above depends on the existence of some set of antidiffusive fluxes so that the relation (8.43) can be established. Yet, (8.45) has only been defined so far for a time-matching context.



(a) Consistent L^2 projection

(b) Unlimited DGIGA

(c) Lumped L^2 projection

(d) DGIGA-AFC predictor

(e) Constrained L^2 projection

(f) DGIGA-AFC + FCT limiter

Figure 8.4: Unlimited, predictor and FCT-corrected initial condition projection (left) and approximate solution at $t = 0.45$ (right) of Burgers equation in $\Omega = [-1, 1]$, subject to periodic boundary conditions. Spatial discretization consists of 3 elements (blue, red and green), each with 11 uniformly distributed breakpoints, $\mathrm{p} = 2$ and no repeated interior knots ($\mathrm{J} = 12$), resulting in a total of 36 degrees of freedom. Time-scheme is SSP-RK3 with $\varsigma = 10^{-3}$. Cross markers correspond to control points of the approximate solution; dashed line is the exact solution.

All conservation laws in this report are time-dependent, and only unsteady solutions are considered. Even so, there is a special case in which the approximate solution might need to be limited in a steady-state fashion: when it is obtained by projecting a given initial condition (see §3.5).

The *consistent* L^2 projection of an initial condition $\boldsymbol{q}^0(x)$ onto Ω_k, an arbitrary DGIGA element, implies:

$$\left(\int_{\Omega_k} \boldsymbol{q}^0 \boldsymbol{N}^\intercal \, \mathrm{d}x\right) = \hat{\boldsymbol{Q}}_k^H \boldsymbol{\mathcal{M}}_k \,. \tag{8.58}$$

Doing the same on a DGIGA-AFC element, gives rise to a *lumped* L^2 projection:

$$\left(\int_{\Omega_k} \boldsymbol{q}^0 \boldsymbol{N}^\intercal \, \mathrm{d}x\right) = \hat{\boldsymbol{Q}}_k^L \boldsymbol{\mathcal{M}}_k^L \,. \tag{8.59}$$

It follows that:

$$\hat{\boldsymbol{Q}}_k^H \boldsymbol{\mathcal{M}}_k = \hat{\boldsymbol{Q}}_k^L \boldsymbol{\mathcal{M}}_k^L \,. \tag{8.60}$$

Meaning that the high-order solution can be constructed algebraically from the low-order one.

This fact can be exploited to determine the antidiffusive fluxes needed for FCT—even avoiding, in this case, any linearization. In practice, the limiting procedure described above can still be applied, the only difference being in the definition of raw antidiffusive fluxes (8.45), which is replaced by [77, §8]:

$$\boldsymbol{f}_{rjk} \coloneqq m_{jr} \left(\hat{\boldsymbol{q}}_{rk}^H - \hat{\boldsymbol{q}}_{jk}^H\right) . \tag{8.61}$$

8.7.4. Troubled element detection

By construction, no discontinuity sensor should activate if applied to a DGIGA-AFC element before FCT limiting (since it contains a TVD predictor solution, at that point). The straight-forward way to combine a sensor with this limiter is to first approximate the unlimited high-order solution using linearized antidiffusive fluxes, and determine the presence of discontinuities in it. The limiter is then applied as originally, but only to those $\boldsymbol{f}_{rjk}^\nu$ for which Ω_k has been determined to be troubled.

8.8. Fail-safe limiting for the Euler equations

The Euler system of equations has some additional constraints on the values of primitive variables: neither density nor pressure can ever be negative. This makes perfect sense from a physical point of view on its own, but it is also important from a numerical perspective, as it ensures that the speed of sound remains real-valued. Should that not be the case, even if only at one degree of freedom and by a small amount, any nonzero imaginary part would propagate and eventually render the entire numerical solution meaningless—assuming that the implementation supports complex numbers in the first place.

All limiters considered in this study except FCT are meant to be applied in local characteristic variables (when these are defined, as in the Euler case). It is possible that, even when local characteristic variables are TVDM, density and/or pressure experience small fluctuations. Nothing prevents these from reaching negative values[14]. I propose to do so, in line with e.g. [69], by applying two additional limiting steps (which can be seen and used as separate limiters), according to the following sequence:

1. Apply a RK stage
2. Apply the main limiter
3. Apply the first fail-safe limiter (§8.8.1; optional)
4. Apply the second fail-safe limiter (§8.8.2)
5. Repeat

The FCT limiter of §8.7.2 acts on primary variables, so it should not experience this issue. Nevertheless, the correction procedure employs a linealization; this is believed to be the reason why invalid density and pressure values may still arise in the constrained solution [77]. In situations with very

[14] I should mention, however, that there are limiters specifically designed to preserve positivity in this sense; see figure 8.1.

strong shocks, I have observed that control values in B-spline–based DG discretizations can reach invalid density and pressure values even when the actual primitive variable function they are associated with remains positive (e.g. figure 8.5a).

This might, at first, seem to be a *feature* of the B-spline basis: there is no reason (a priori) why control values themselves should be non-negative—what matters is the actual solution. Negative control values "pull" the latter towards zero, actually making the discontinuity sharper; should the solution become invalid, fail-safes in §§8.8.1 and/or 8.8.2 would take care of the problem (assuming DGIGA proper, i.e. multiple IGA patches). Nevertheless, most diffusion functions used in the AFC procedure, including both (6.34) and (6.36), happen to fail in the presence of "invalid" control values (in the previous sense). This is because they perform a decomposition into local characteristic variables at the control point level, and this fails if pressure and/or density is negative.

To overcome this impasse reliably, I employ the alternative fail-safe strategy of §8.8.4 with DGIGA-AFC (6.30) basis types, rather than the two (much simpler) fail-safe limiters presented next. I use it as follows:

1. Apply all RK stages (to the DGIGA-AFC predictor)
2. Apply the main FCT limiter (§8.7.2)
3. Apply the fail-safe FCT limiter (§8.8.4)
4. Repeat for the next step

8.8.1. Fail-safe slope limiter

If a given element is considered troubled by a sensor, and is subsequently limited using one of the hierarchical limiters (BDF, BSB, Krivodonova's or HWENO), it is possible that only higher order Legendre coefficients have been adjusted, yet unlimited lower order ones are still causing nonphysical primary variable values. This first fail-safe step consists on detecting any of such invalid states that may have remained after applying the main limiter. If found, set $\hat{\boldsymbol{q}}_{3\,k} = \hat{\boldsymbol{q}}_{4\,k} = \ldots = \hat{\boldsymbol{q}}_{\mathrm{J}\,k} = 0$ (Ω_k is the offending element), limit $\hat{\boldsymbol{q}}_{2\,k}$ (slope) according to (8.25) or (8.33), and leave $\hat{\boldsymbol{q}}_{1\,k}$ (element-wide average) untouched.

8.8.2. Last-resort fail-safe limiter

Given that the first fail-safe step (§8.8.1) is also done in characteristic variables, the problem might still persist. If so, a last resort is to set every Legendre coefficient above $j = 1$ to zero. As long as the main limiter is able to ensure that the approximate solution is TVDM, this is guaranteed to result in valid states—at the cost of reducing the offending element to first order (although only temporarily, until the next stage). This is nothing else than degree-adaptation.

8.8.3. Invalid element criteria for inter-cell fail-safe limiters

I propose the following criteria to determine which elements incur into invalid Euler states.

1. First consider each edge in the mesh, $\partial\Omega_k$, and the two elements sharing it: Ω_L and Ω_R; if any of the following holds, flag it as invalid.
   - The density or pressure (either side) is negative:
   $$\rho_L < 0 \cup \rho_R < 0 \cup p_L < 0 \cup p_R < 0\,. \tag{8.62}$$
   - The Riemann problem at this edge would "generate vacuum" [114, equation 4.40]:
   $$\frac{u_R - u_L}{2} \geq \frac{c_L + c_R}{\gamma - 1}\,. \tag{8.63}$$
2. Then, flag as invalid both elements, Ω_{k-1} and Ω_k, adjacent to every invalid edge $\partial\Omega_k$ (as detected in the previous step).
3. Last, flag each of the remaining elements, Ω_k, in which $\hat{q}_{ijk} < 0$ for $i = 1, 3$ and $j = 1, 2, \ldots, \mathrm{J}$. These represent, either:

- Nodal values, if the basis functions are nodal (DGSEM and FR/CPR)
- Control values, if the basis functions are B-splines (DGIGA)

The requirement that density and pressure control values be non-negative can be overly conservative and may lead to unnecessary limiting. Yet, I have found this to be necessary to ensure a valid AFC predictor.

8.8.4. Sub-cell FCT fail-safe limiter

I propose the following straight-forward extension to B-spline basis functions of the fail-safe limiter detailed in [77, §5] (also mentioned in [78, page 15]). In essence, this algorithm consists on gradually reducing the magnitude of antidiffusive fluxes to be applied to the low-order predictor, until no invalid control values remain; in the worst case scenario, the FCT correction is undone entirely. Note that negative pressures and densities are not targeted explicitly: any over/undershoot with respect to the predictor's local maxima/minima is to be removed.

Let $M \in \mathbb{Z}^+$ be a number of limiting stages[15]. The rest of variables are carried over from §8.7. For $m = 1, 2, \dots, M$, and for all k in need of limiting (e.g. designated by a sensor), do:

1. Set i to the vector component associated to a desired control variable[16]
2. Compute a limiting coefficient associated to each pair of control points within mutual nonzero basis function support, with indices r and j, as follows:

$$\beta_{irjk}^{(m)} = \begin{cases} \beta_{irjk}^{(m-1)} & \text{if } \left(\hat{v}_{irk}^{\min} - \epsilon \leq \hat{v}_{irk} \leq \hat{v}_{irk}^{\max} + \epsilon\right) \cap \left(\hat{v}_{ijk}^{\min} - \epsilon \leq \hat{v}_{ijk} \leq \hat{v}_{ijk}^{\max} + \epsilon\right) \\ \frac{m}{M} & \text{otherwise} \end{cases}, \tag{8.64}$$

   where $\beta_{irjk}^{(0)} \coloneqq 0$ and $\epsilon = 10^{-6}$.

3. Limit state vector control values by one stage, using the limiting coefficients just obtained in the previous step:

$$\hat{q}_{irk} \leftarrow \hat{q}_{irk} - \sum_{j \neq r} \beta_{irjk}^{(m)} \frac{f_{irjk}^L}{m_{rk}^L}. \tag{8.65}$$

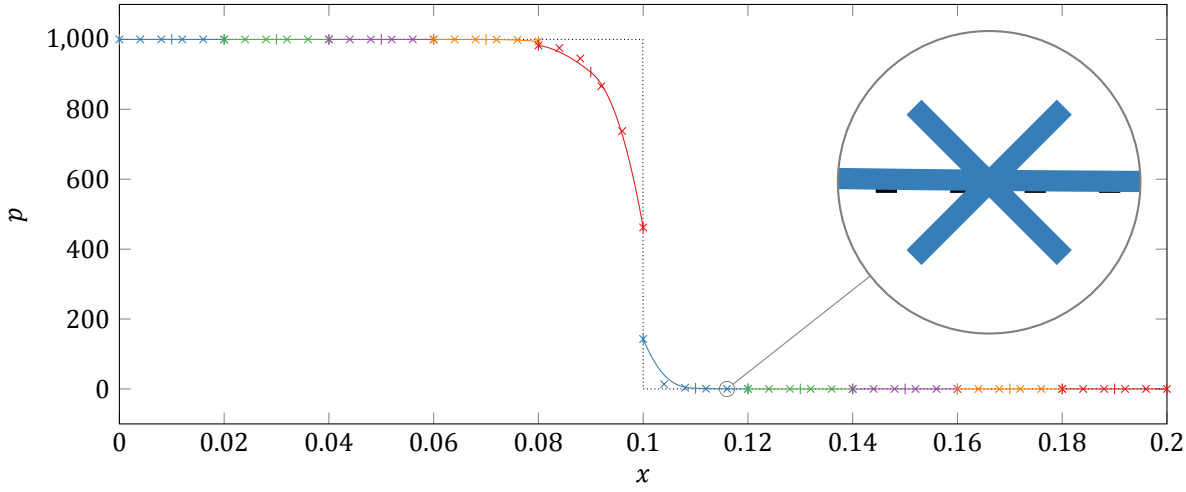
4. Repeat for the next control variable, and so on.

Bear in mind that this fail-safe limiter, as well as FCT correction itself, assumes the predictor to be free of invalid control values in the first place. In some cases (for modal DGIGA-AFC, at least) I have found that the control points of the low-order predictor itself can attain invalid values of density and/or pressure. An example is shown in figure 8.5a. Despite the approximate pressure itself being everywhere positive, one of its expansion coefficients has reached a negative value ($p \approx -0.0058$, encircled and magnified); this will cause the simulation to fail as soon as the solver attempts to compute new antidiffusive fluxes "into" or "out from" this particular control point. This can be avoided by fail-safe limiting the predictor control values as in §8.8.2, leading to the approximate solution shown in subfigure 8.5b.

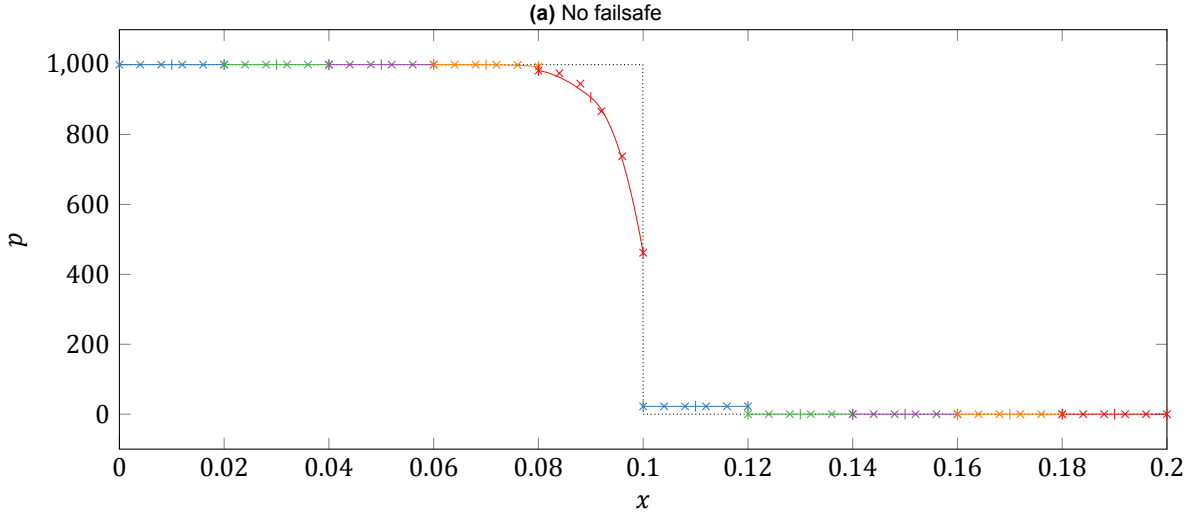
I have been unable to reproduce this problem (the presence in the AFC predictor of negative control values for pressure or density) with *nodal* DGIGA-AFC; it seems to be immune or, at least, less prone to it. Nevertheless, both nodal and modal treatments do require, in addition, fail-safe FCT limiting (§8.8.4) after performing the correction to high-order as detailed in §8.7. Therefore, to ensure stability with AFC-limited DGIGA, it seems necessary that one must apply the inter-cell fail-safes (§8.8.1 and/or §8.8.2) on the low-order predictor, once per time-scheme stage, as well as the sub-cell FCT fail-safe (§8.8.4) to the corrected approximate solution, at the end of each time-step.

[15] It is possible that this fail-safe strategy over-corrects one control value at the expense of making some other one invalid; I have found that using multiple stages prevents this (based on my own experiments, 2 or 3 stages seem to be enough).

[16] For the Euler equations, Kuzmin et al. [77, equation 13] seem to favor the following order: density, pressure and velocity.



(a) No failsafe



(b) Patch-wide failsafe, $\mathrm{p} = 0$

Figure 8.5: A very strong shock wave, discretized via DGIGA-AFC (2 knot spans, $\mathrm{p} = 3$, C^1; modal treatment) using the most robust diffusion function (6.36), *without* attempting to perform the FCT correction to high-order (i.e. the uncorrected low-order predictor is the approximate solution). The numerical solution at $t = 1.80 \times 10^{-4}$ is represented with solid lines; cross markers denote control points and vertical line ones are break points. A change of color indicates a separate patch. The dotted black line is the exact solution at $t = 0$.

II

Experiments

9 Methodology

Part II contains the bulk of numerical results of this report. They can be split in two categories: solution to test cases and semi-analytical modified wavenumber analysis results. The former are always accompanied by a test matrix, in which each test's conditions are summarized. The latter are detailed case-by-case; the general formulation of the approach and all related magnitudes are described in appendix A.

Each of the three research objects under study—DGSEM (§4), FR/CPR (§5) and DGIGA (§6)—has multiple free parameters. The experimental campaign of this thesis is envisioned as a search for optimum configurations of each of these methods, when applied in high-speed, turbulent flow simulations (but remaining within scope, i.e. one spatial dimension and inviscid flow). Each of these optima is a point in the "design" or "parameter" space of the family of methods it belongs. All results assume a uniform discretization, i.e. S_k^h and Δx_k are the same for all k. In the DGIGA case, the distribution of breakpoints is uniform as well (but not the distribution of knots, which may still have multiplicity).

9.1. MATLAB implementation

All results shown in this thesis have been obtained using my own implementation of the formulation of the various schemes presented in part I. The code, written for MATLAB 2017b, is publicly available at `https://github.com/mikiandh/dg-matlab-scripts`. It also includes all scripts used to obtain the experimental results in this second part of the report.

9.2. Test matrices

Every data point in these results corresponds to a numerical solution of a particular model problem—this I will refer to as a *run*[1]. Associated to each run is a set of parameters, uniquely defining every detail of the computation, from the temporal and spatial discretizations to the limiter employed (if any); these are summarized in *test matrices*, tables each row of which uniquely specifies one run (or batch of related runs). Whenever a column is empty, it should be interpreted as having the same value as in the previous row. Every test matrix has the following categories:

Table 9.1: Empty test matrix, for example purposes.

		Time discr.			Space discretization						Limiting	
Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N_{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor

- Figure: the figure in which the corresponding run (marker) and batch (plot line) is shown
- Problem: the initial-boundary value problem solved, among those in §9.3

[1] To clarify, consider for example figure 10.1. In it, each point corresponds to the L^2 error obtained after simulating 2 units of time with a given time-step size. This means that over a hundred simulation runs have been necessary to obtain its data—and, reciprocally, over a hundred simulation results are neatly summarized into one single picture.

- Time discretization
  - RK: which of the SSP-RK schemes in §7.3 is used (order and number of stages)
  - Δt: time-step size (can be either fixed or relative to largest stable one)
  - ΔT: simulated span of time, i.e. $t - t_0$ at the instant when results are obtained
- Space discretization
  - Method: DGSEM, FR/CPR or DGIGA (modal formulation, unless indicated otherwise)
  - $\mathrm{N_{dofs}}$: total number of degrees of freedom, per equation
  - K: number of elements in the mesh
    - $\mathrm{N_{dofs}} \coloneqq \mathrm{JK}$, where J is the number of basis functions per element
  - p: polynomial degree
    - If $\mathrm{J} > \mathrm{p} + 1$, the discretization method is DGIGA
  - η: correction function parameter; only specified[2] for FR/CPR
  - k: number of breakpoint spans per element; only DGIGA can have $\mathrm{k} > 1$
  - $\varkappa$: smoothness of the approximate solution *within* an element (differentiability class $C^{\varkappa}$)
    - For DGIGA, $\varkappa < \mathrm{p}$; in particular, $\varkappa = \mathrm{p} - 1 \implies \mathrm{J} = \mathrm{k} + \mathrm{p}$
- Limiting.
  - Limiter: if any, one of those in §8
  - Sensor: idem, for §8.3

Several runs are grouped into a *batch*—a row of the test matrix—and are associated with a figure (as indicated). Every one of such batches shows the change in one dependent quantity (e.g. a measure of the error) as some parameter of the run changes discretely (i.e. independent variable, indicated as a range), for fixed values of the remaining free parameters. These can be seen as a series of 2D slices of some multidimensional function the domain of which is the parameter space of the method.

9.3. List of model problems

The following are all combinations of *boundary conditions* (BC), *initial condition* (IC) and *conservation law* (PDE) used in this thesis. In all cases $t_0 = 0$ (time starts at zero).

9.3.1. Monochromatic wave (linear)

A sinusoidal signal of wavenumber $\kappa = n\pi$ ($n \in \mathbb{Z}$), centered around the unit amplitude:

$$\text{PDE}: \quad (2.19), \quad a = 1\,, \tag{9.1a}$$

$$\text{IC}: \quad q^0(x) = 1 + 0.1\sin(n\pi x)\,, \tag{9.1b}$$

$$\text{BC}: \quad \text{Periodic}, \quad \Omega = [-1, 1]\,, \tag{9.1c}$$

which travels unmodified at a rate of one domain length every two time units, towards $+\infty$. Equation (8.3) indicates that the total variation of the exact solution *per period* is $\frac{2n}{5}$.

9.3.2. Monochromatic wave (nonlinear)

The same initial and boundary conditions as (9.1), but evolved according to the inviscid Burgers equation instead:

$$\text{PDE}: \quad (2.29)\,, \tag{9.2a}$$

$$\text{IC}: \quad u^0(x) = 1 + 0.1\sin(n\pi x)\,, \tag{9.2b}$$

$$\text{BC}: \quad \text{Periodic}, \quad \Omega = [-1, 1]\,. \tag{9.2c}$$

According to (2.30), the solution of this case breaks at $t_{\text{shock}} = \frac{10}{n\pi}$. Until then, the total variation of the exact solution is preserved and, hence, equal to that of (9.1).

[2] The symbol '-' indicates that some parameter is not defined for the current run—in the case of limiters and sensors, it simply means that none have been used.

9.3.3. Monochromatic wave (Euler)

Setup for the Euler equations that mimics problem (9.3.1)—density is a monochromatic wave undergoing linear advection. The actual value of pressure is irrelevant; it is only necessary that it is uniform across the domain. Other initial density distributions and (uniform) initial velocities would have been equally valid; these reproduce (9.1) specifically.

$$\text{PDE}: \qquad (2.39)\,, \tag{9.3a}$$

$$\text{IC}: \qquad \boldsymbol{v}(x) = \begin{bmatrix} \rho^0(x) \\ u^0 \\ p^0 \end{bmatrix} = \begin{bmatrix} 1 + 0.1\sin(n\pi x) \\ 1 \\ 1 \end{bmatrix}, \tag{9.3b}$$

$$\text{BC}: \qquad \text{Periodic}, \quad \Omega = [-1, 1]\,. \tag{9.3c}$$

This initial condition is borrowed from [48, appendix C]. The exact solution to this problem is:

$$\boldsymbol{v}(t,x) = \begin{bmatrix} \rho^0(x-t) \\ 1 \\ 1 \end{bmatrix}. \tag{9.4}$$

Proof. Clearly, $\boldsymbol{v}(t,x)$ satisfies the initial and boundary conditions of (9.3). It also satisfies (2.39), since:

$$\frac{\partial \rho}{\partial t} = -\frac{\mathrm{d}\rho^0}{\mathrm{d}x}\,, \qquad \frac{\partial \rho}{\partial x} = \frac{\mathrm{d}\rho^0}{\mathrm{d}x}\,, \tag{9.5}$$

and:

$$\boldsymbol{q} = \begin{bmatrix} \rho \\ \rho \\ \frac{1}{2}\rho + \frac{1}{\gamma-1} \end{bmatrix}, \qquad \boldsymbol{f} = \begin{bmatrix} \rho \\ \rho + 1 \\ \frac{1}{2}\rho + \frac{\gamma}{\gamma-1} \end{bmatrix}. \tag{9.6}$$

And, because $\rho^0(x)$ is *analytic*, this solution is unique [114, §19.3]. □

9.3.4. Gaussian hump (linear)

The linear advection equation with periodic boundary conditions, and the following initial condition:

$$\text{PDE}: \qquad (2.19), \quad a = 1\,, \tag{9.7a}$$

$$\text{IC}: \qquad u^0(x) = \mathrm{e}^{-\frac{9\pi}{4}x^2}\,, \tag{9.7b}$$

$$\text{BC}: \qquad \text{Periodic}, \quad \Omega = [-1, 1]\,, \tag{9.7c}$$

which is a Gaussian bell curve of variance $\sigma = \frac{1}{3}\sqrt{\frac{2}{\pi}}$, chosen so that $\int_{-\infty}^{\infty} u^0\,\mathrm{d}x = \frac{2}{3}$ (>99.98 % of this area is contained in Ω).

9.3.5. Gaussian hump (nonlinear)

Identical to problem (9.7), but for the Burgers equation:

$$\text{PDE}: \qquad (2.29)\,, \tag{9.8a}$$

$$\text{IC}: \qquad u^0(x) = \mathrm{e}^{-\frac{9\pi}{4}x^2}\,, \tag{9.8b}$$

$$\text{BC}: \qquad \text{Periodic}, \quad \Omega = [-1, 1]\,. \tag{9.8c}$$

Using (8.3), $\left\|u^0\right\|_{\mathrm{TV}} = 2$; until the breaking time, the total variation per period of the exact solution is >99.80 % of the previous. According to (2.30), this problem's solution becomes discontinuous at $t_{\text{shock}} = \frac{1}{3}\sqrt{\frac{2\mathrm{e}}{\pi}} \approx 0.4385$.

9.3.6. Triangular pulse (linear)

Analogous to (9.7), but with a triangular initial condition (instead of a smooth hump):

$$\text{PDE}: \quad (2.19), \quad a = 1\,, \tag{9.9a}$$
$$\text{IC}: \quad q^0(x) = 1 - |x|\,, \tag{9.9b}$$
$$\text{BC}: \quad \text{Periodic}, \quad \Omega = [-1, 1]\,. \tag{9.9c}$$

Note that the exact solution of this problem is continuous but non-smooth. Its total variation per period is $\|q^0\|_{\text{TV}} = 2$.

9.3.7. Triangular pulse (nonlinear)

Counterpart to (9.9) for the Burgers equation and analogue, in turn, to (9.8), i.e. :

$$\text{PDE}: \quad (2.29)\,, \tag{9.10a}$$
$$\text{IC}: \quad u^0(x) = 1 - |x|\,, \tag{9.10b}$$
$$\text{BC}: \quad \text{Periodic}, \quad \Omega = [-1, 1]\,. \tag{9.10c}$$

My interest in this problem stems from the fact that its exact solution can be deduced for all times, $t > t_{\text{shock}}$ included. The method of characteristics readily shows that the initial triangular pulse deforms—maintaining its amplitude—until it attains a sawtooth or N-wave shape, at $t_{\text{shock}} = 1$, which is then retained. It can be shown that such an N-wave experiences a *decay in amplitude* such that, for all x, $u(t, x) \propto 1/t$ [112, theorem 16.14, page 298]. Thanks to the simple geometry of this solution, I have been able to determine this damping rate *exactly* through careful numerical experimentation; it is such that the total variation within Ω is $\frac{2}{t+1}$ for $t \geq 1$. All in all, the exact solution to (9.10) can be expressed as:

$$u(t,x) = \begin{cases} \frac{1+x}{1+t} & \text{if } x \leq t < 1\,, \\ \frac{1-x}{1-t} & \text{if } x > t < 1\,, \\ \frac{1}{2} + \frac{\left(\left(x + \frac{3}{2} - \frac{t}{2}\right) \bmod 2\right) - 1}{1+t} & \text{if } t \geq 1\,. \end{cases} \tag{9.11}$$

9.3.8. Jiang-Shu problem

An initial condition for the linear advection equation first proposed by Jiang and Shu [62], made up (left to right) of a superposition of three Gaussians, a square pulse, a triangular pulse, and a superposition of three half-ellipses.

$$\text{PDE}: \quad (2.19), \quad a = 1\,, \tag{9.12a}$$

$$\text{IC}: \quad q^0(x) = \begin{cases} \frac{1}{6}\left(G(x, \alpha - \delta) + 4G(x, \alpha) + G(x, \alpha + \delta)\right) & \text{if } -0.8 \leq x \leq -0.6 \\ 1 & \text{if } -0.4 \leq x \leq -0.2 \\ 1 - 10\,|x - 0.1| & \text{if } 0 \leq x \leq 0.2 \\ \frac{1}{6}\left(F(x, \beta - \delta) + 4F(x, \beta) + F(x, \beta + \delta)\right) & \text{if } 0.4 \leq x \leq 0.6 \\ 0 & \text{otherwise} \end{cases}, \tag{9.12b}$$

$$\text{BC}: \quad \text{Periodic}, \quad \Omega = [-1, 1]\,, \tag{9.12c}$$

where $G(x, \alpha) = \mathrm{e}^{-\frac{\ln 2}{36\delta^2}(x-\alpha)^2}$, $F(x, \beta) = \sqrt{\max\{0, 1 - 100(x-\beta)^2\}}$, $\alpha = -0.7$, $\beta = 0.5$ and $\delta = 0.005$.

This problem's exact solution involves the transport of piece-wise sharp yet smooth features, joined together in either C^{-1} or C^0 fashion. It is a rather challenging test case, especially if simulated over long times, as its extrema tend to be (wrongfully) targeted by sensors and limiters.

9.3.9. Toro's transonic shock tube

A minor modification of the classical shock tube test case for the Euler equations due to Sod [113], used in [114]. The solution includes, left to right, a left-going expansion wave (along which the flow speed crosses the speed of sound, inducing nonphysical expansion shocks in some Riemann solvers), a contact discontinuity and a shock wave (both moving towards the right).

$$\text{PDE}: \quad (2.39)\,, \tag{9.13a}$$

$$\text{IC}: \quad \boldsymbol{v}^0(x) = \begin{cases} \begin{bmatrix} 1 & 0.75 & 1 \end{bmatrix}^\intercal & \text{if } 0 \leq x \leq 0.5 \\ \begin{bmatrix} 0.125 & 0 & 0.1 \end{bmatrix}^\intercal & \text{if } 0.5 < x \leq 1 \end{cases}, \tag{9.13b}$$

$$\text{BC}: \quad \text{Farfield}, \quad \Omega = [0,1]: \quad \boldsymbol{v}(t,x) = \boldsymbol{v}^0(x) \text{ for } x \in \partial\Omega\,. \tag{9.13c}$$

All problems for the Euler equations the initial condition of which is a pair of piece-wise constant states can be solved exactly by means of an exact Riemann solver. For this purpose, I have implemented (in MATLAB) an exact Riemann solver based on the FORTRAN code in [114, §4.9].

9.3.10. The 1-2-3 problem

Another test case for the Euler system, due to Einfeldt et al. [33], known as the "1-2-3" problem because of its initial state vector. In it, the exact solution consists on two strong rarefaction waves moving away from each other, leaving a near-vacuum state in the middle. The goal in this case is to test the robustness of a discretization by forcing it to approximate near-zero densities and pressures; some Riemann solvers (e.g. Roe's) are known to fail under these conditions.

$$\text{PDE}: \quad (2.39)\,, \tag{9.14a}$$

$$\text{IC}: \quad \boldsymbol{q}^0(x) = \begin{cases} \begin{bmatrix} 1 & -2 & 3 \end{bmatrix}^\intercal & \text{if } 0 \leq x \leq 0.5 \\ \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^\intercal & \text{if } 0.5 < x \leq 1 \end{cases}, \tag{9.14b}$$

$$\text{BC}: \quad \text{Farfield}, \quad \Omega = [0,1]: \quad \boldsymbol{q}(t,x) = \boldsymbol{q}^0(x) \text{ for } x \in \partial\Omega\,. \tag{9.14c}$$

Regarding this problem's exact solution, the same as in 9.3.9 applies.

9.3.11. Blast wave interaction

The term "blast wave" is used to refer to a region of increased pressure moving away *supersonically* from an origin point (essentially, a very strong shock wave). These can be the result of a sudden and concentrated release of a large amount of energy, i.e. an explosion, hence their name.

This problem, often referred by the name of the authors that first studied it at length, Woodward and Colella [131], simulates the frontal collision between two of such blast waves.

$$\text{PDE}: \quad (2.39)\,, \tag{9.15a}$$

$$\text{IC}: \quad \boldsymbol{v}^0(x) = \begin{cases} \begin{bmatrix} 1 & 0 & 1000 \end{bmatrix}^\intercal & \text{if } 0 \leq x < 0.1 \\ \begin{bmatrix} 1 & 0 & 0.01 \end{bmatrix}^\intercal & \text{if } 0.1 \leq x < 0.9 \\ \begin{bmatrix} 1 & 0 & 100 \end{bmatrix}^\intercal & \text{if } 0.9 \leq x \leq 1 \end{cases}, \tag{9.15b}$$

$$\text{BC}: \quad \text{Reflective}, \quad \Omega = [0,1]\,. \tag{9.15c}$$

This is a very challenging test case, requiring robust nonlinear stabilization techniques if attempted with a high-order discretization. Unlike the previous two, its exact solution is not available (the exact Riemann solver mentioned in 9.3.9 is no longer suitable). In its place, I simply use a very fine numerical solution[3].

[3] Discretized into 2500 DGSEM *linear* elements ($\mathrm{N}_{\mathrm{dofs}} = 5000$), and obtained employing the TVD limiter (§8.4) in combination with the KXRCF sensor (§8.3.1).

9.3.12. Acoustic wave–shock wave interaction

A test case simulating the interaction between a right-moving shock wave and a stationary density fluctuation of sinusoidal form. This roughly mimics a shock-turbulence interaction. Attributed to Shu and Osher [110].

$$\text{PDE}: \quad (2.39)\,, \tag{9.16a}$$

$$\text{IC}: \quad \boldsymbol{v}^0(x) = \begin{cases} \begin{bmatrix} 3.857143 & 2.629369 & 10.33333 \end{bmatrix}^\intercal & \text{if } x < -4 \\ \begin{bmatrix} 1 + 0.2\sin(5x) & 0 & 1 \end{bmatrix}^\intercal & \text{if } x \geq -4 \end{cases}, \tag{9.16b}$$

$$\text{BC}: \quad \text{Farfield}, \quad \Omega = [-5, 5]: \quad \boldsymbol{v}(t, x) = \boldsymbol{v}^0(x) \text{ for } x \in \partial\Omega\,. \tag{9.16c}$$

This problem is often used to test the degree to which a nonlinear stabilization strategy hinders accuracy in smooth regions. It is hence similar to (9.12) in spirit, but using more realistic (nonlinear) physics. No exact solution exists for this case either. Instead, I show a numerical reference solution—same discretization as that of (9.15).

10

Order of Accuracy

In this chapter, the parameter space of each method is explored by observing how each of their parameters affects the numerical error, on a smooth solution test case. This encompasses:

- Order of accuracy in time for all SSP-RK methods, in §10.1 (only for a particular DGIGA configuration and in a linear setting).
- Accuracy vs. time-step size and number of degrees of freedom (changing either p or K) for DGSEM (§10.2), in the nonlinear setting of Burgers equation (2.29).
- Idem for FR/CPR, adding one dimension: the choice of correction function (§10.3).
- Idem for DGIGA, but with two additional dimensions over DGSEM: number of breakpoint spans and smoothness class (§10.4).

Most results in this chapter involve the computation of the L^2 norm of the error between exact and discrete solutions. This norm, for a given time t, is defined as [129, p. 816 (bottom)]:

$$\left\| q(t,x) - q^h(t,x) \right\|_2 \coloneqq \sqrt{\frac{\int_\Omega \left(q(t,x) - q^h(t,x) \right)^2 \, \mathrm{d}x}{\int_\Omega \, \mathrm{d}x}} \,. \tag{10.1}$$

The integral in the numerator is approximated using breakpoint span–wise adaptive Gauss-Kronrod quadrature [106] with 10^{-13} and 10^{-9} absolute and relative tolerances[1], respectively (also when projecting initial conditions, see §3.5).

10.1. Time schemes

The order of accuracy in time (i.e. for varying Δt) of each SSP-RK scheme (see §7.3) is verified numerically for the setup summarized in table 10.1. It is seen to match the formal one in every case—for the particular spatial discretization and (linear) problem tested. Since the implementation of temporal schemes is unique, any deviations from these trends observed in upcoming results will be attributed to the spatial discretization used and/or the test case solved.

[1] All reported L^2 error results stagnate near a common lower bound; I attribute this to said integration tolerance, given the fact that $\sqrt{10^{-13}} \approx 10^{-7}$. Note, however, that lowering this further (to machine precision) would not be much better, since $\sqrt{10^{-16}} \approx 10^{-8}$.

Table 10.1: Verifying the order of accuracy of the SSP-RK schemes.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	10.1	(9.7)	1(1)	10^{-6}–10^{-1}	2	DGIGA	30	1	3	-	27	2	-	-
2			2(2)	10^{-4}–10^{-1}										
3			3(3)	10^{-3}–10^{-1}										
4			4(5)	10^{-2}–1										
5			4(10)	10^{-2}–1										

10.2. DGSEM

This section's test matrix is table 10.2. Runs are divided into 12 batches; the first 6 show the time-independence of the rest. Three of the remaining 6 explore the K dimension, and the remaining three focus instead on refining p at fixed values of K. DGSEM is found to achieve its formal order of $p+1$, at least for degrees up to 4; the exponential convergence associated with p-refinement is also observed. Notice that Lagrange polynomial basis functions can reach degrees as high as 119 without issues.

Table 10.2: Exploring DGSEM's parameter space.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	10.2	(9.8)	3(3)	10^{-5}–10^{-2}	0.25	DGSEM	900	300	2	-	1	∞	-	-
2								225	3					
3								180	4					
4							120	20	5					
5								10	11					
6								1	119					
7	10.3			10^{-4}			3–900	1–300	2					
8							4–900	1–225	3					
9							5–900	1–180	4					
10	10.4						20–120	20	0–5					
11							10–120	10	0–11					
12							1–120	1	0–119					

10.3. FR/CPR

Table 10.3 expands upon the previous by incorporating the four main correction functions of FR/CPR. It is quite apparent for this method that the larger η is, the lower the accuracy for a given N_{dofs}, with $\frac{\eta_-}{2}$ and η_{DG} (i.e. DGSEM) results being very close to each other.

Table 10.3: Exploring FR/CPR's parameter space.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	10.5	(9.8)	3(3)	10^{-5}–10^{-2}	0.25	FR/CPR	900	180	4	$\eta_-/2$	1	∞	-	-
2										η_{Ga}				
3										η_2				
4										η_∞				
5							120	10	11	$\eta_-/2$				
6										η_{Ga}				
7										η_2				
8										η_∞				
9	10.6			10^{-4}			3–900	1–300	2	$\eta_-/2$				
10										η_{Ga}				
11										η_2				
12										η_∞				
13							4–900	1–225	3	$\eta_-/2$				
14										η_{Ga}				

(continues in the next page)

Table 10.3: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
15										η_2				
16										η_∞				
17							5–900	1–180	4	$\eta_-/2$				
18										η_{Ga}				
19										η_2				
20										η_∞				
21	10.7						20–120	20	0–5	$\eta_-/2$				
22										η_{Ga}				
23										η_2				
24										η_∞				
25							10–120	10	0–11	$\eta_-/2$				
26										η_{Ga}				
27										η_2				
28										η_∞				
29							1–120	1	0–119	$\eta_-/2$				
30										η_{Ga}				
31										η_2				
32										η_∞				

10.4. DGIGA

DGIGA expands considerably the design space of the discretization, and so table 10.4 is rather long; hopefully helpful in better transmitting the concept of DGIGA's "4-dimensional refinement space" is the comparison between batches 46 to 49 (figure 10.12). As before, all spatial refinement results are checked to be Δt-independent (batches 1 to 46). The former are divided into:

- K or Δx–refinement: increasing the number of DG elements—i.e. patches—in the domain.
- k-refinement: adding more polynomial segments to the B-spline basis functions, keeping their degree and smoothness fixed; sometimes referred to in the IGA literature as *knot insertion* [28, §2.1.4.1].
- p-refinement (i.e. order elevation [28, §2.1.4.2]): increase the degree of the basis functions while adding knot multiplicities to preserve their smoothness.
- $\varkappa$-refinement (necessarily combined with increased p): both degree and smoothness are increased, with no addition of knots [28, §2.1.4.3].

In contrast to the previous two methods, DGIGA seems to become unstable under some combinations of its parameters (unrelated to time-step size limitations). It appears to experience a weak (nonlinear) instability for high $\varkappa$ and $\mathrm{p} > 2$, and which seems less pronounced if multiple patches are used. Note that this can not be attributed directly to the ill-conditioning of the discrete operators in DGIGA (e.g. Vandermonde matrix, see figure 6.11), because it occurs even for moderate condition numbers.

Lastly, the direct comparisons with its modal counterpart (6.14) clearly show that the nodal approach (6.15) is unsuitable for high-order approximations; these results indicate that the latter is so diffusive that it reduces to second order, regardless of the basis function degree (in the unlimited, nonlinear, smooth-solution case under consideration).

Table 10.4: Exploring DGIGA's parameter space.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	10.8	(9.8)	3(3)	10^{-5}–10^{-2}	0.25	DGIGA	40	10	2	-	2	1	-	-
2							120	30						
3								10	6					
4									2		10			

(continues in the next page)

Table 10.4: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
5									10		2	9		
6	10.9						900	300	2		1	∞		
7								100			4	0		
8								150				1		
9							891	27			16	0		
10							900	50				1		
11								225	3			∞		
12							897	69			4	0		
13							896	128				2		
14							882	18			16	0		
15							893	47				2		
16								180	4			∞		
17							884	52			4	0		
18							896	112				3		
19							845	13			16	0		
20							900	45				3		
21	10.10						899	1	2		449	0		
22							900				898	1		
23							890	10			44	0		
24											87	1		
25							900	20			22	0		
26											43	1		
27							898	1	3		299	0		
28							56				53	2		
29							790	10			26	0		
30							360				33	2		
31							860	20			14	0		
32							760				35	2		
33							897	1	4		224	0		
34							13				9	3		
35							890	10			22	0		
36							200				16	3		
37							900	20			11	0		
38							400				16	3		
39	10.11						340	20	16		1	∞		
40							620		15		2	0		
41							900		11		4	0		
42							820		5		8	0		
43							280		12		2	11		
44							300		11		4	10		
45							280		6		8	5		
46	10.12			10^{-4}			40–120	10–30	2		2	1		
47								10	2–6					
48									2		2–10			
49									$\varkappa+1$		2	1–9		
50	10.13						3–900	1–300	2		1	∞		
51							4–900	1–225	3					
52							5–900	1–180	4					
53						(nodal)	3–900	1–300	2					
54							4–900	1–225	3					
55							5–900	1–180	4					
56	10.14					(modal)	3–900	1–300	2					
57							9–900	1–100			4	0		
58							6–900	1–150				1		
59							33–891	1–27			16	0		
60							18–900	1–50				1		
61	10.15						4–900	1–225	3		1	∞		
62							13–897	1–69			4	0		
63							7–896	1–128				2		
64							49–882	1–18			16	0		

(continues in the next page)

Table 10.4: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
65							19–893	1–47				2		
66	10.16						5–900	1–180	4		1	∞		
67							17–884	1–52			4	0		
68							8–896	1–112				3		
69							65–845	1–13			16	0		
70							100–900	5–45				3		
71	10.17						2–900	1	1		1–899	0		
72							3–899		2		1–449			
73							4–898		3		1–299			
74							5–897		4		1–224			
75						(nodal)	3–899		2		1–449			
76							4–898		3		1–299			
77							5–897		4		1–224			
78	10.18					(modal)	2–900		1		1–899			
79							3–900		2		1–898	1		
80							4–900		3		1–897	2		
81							5–900		4		1–896	3		
82						(nodal)	3–900		2		1–898	1		
83							4–900		3		1–897	2		
84							5–900		4		1–896	3		
85	10.19					(modal)	3–899		2		1–449	0		
86							3–900				1–898	1		
87							30–890	10			1–44	0		
88							30–890				1–87	1		
89							60–900	20			1–22	0		
90							60–900				1–43	1		
91	10.20						4–898	1	3		1–299	0		
92							4–56				1–53	2		
93							40–790	10			1–26	0		
94							40–360				1–33	2		
95							80–860	20			1–14	0		
96							80–760				1–35	2		
97	10.21						5–897	1	4		1–224	0		
98							5–13				1–9	3		
99							50–890	10			1–22	0		
100							50–200				1–16	3		
101							100–900	20			1–11	0		
102							100–400				1–16	3		
103	10.22						2–9	1	1–8		1	0		
104							3–17				2			
105							5–61		1–15		4			
106							9–129		1–16		8			
107							40–340	20			1			
108							60–620		1–15		2			
109							100–900		1–11		4			
110							180–820		1–5		8			
111	10.23						40–260		$\varkappa+1$		1	0–11		
112							60–280				2	0–11		
113							100–320				4	0–10		
114							180–280				8	0–5		

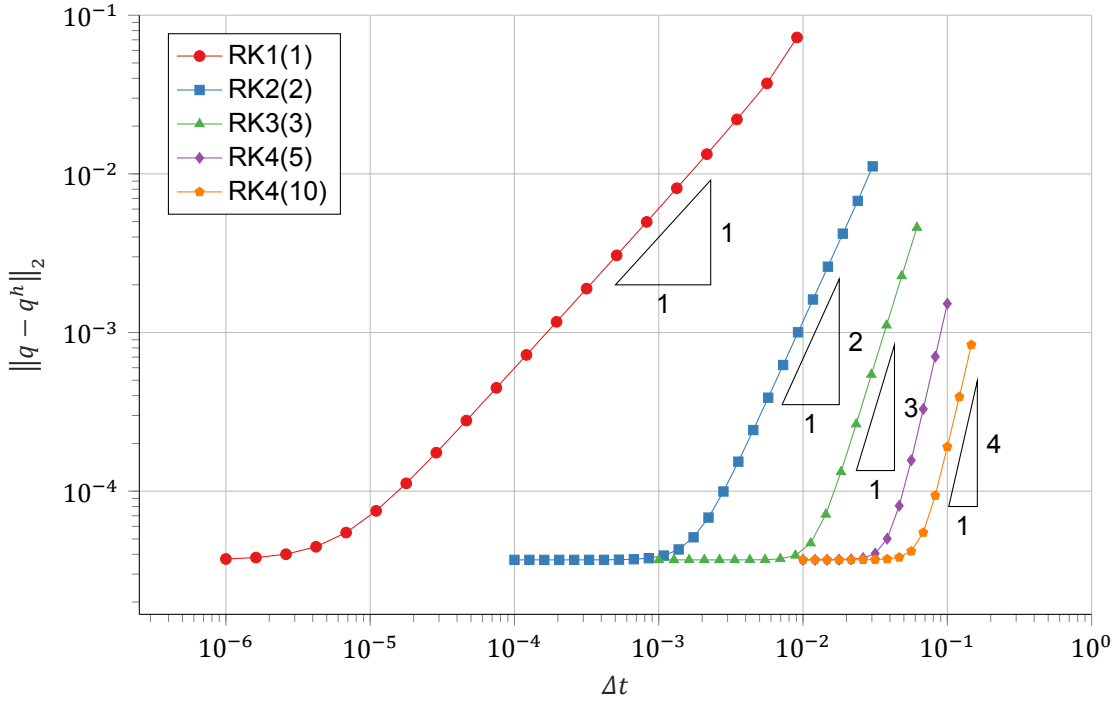


Figure 10.1: Order of accuracy of the SSP-RK schemes (table 10.1).

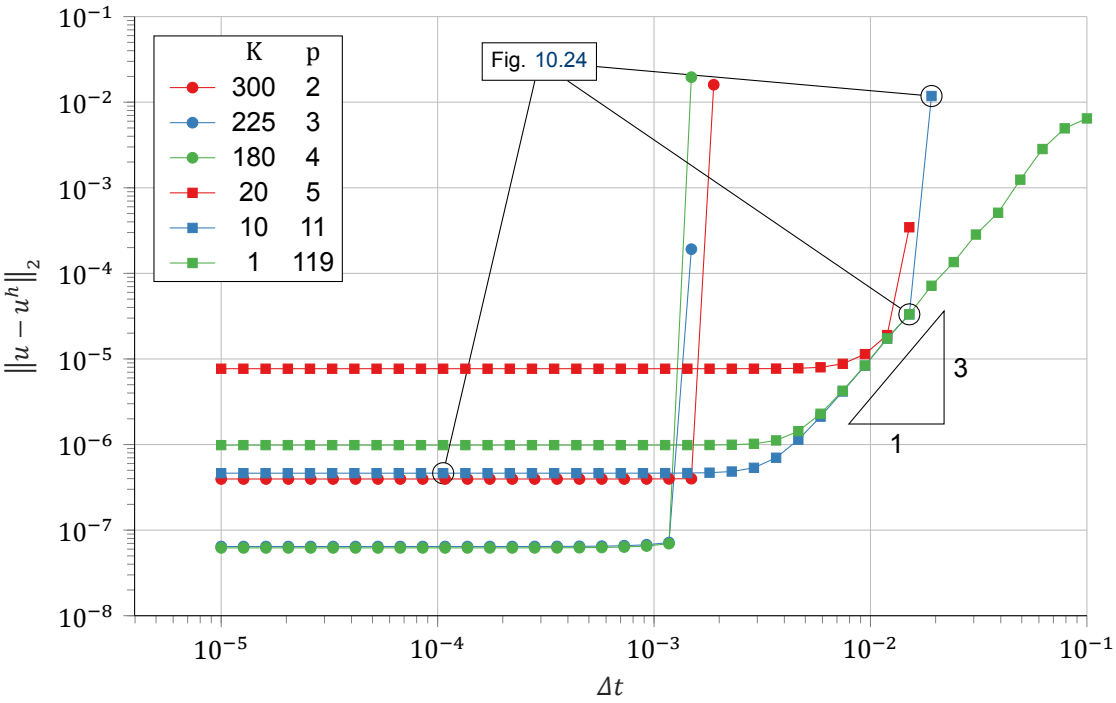


Figure 10.2: DGSEM, time-step size independence (table 10.2, batches 1–6).

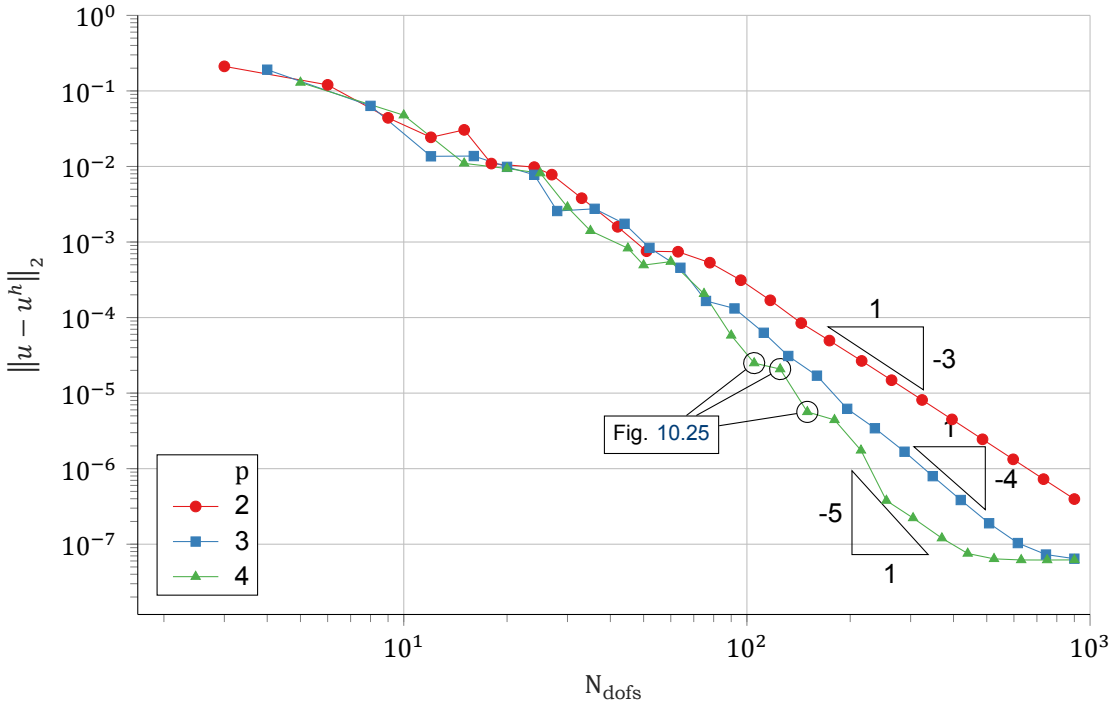


Figure 10.3: DGSEM, Δx refinement (table 10.2, batches 7–9).

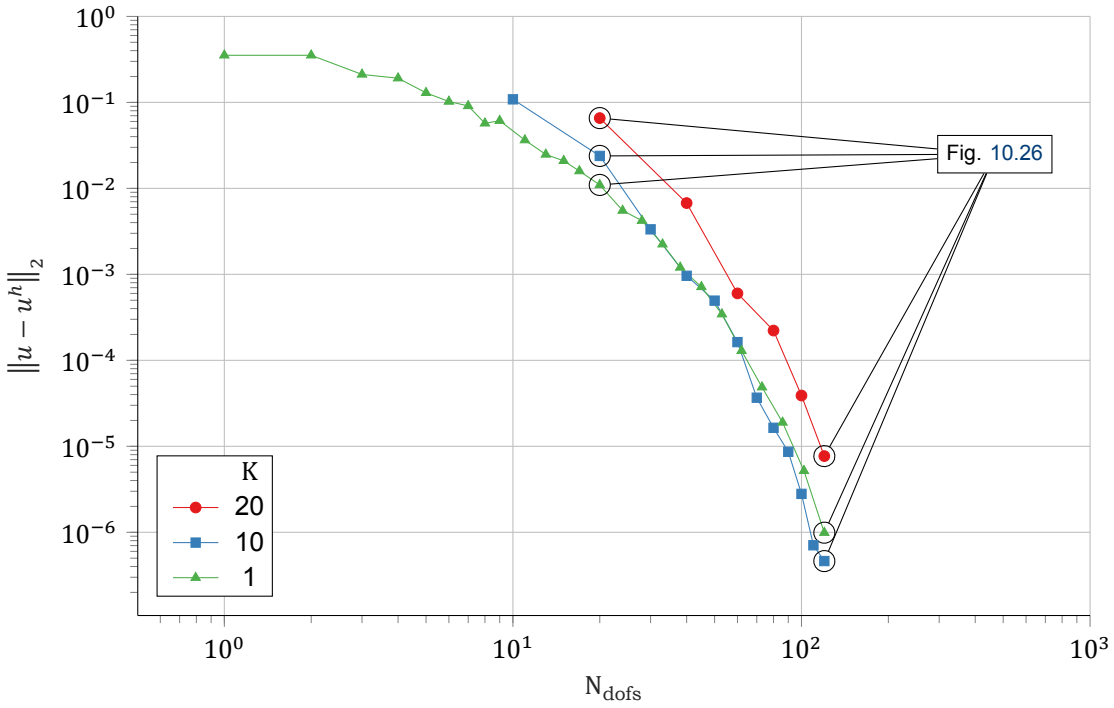


Figure 10.4: DGSEM, p refinement (table 10.2, batches 10–12).

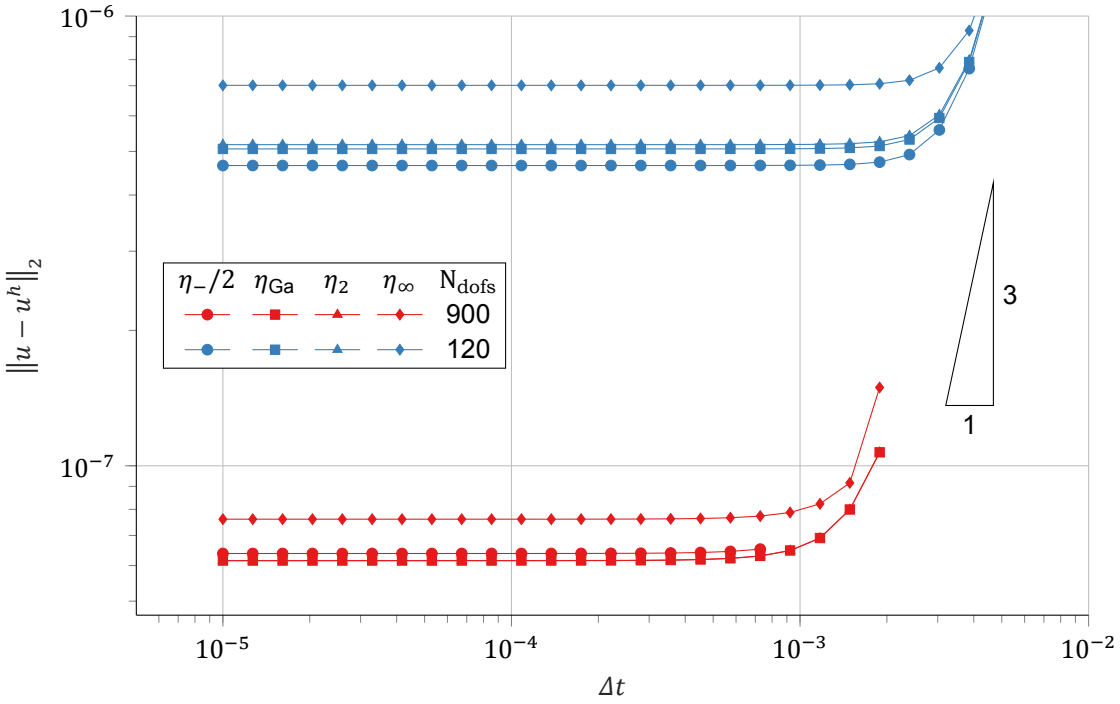


Figure 10.5: FR/CPR, time-step size independence (table 10.3, batches 1–8).

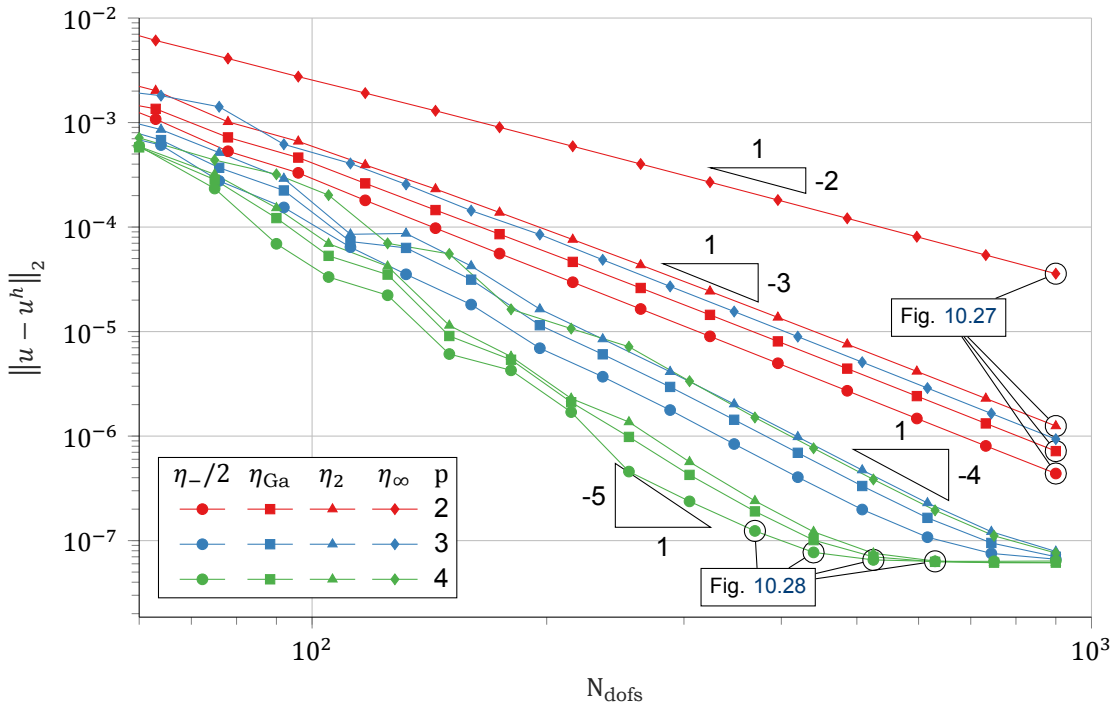


Figure 10.6: FR/CPR, Δx refinement (table 10.3, batches 9–20).

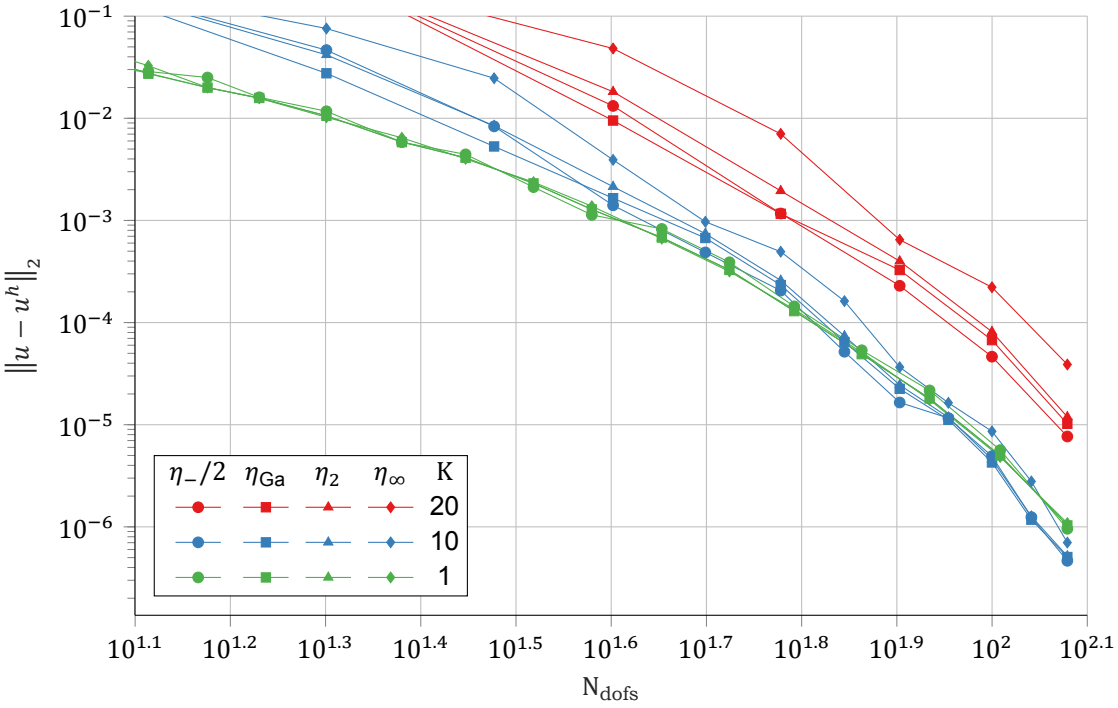


Figure 10.7: FR/CPR, p refinement (table 10.3, batches 21–32).

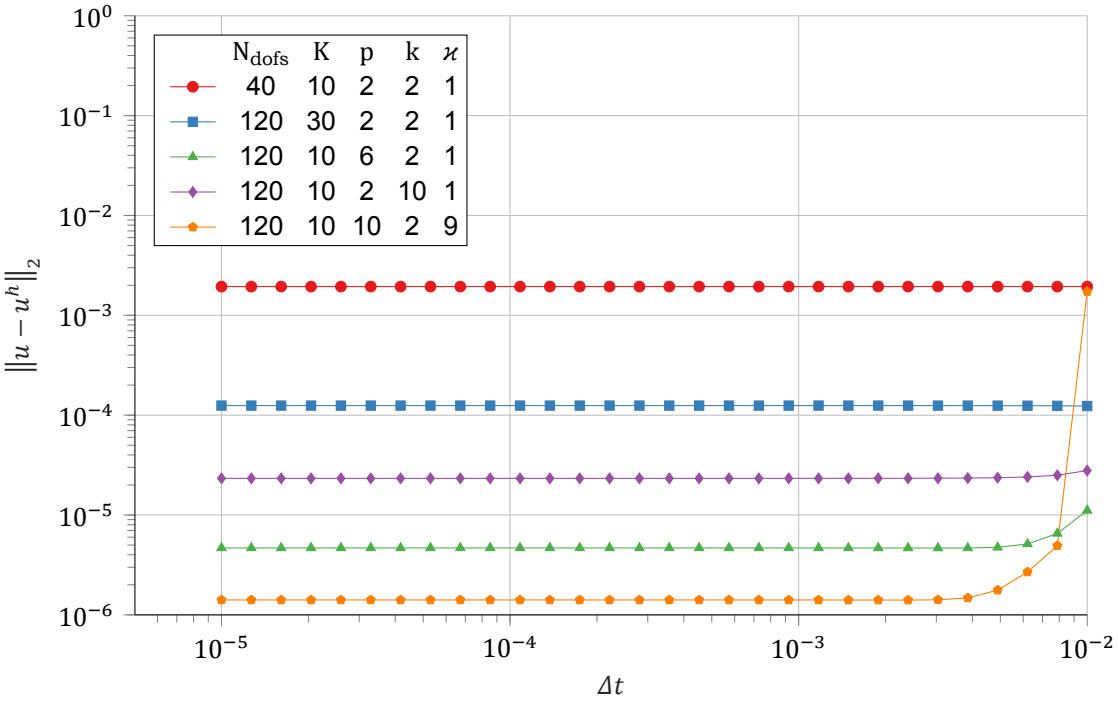


Figure 10.8: DGIGA's four refinement directions, time-step size independence (table 10.4, batches 1–5).

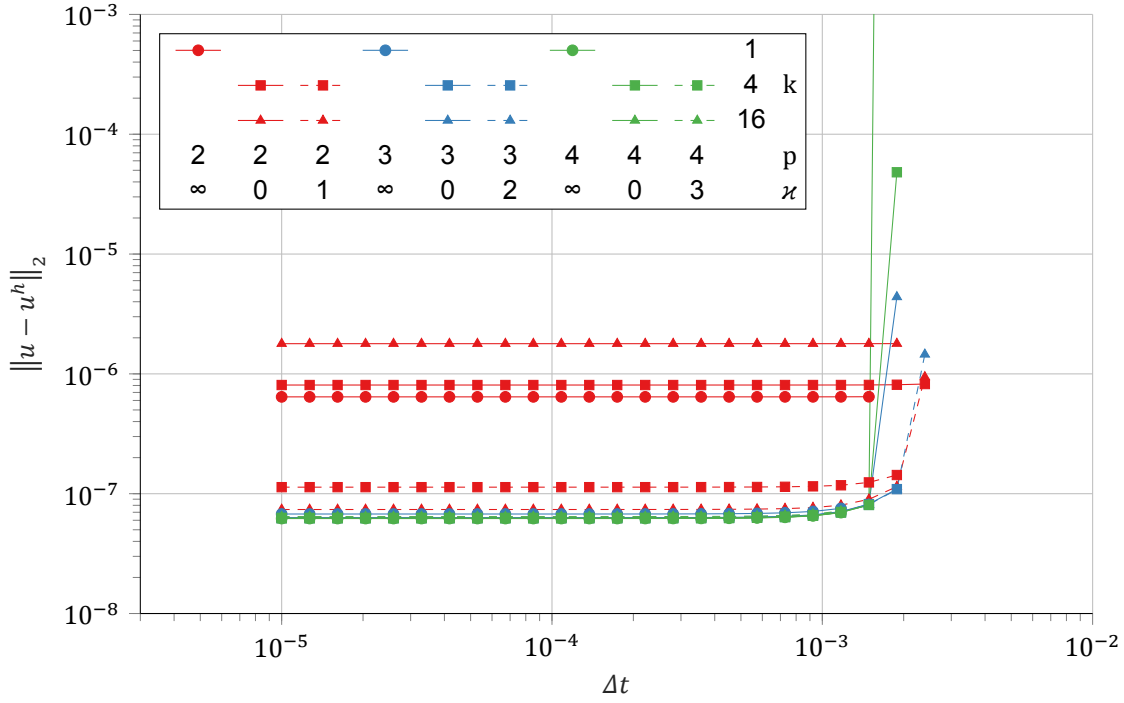


Figure 10.9: DGIGA, $\mathrm{K} \gg 1$, time-step size independence (table 10.4, batches 6–20).

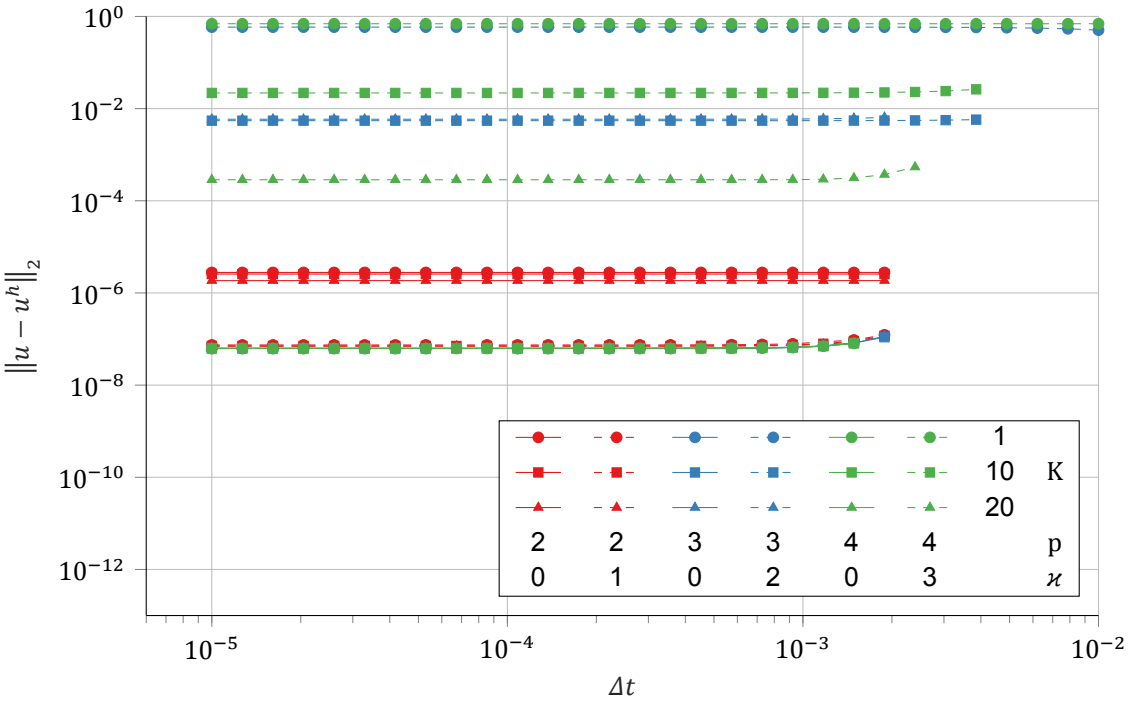


Figure 10.10: DGIGA, $\mathrm{k} \gg 1$, time-step size independence (table 10.4, batches 21–38).

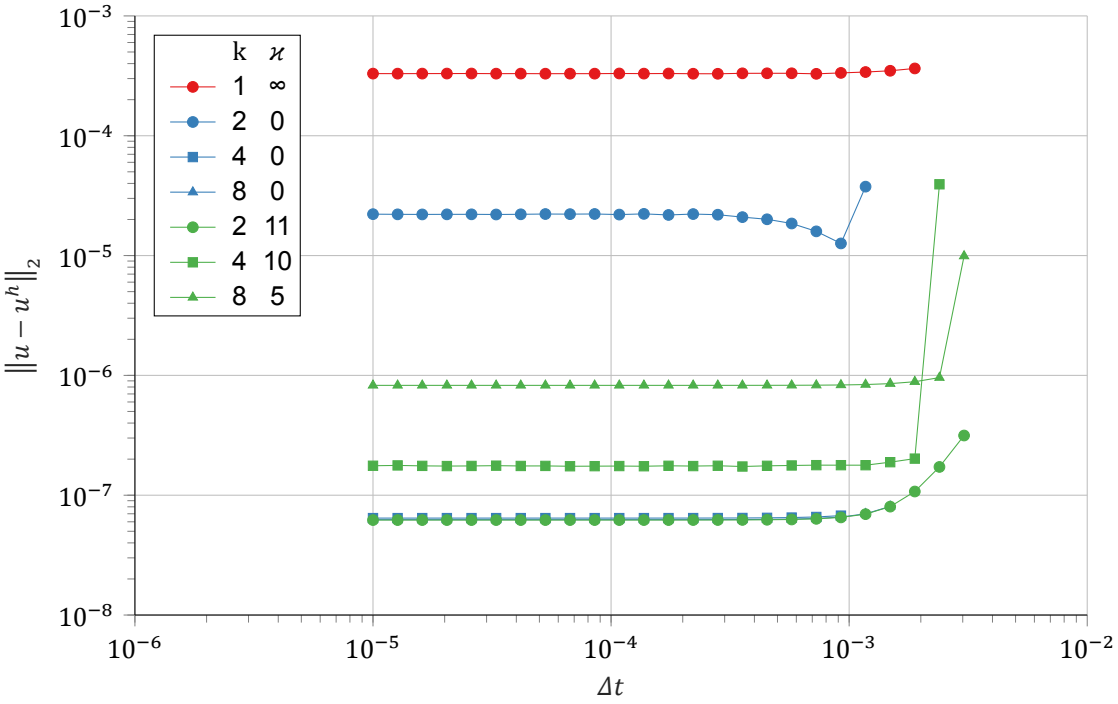


Figure 10.11: DGIGA, $p \gg 1$, time-step size independence (table 10.4, batches 39–45).

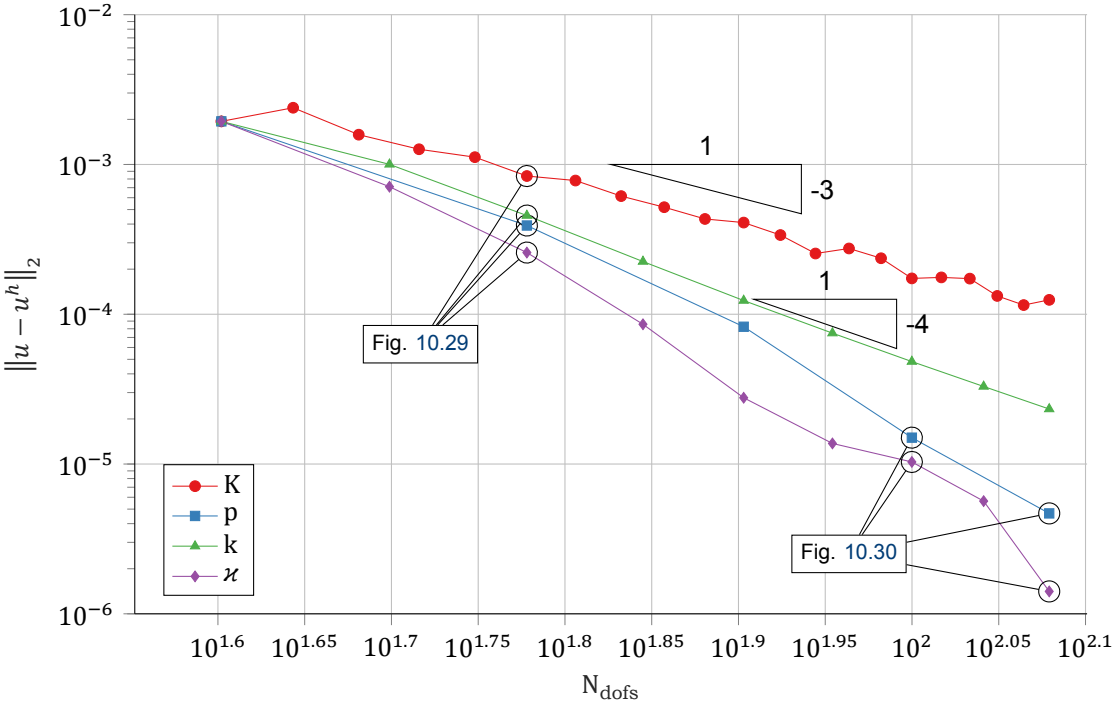


Figure 10.12: DGIGA's four refinement directions (table 10.4, batches 46–49).

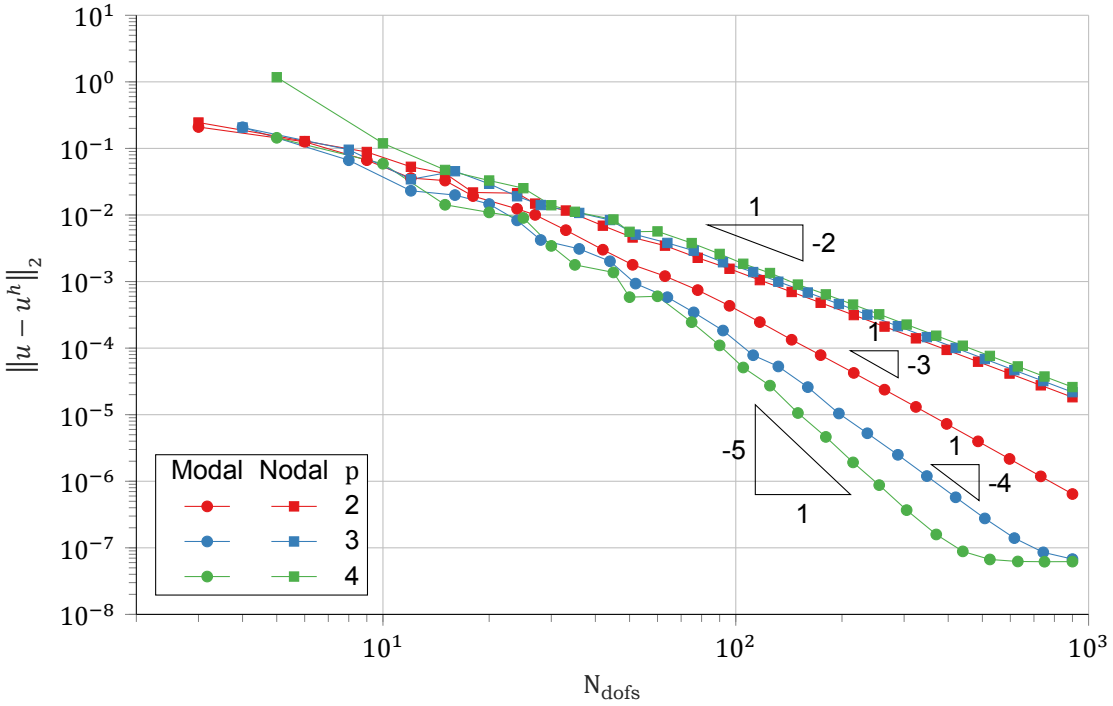


Figure 10.13: DGIGA, $k = 1$, Δx refinement (table 10.4, batches 50–55).

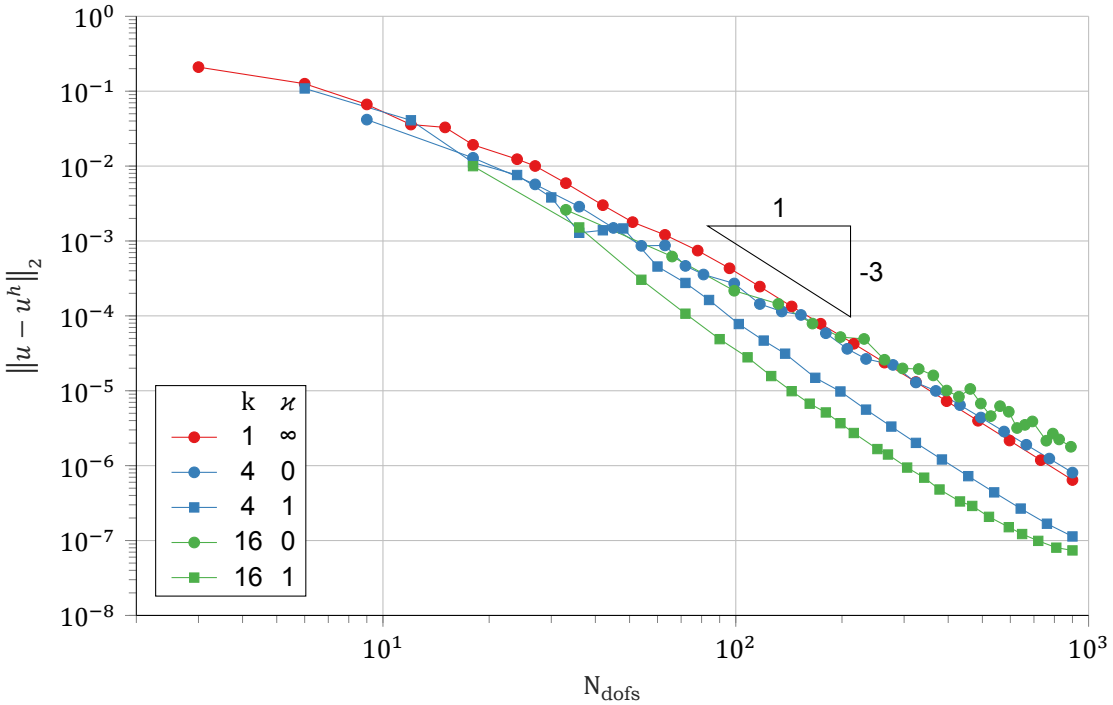


Figure 10.14: DGIGA, $p = 2$, Δx refinement (table 10.4, batches 56–60).

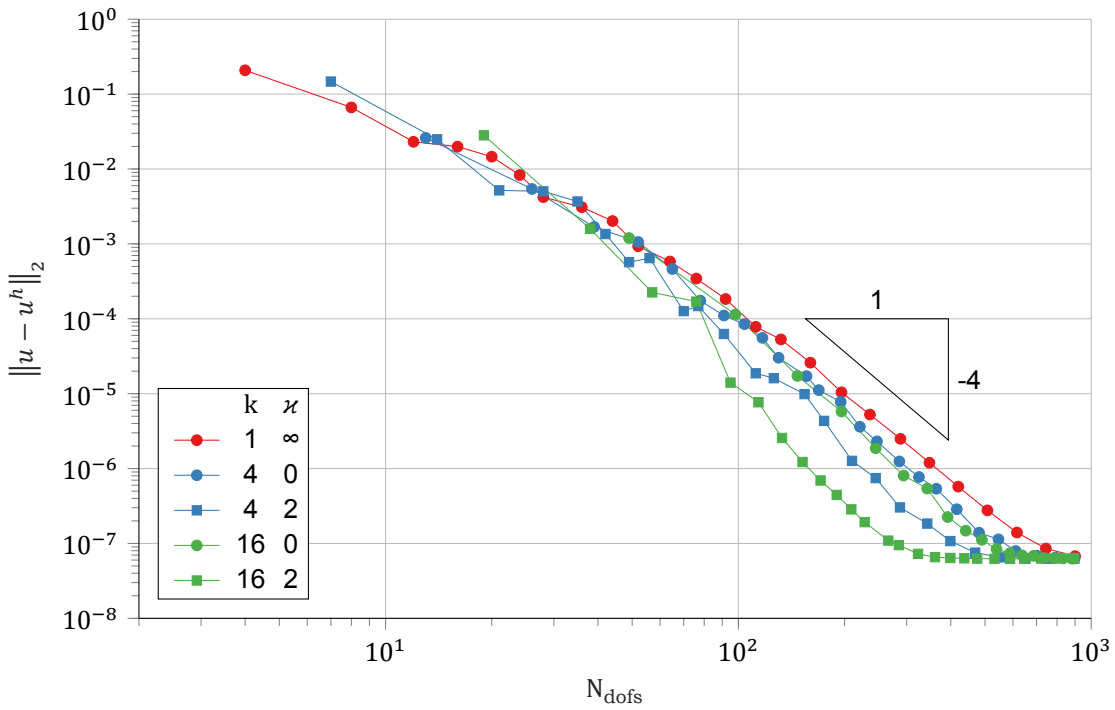


Figure 10.15: DGIGA, p = 3, Δx refinement (table 10.4, batches 61–65).

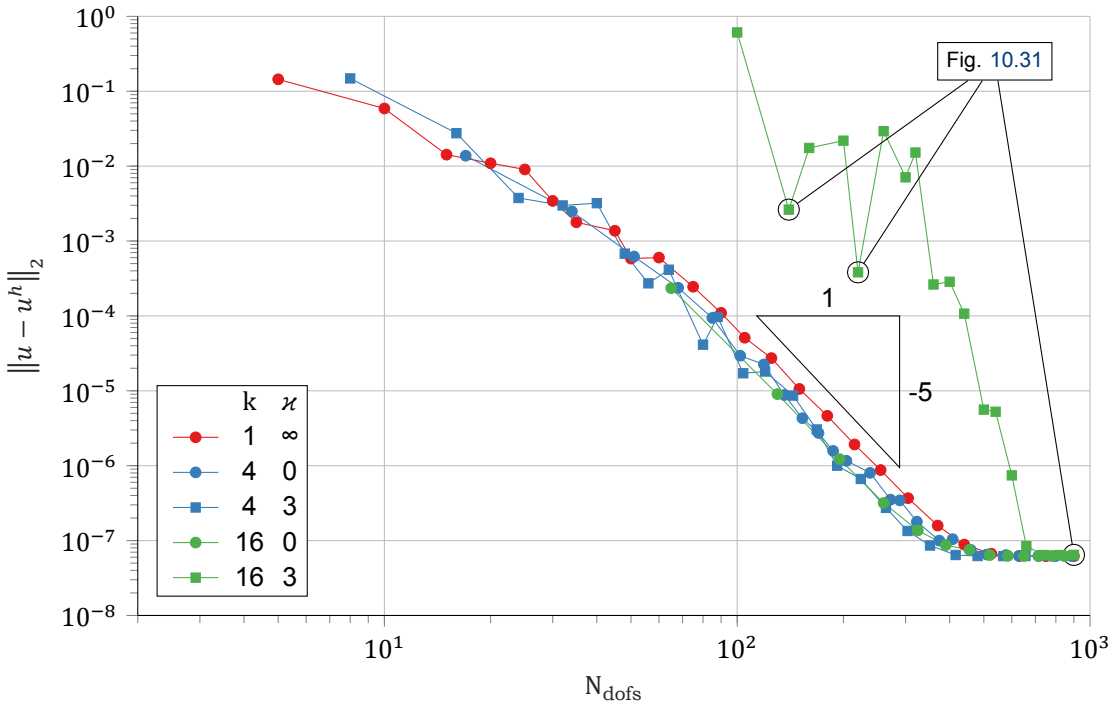


Figure 10.16: DGIGA, p = 4, Δx refinement (table 10.4, batches 66–70).

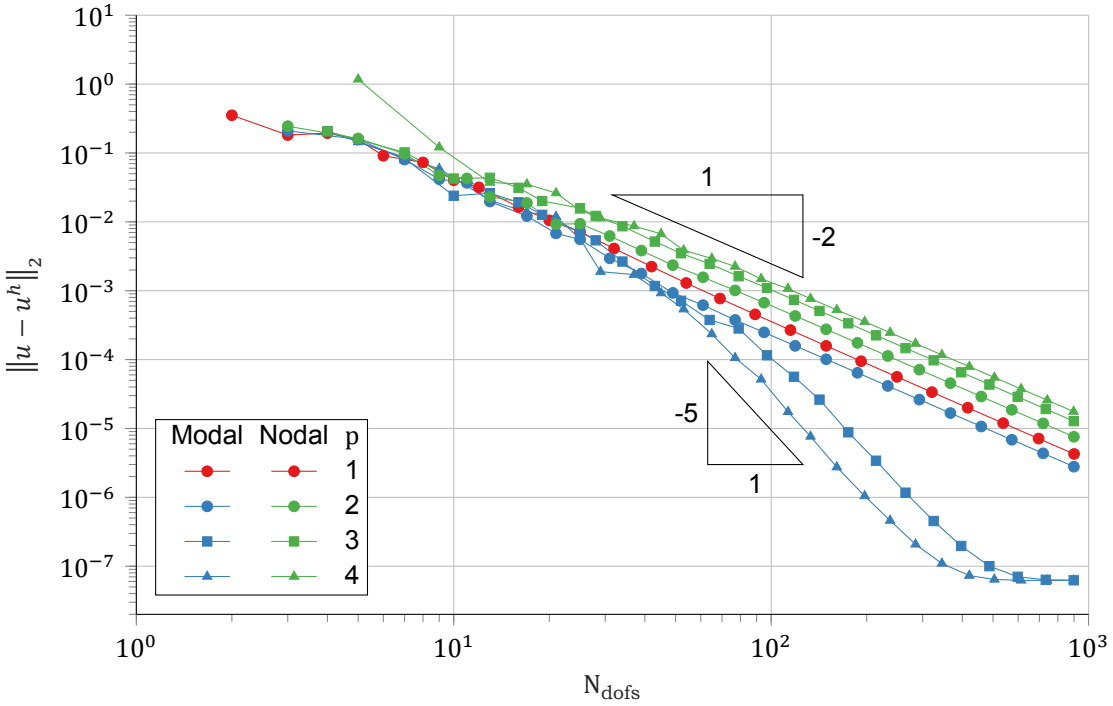


Figure 10.17: CG, k refinement (table 10.4, batches 71–77).

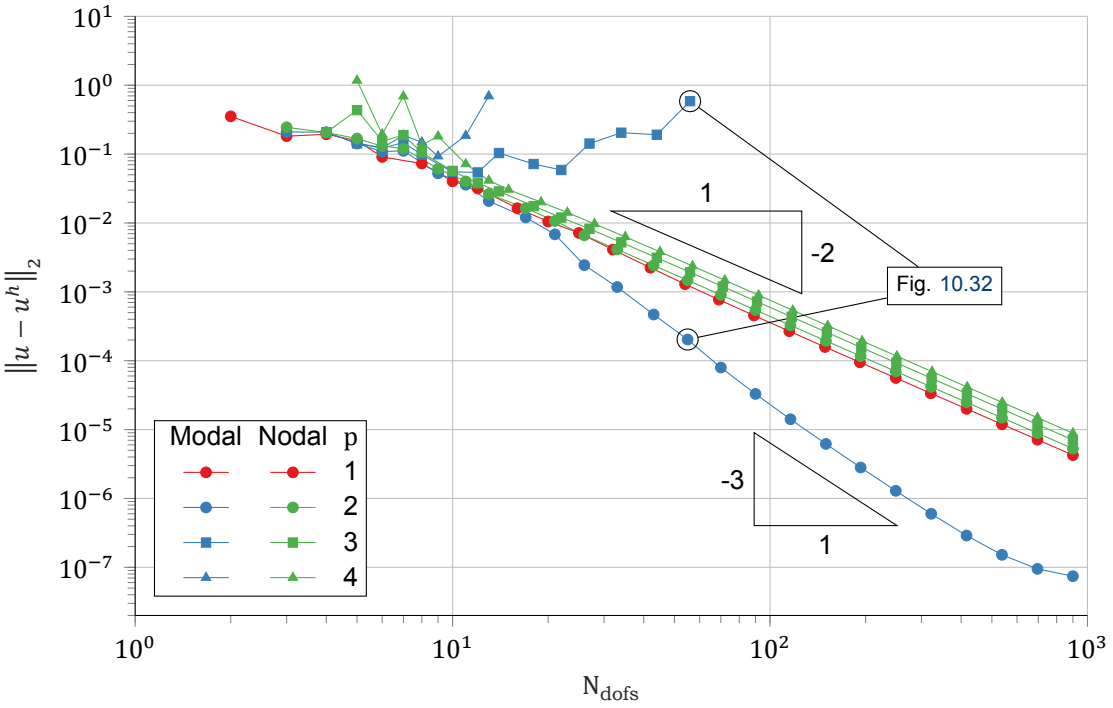


Figure 10.18: IGA, k refinement (table 10.4, batches 78–84).

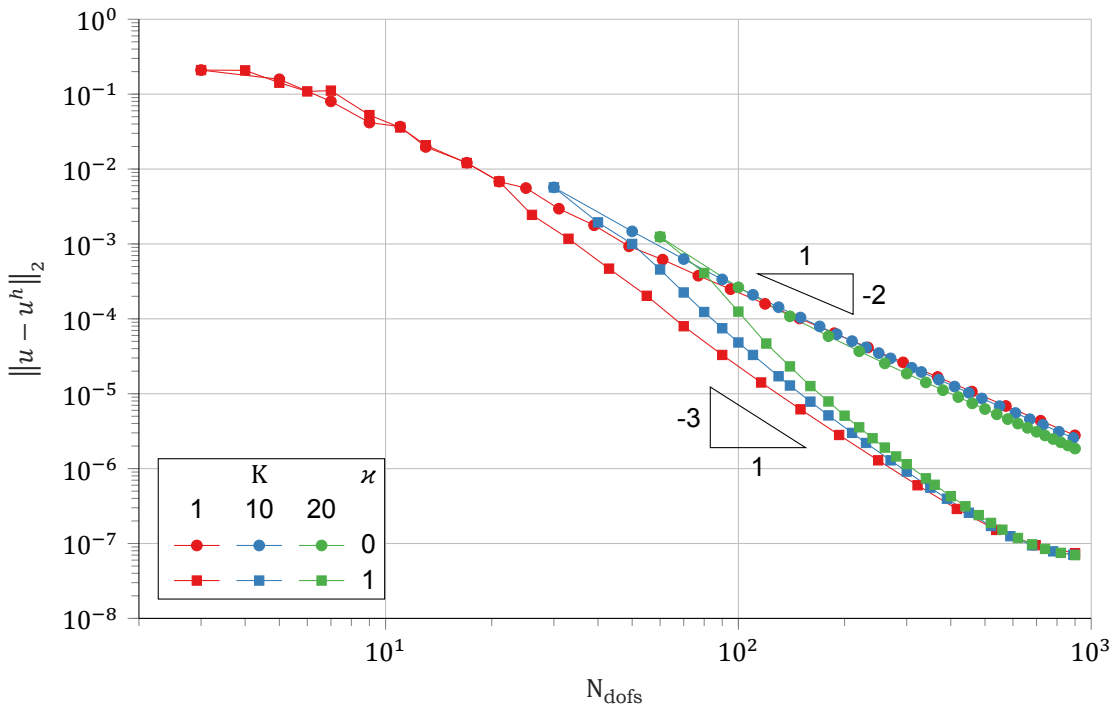


Figure 10.19: DGIGA, p = 2, k refinement (table 10.4, batches 85–90).

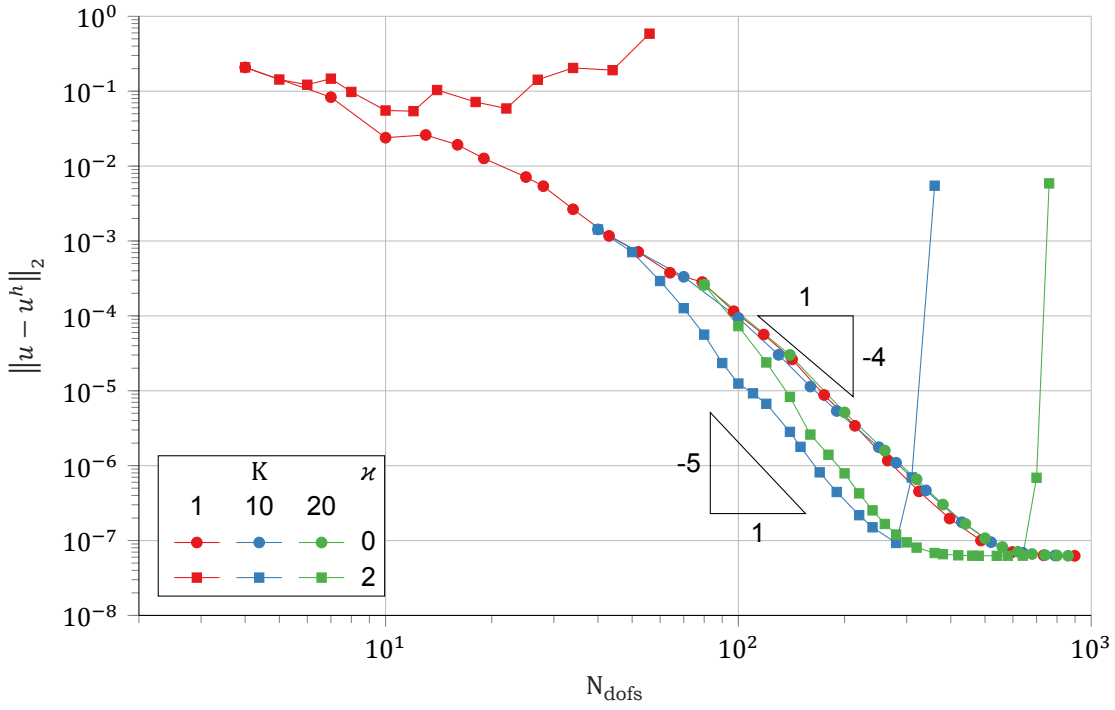


Figure 10.20: DGIGA, p = 3, k refinement (table 10.4, batches 91–96).

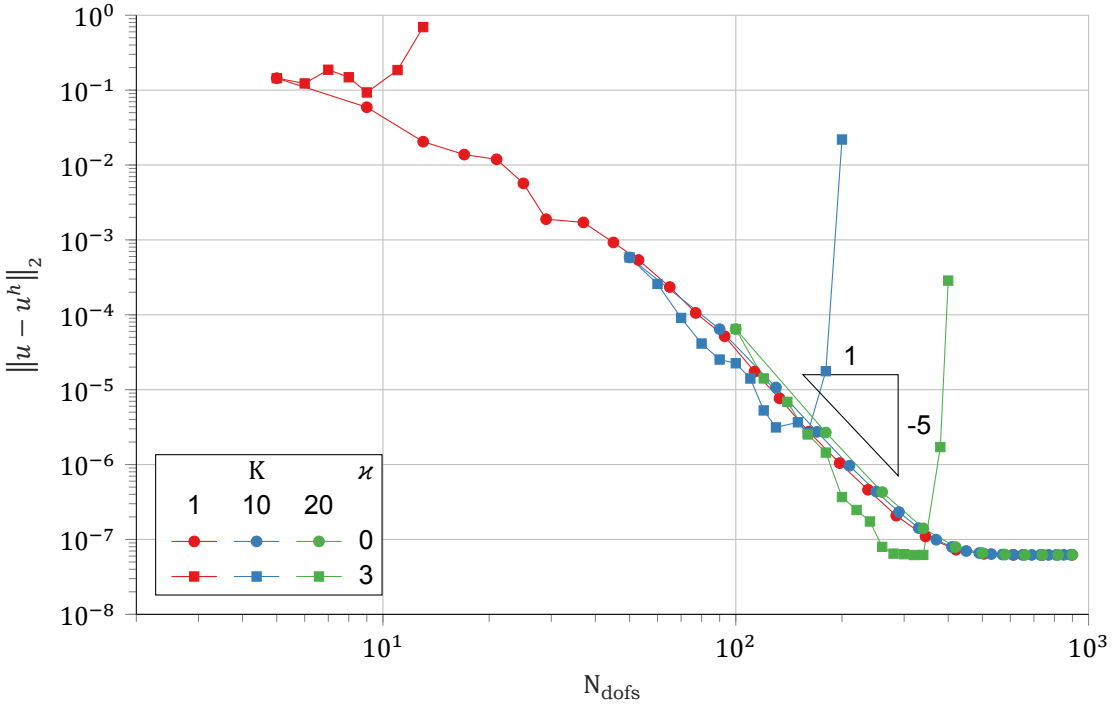


Figure 10.21: DGIGA, $p = 4$, k refinement (table 10.4, batches 97–102).

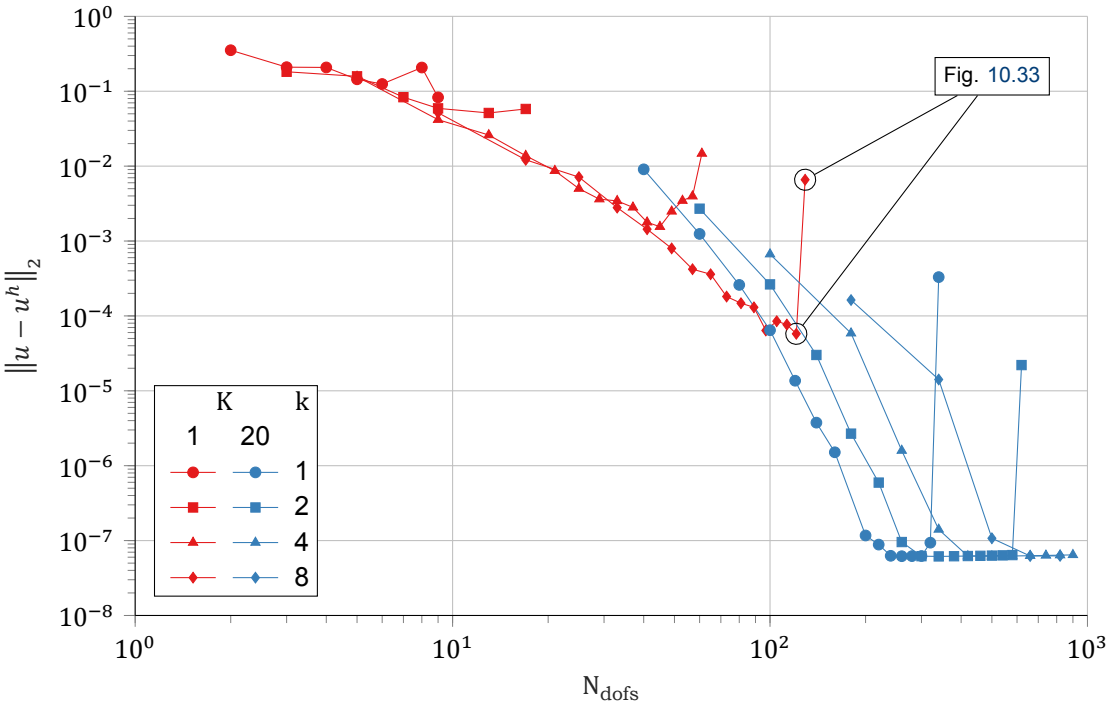


Figure 10.22: DGIGA, p refinement (table 10.4, batches 103–110).

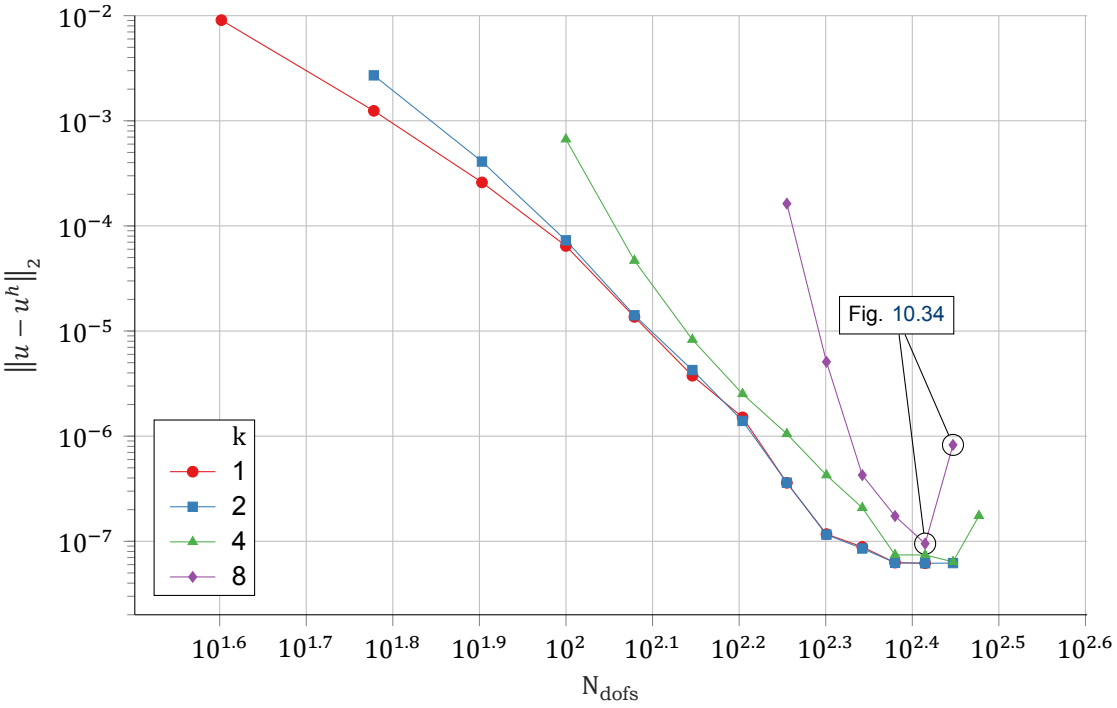


Figure 10.23: DGIGA, combined p and $\varkappa$ refinement (table 10.4, batches 111–114).

DGSEM	Δt	$\|u\|_{\mathrm{TV}}$	$\|u\|_2$	$\|u - u^h\|_2$
	0.000 106	1.998 181	0.485 492	$4.624\,708 \times 10^{-7}$
	0.015 118	1.998 089	0.485 490	$3.321\,760 \times 10^{-5}$
	0.019 145	2.941 220	0.485 381	0.011 802

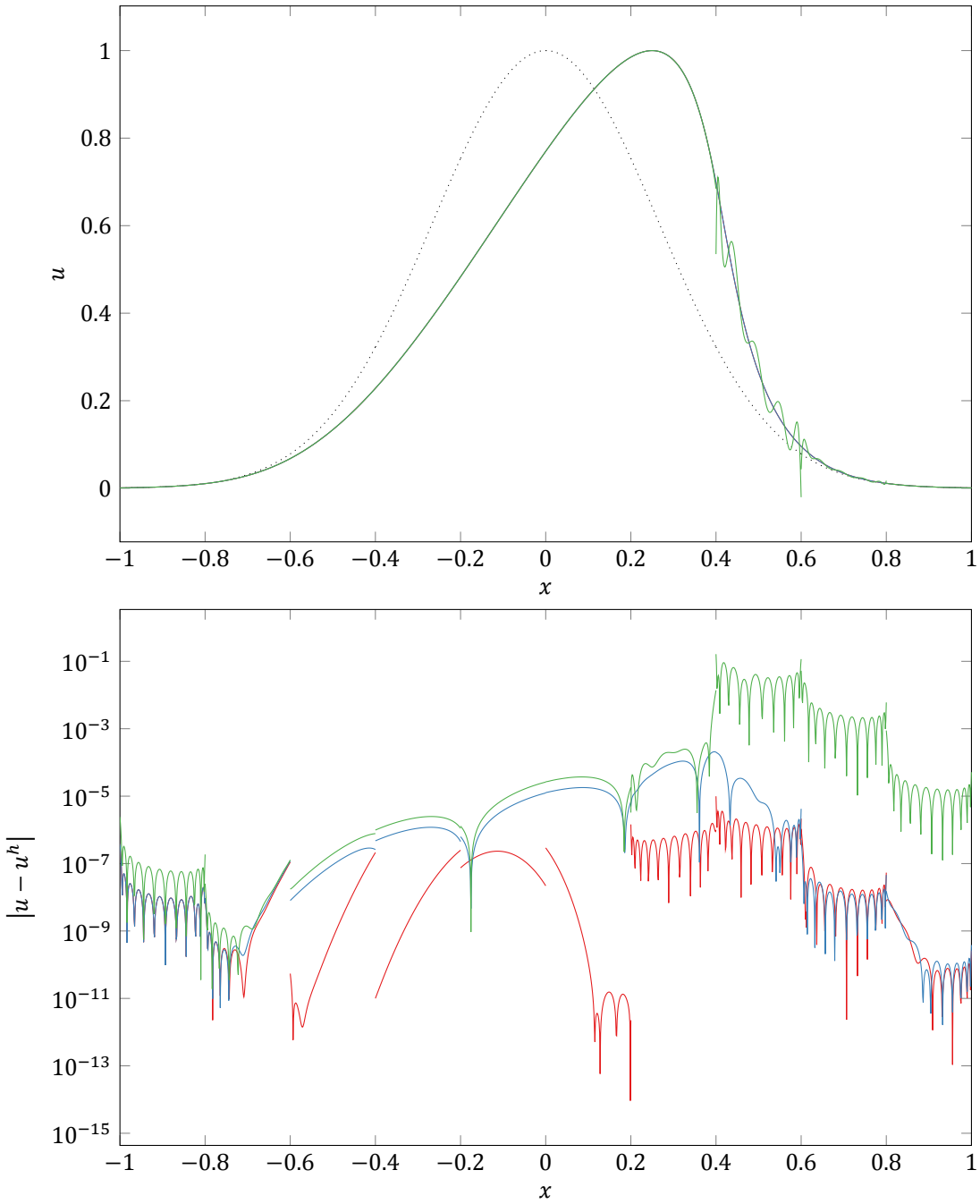


Figure 10.24: DGSEM, selected runs of figure 10.2 (batch 5).

DGSEM	N_{dofs}	$\|u\|_{TV}$	$\|u\|_2$	$\|u - u^h\|_2$
	105	1.998 172	0.485 492	$2.496\,494 \times 10^{-5}$
	125	1.998 197	0.485 492	$2.080\,428 \times 10^{-5}$
	150	1.998 198	0.485 492	$5.639\,772 \times 10^{-6}$

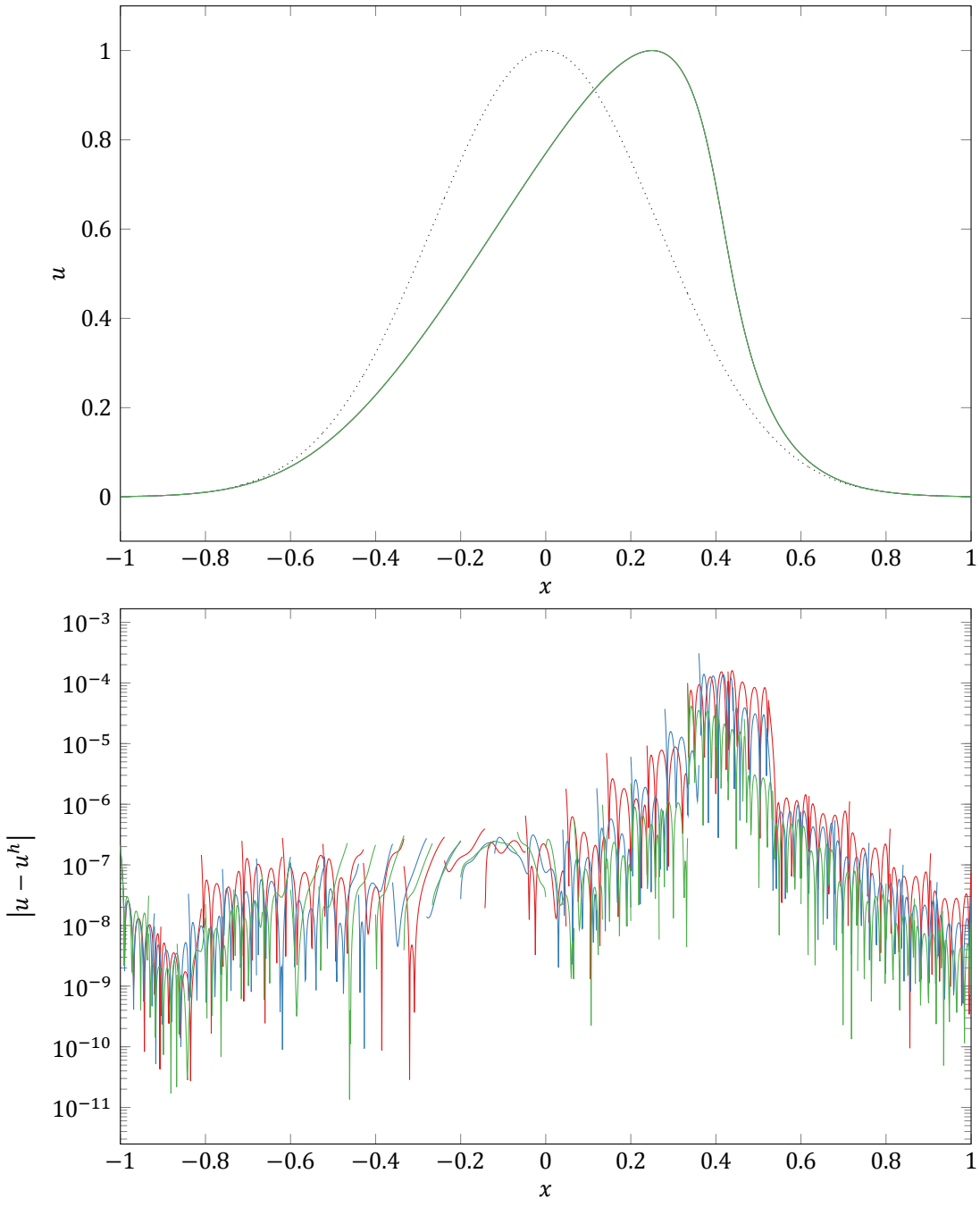


Figure 10.25: DGSEM, selected runs of figure 10.3 (batch 9).

DGSEM	K	N_{dofs}	$\|u\|_{TV}$	$\|u\|_2$	$\|u - u^h\|_2$
	20	20	1.754 952	0.465 025	0.065 691
	10	20	2.376 729	0.483 797	0.023 844
	1	20	2.194 507	0.485 560	0.010 962
	20	120	1.998 213	0.485 492	$7.693\,622 \times 10^{-6}$
	10	120	1.998 181	0.485 492	$4.624\,707 \times 10^{-7}$
	1	120	1.997 965	0.485 492	$9.896\,914 \times 10^{-7}$

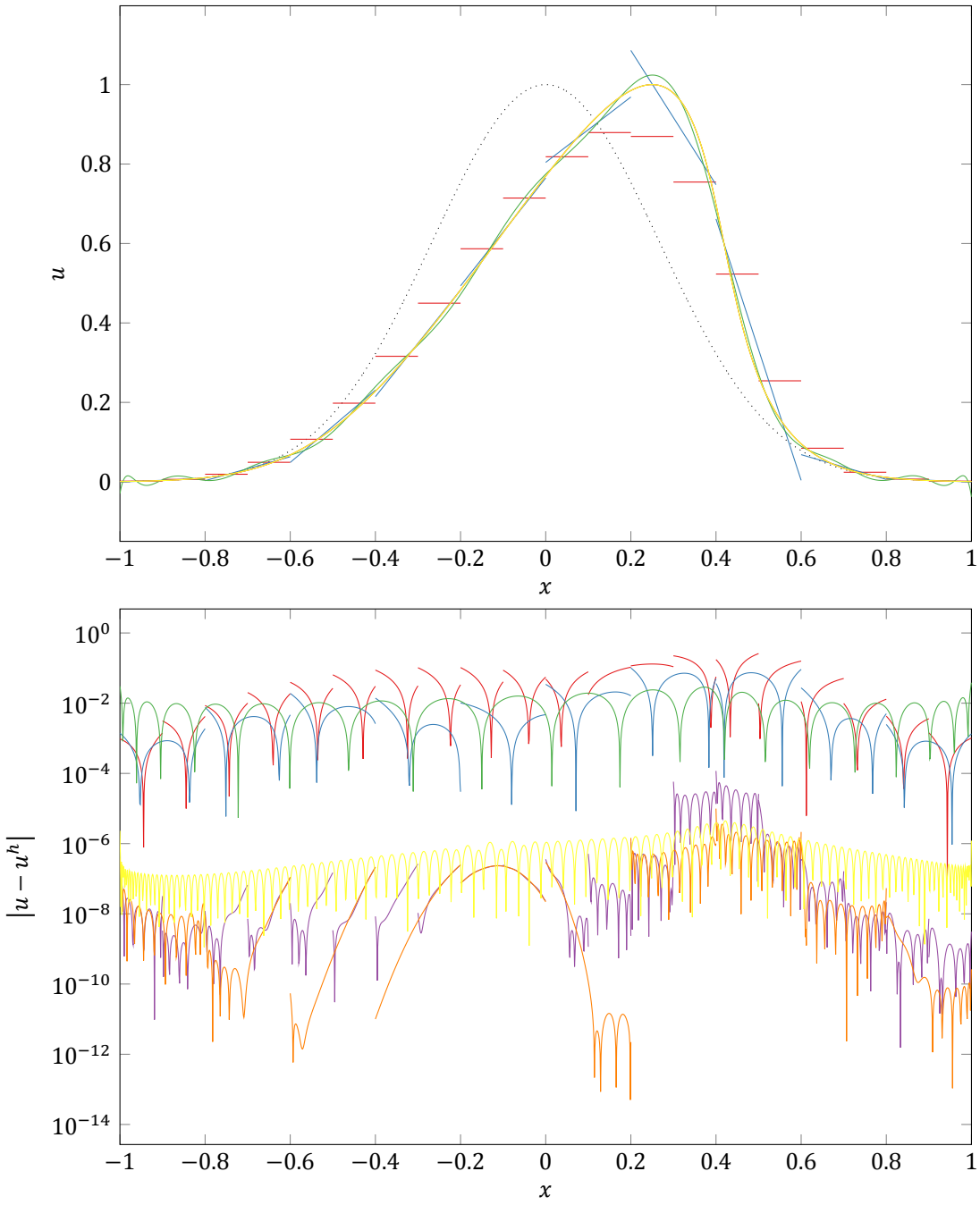


Figure 10.26: DGSEM, selected runs of figure 10.4 (batches 10–12).

FR/CPR	η	$\|u\|_{\mathrm{TV}}$	$\|u\|_2$	$\left\|u-u^h\right\|_2$
	$\eta_{-/2}$	1.998 279	0.485 492	$4.386\,177 \times 10^{-7}$
	η_{Ga}	1.998 279	0.485 492	$7.185\,730 \times 10^{-7}$
	η_2	1.998 280	0.485 492	$1.248\,549 \times 10^{-6}$
	η_∞	2.019 247	0.485 492	$3.581\,458 \times 10^{-5}$

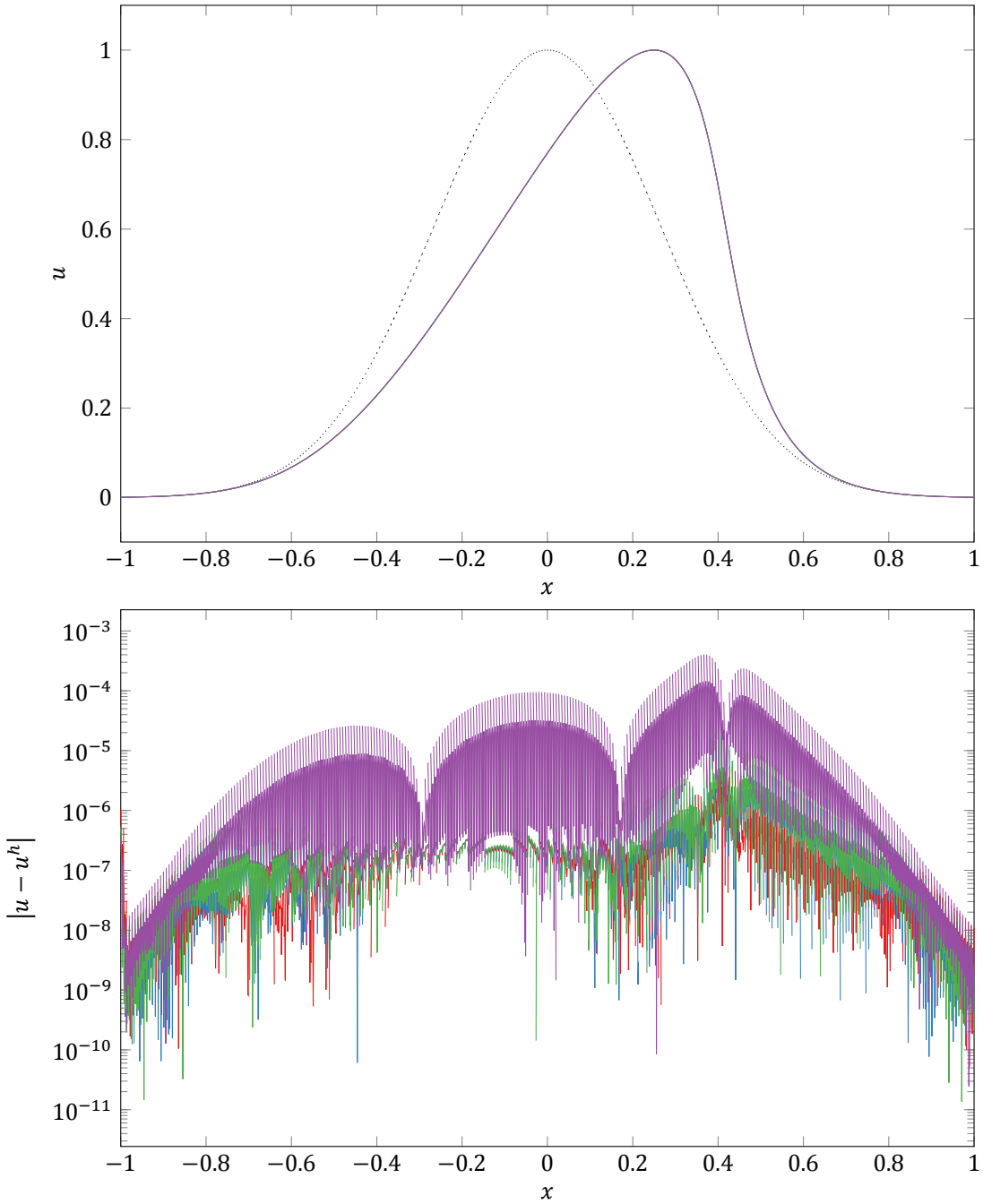


Figure 10.27: FR/CPR, selected runs of figure 10.6 (batches 9–12).

FR/CPR	N_{dofs}	$\|u\|_{TV}$	$\|u\|_2$	$\|u - u^h\|_2$
	370	1.998 262	0.485 492	$1.238\,026 \times 10^{-7}$
	440	1.998 271	0.485 492	$7.715\,097 \times 10^{-8}$
	525	1.998 275	0.485 492	$6.537\,474 \times 10^{-8}$
	630	1.998 277	0.485 492	$6.334\,775 \times 10^{-8}$

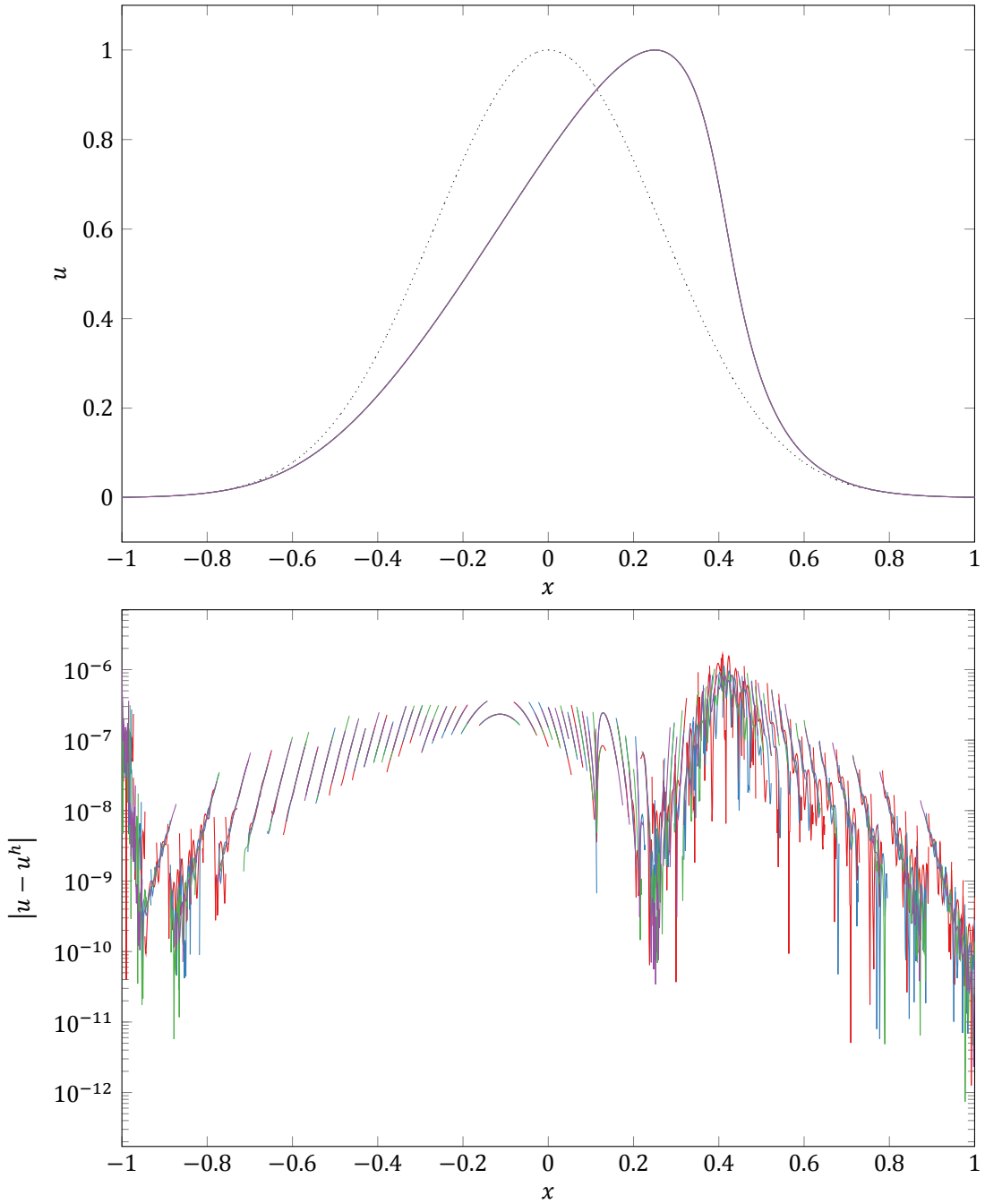


Figure 10.28: FR/CPR, selected runs of figure 10.6 (batch 17).

DGIGA	$\|u\|_{\mathrm{TV}}$	$\|u\|_2$	$\|u - u^h\|_2$
	1.997 998	0.485 491	0.000 837
	1.998 411	0.485 492	0.000 391
	1.997 948	0.485 482	0.000 456
	1.997 681	0.485 496	0.000 258

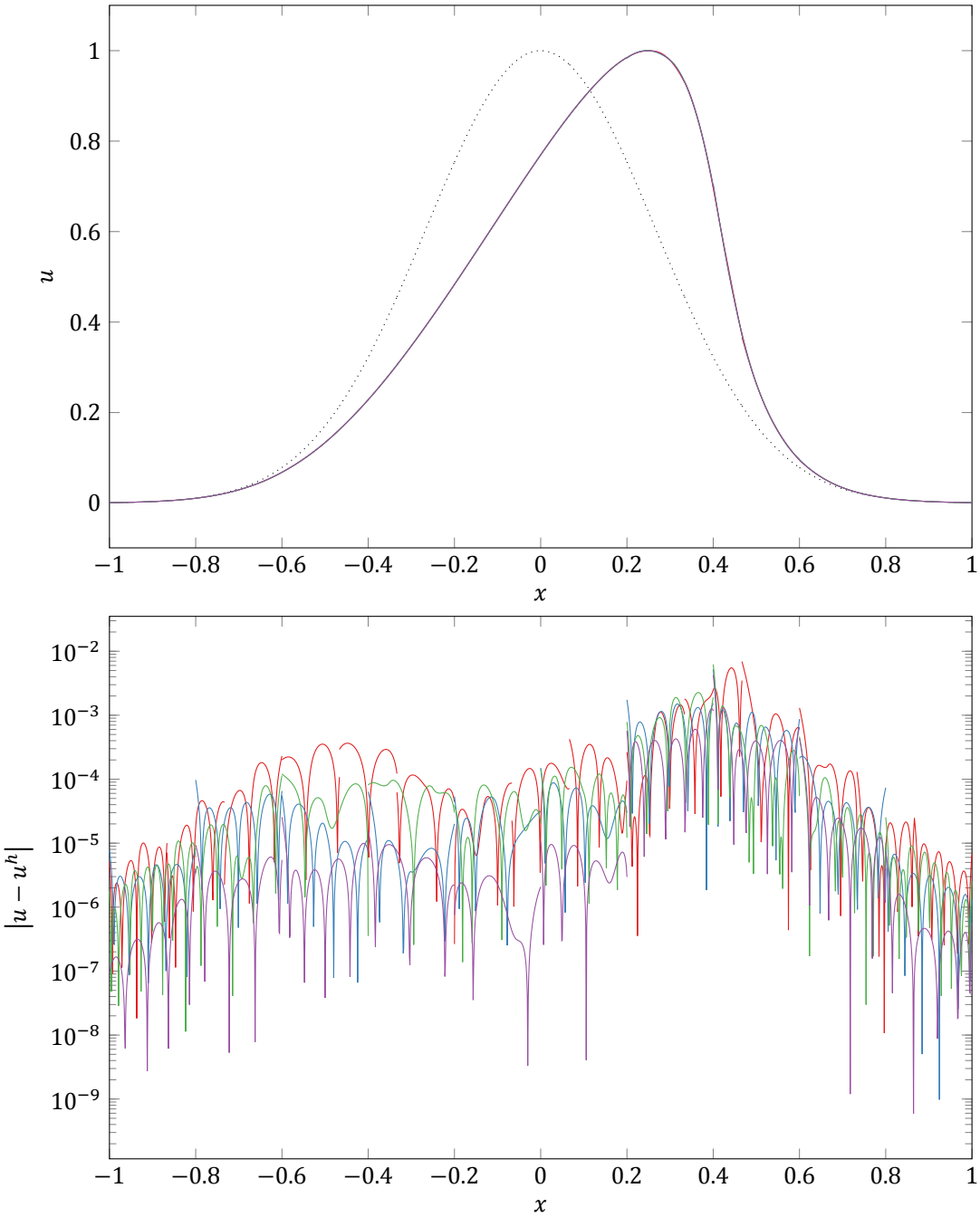


Figure 10.29: DGIGA, selected runs of figure 10.12 (batches 46–49).

DGIGA	p	$\varkappa$	N_{dofs}	$\|u\|_{TV}$	$\|u\|_2$	$\|u - u^h\|_2$
	5	1	100	1.998 235	0.485 492	$1.500\,287 \times 10^{-5}$
	6	1	120	1.998 218	0.485 492	$4.681\,527 \times 10^{-6}$
	8	7	100	1.998 257	0.485 492	$1.032\,280 \times 10^{-5}$
	10	9	120	1.998 272	0.485 492	$1.411\,021 \times 10^{-6}$

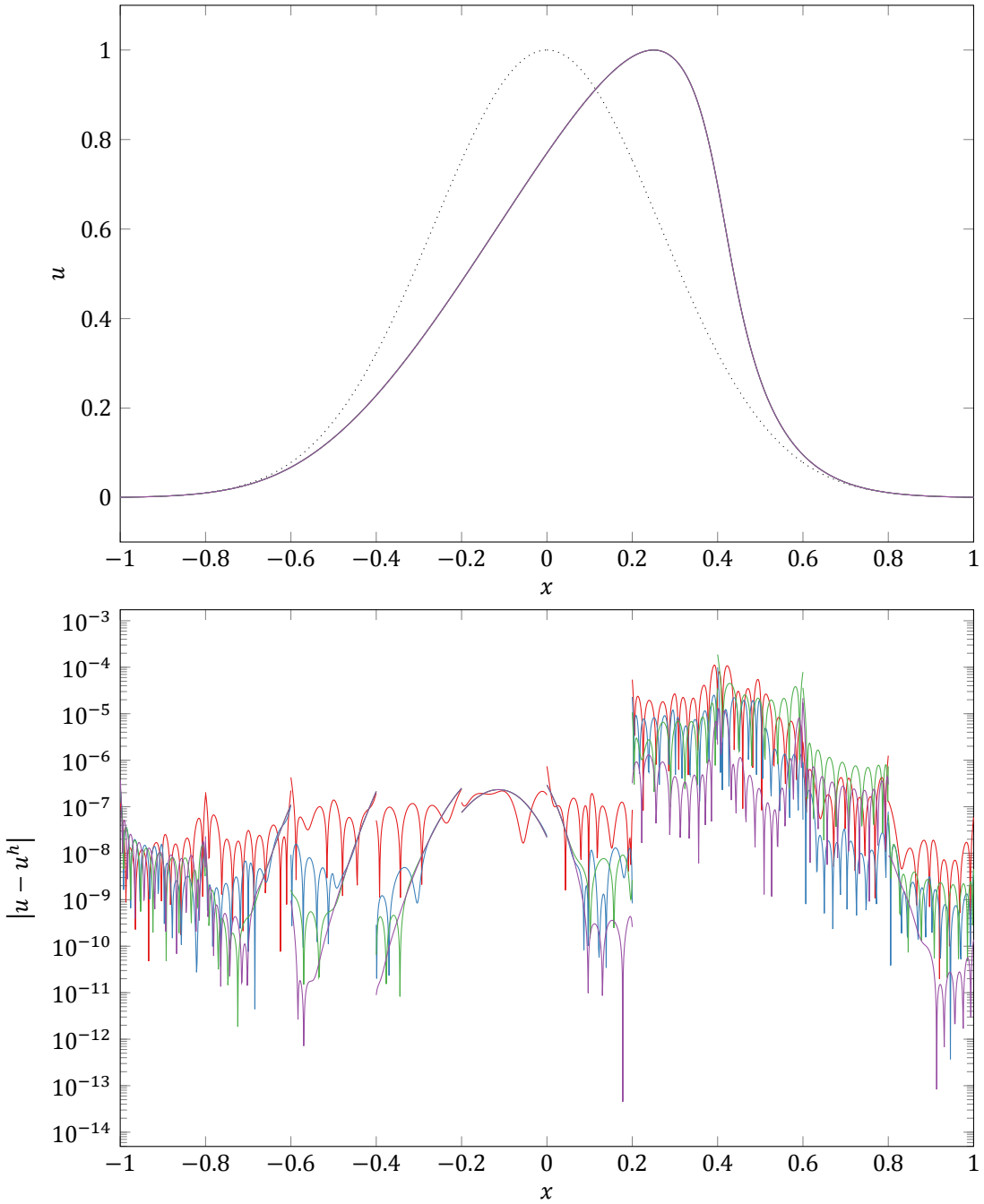


Figure 10.30: DGIGA, selected runs of figure 10.12 (batches 47 and 49).

DGIGA	N_{dofs}	$\|u\|_{TV}$	$\|u\|_2$	$\|u - u^h\|_2$
	140	2.019 155	0.485 492	0.002 626
	220	2.012 106	0.485 492	0.000 382
	900	1.998 292	0.485 492	$6.441\,432 \times 10^{-8}$

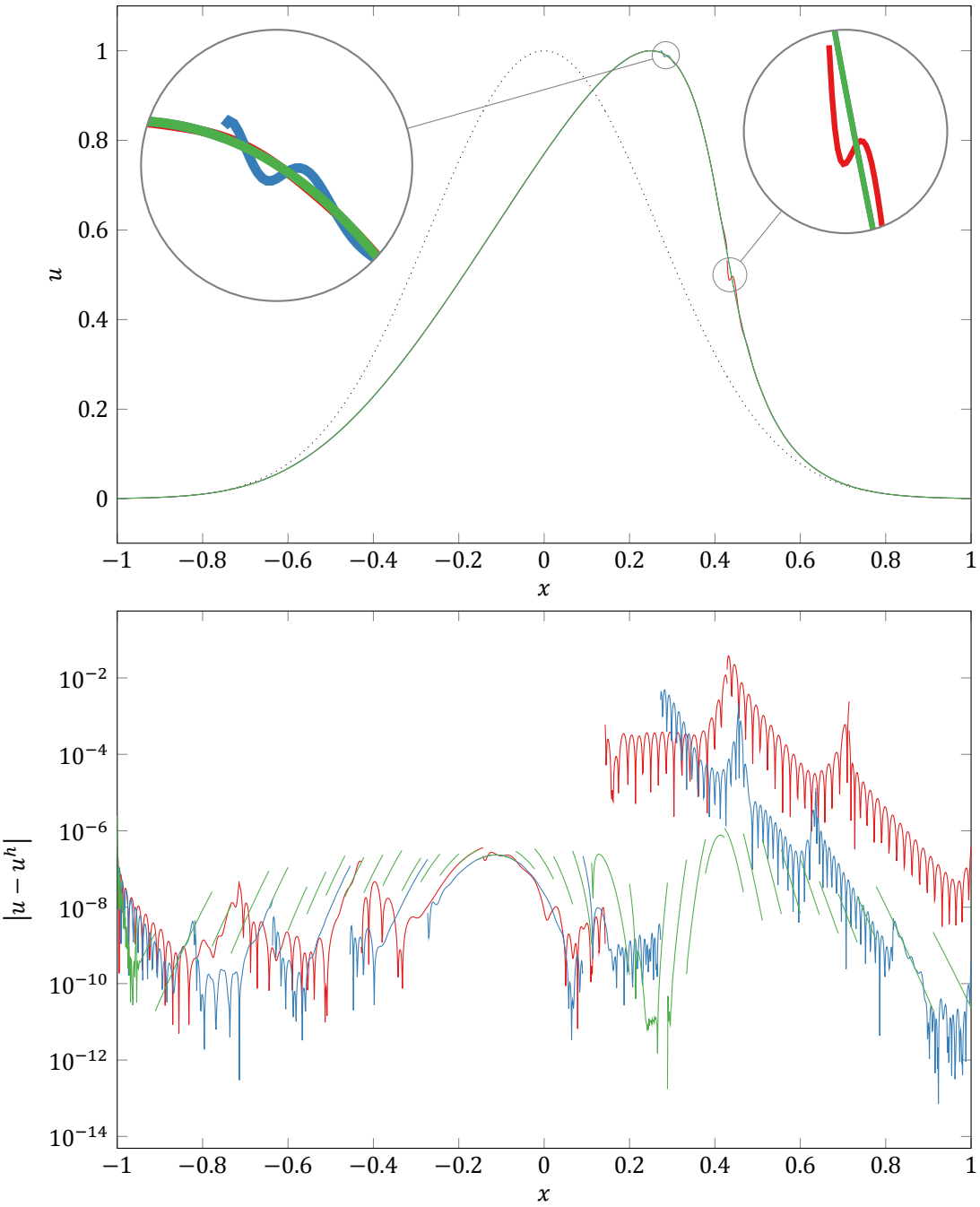


Figure 10.31: DGIGA, selected runs of figure 10.16 (batch 70).

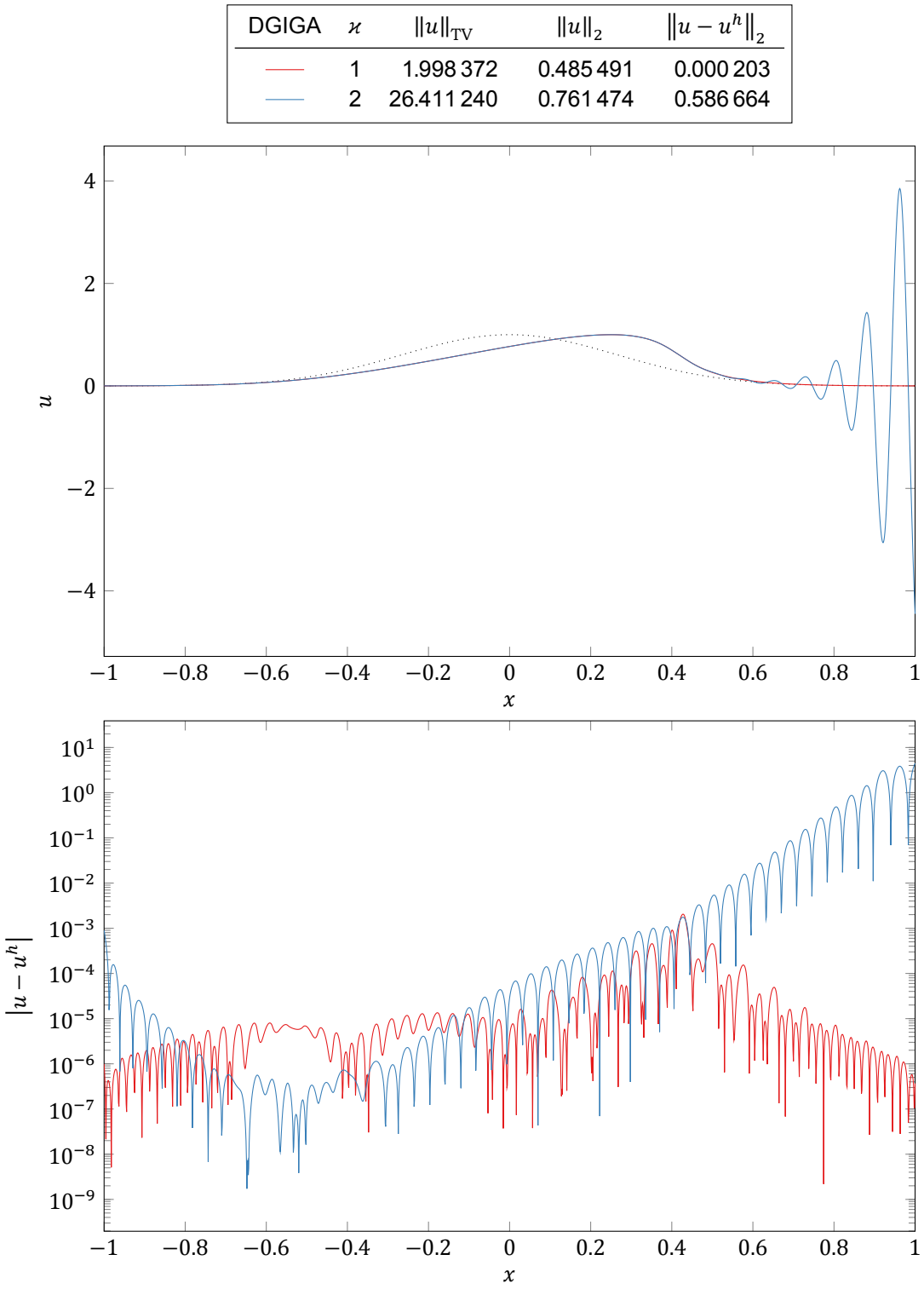


Figure 10.32: DGIGA, selected runs of figure 10.18 (batches 79 and 80).

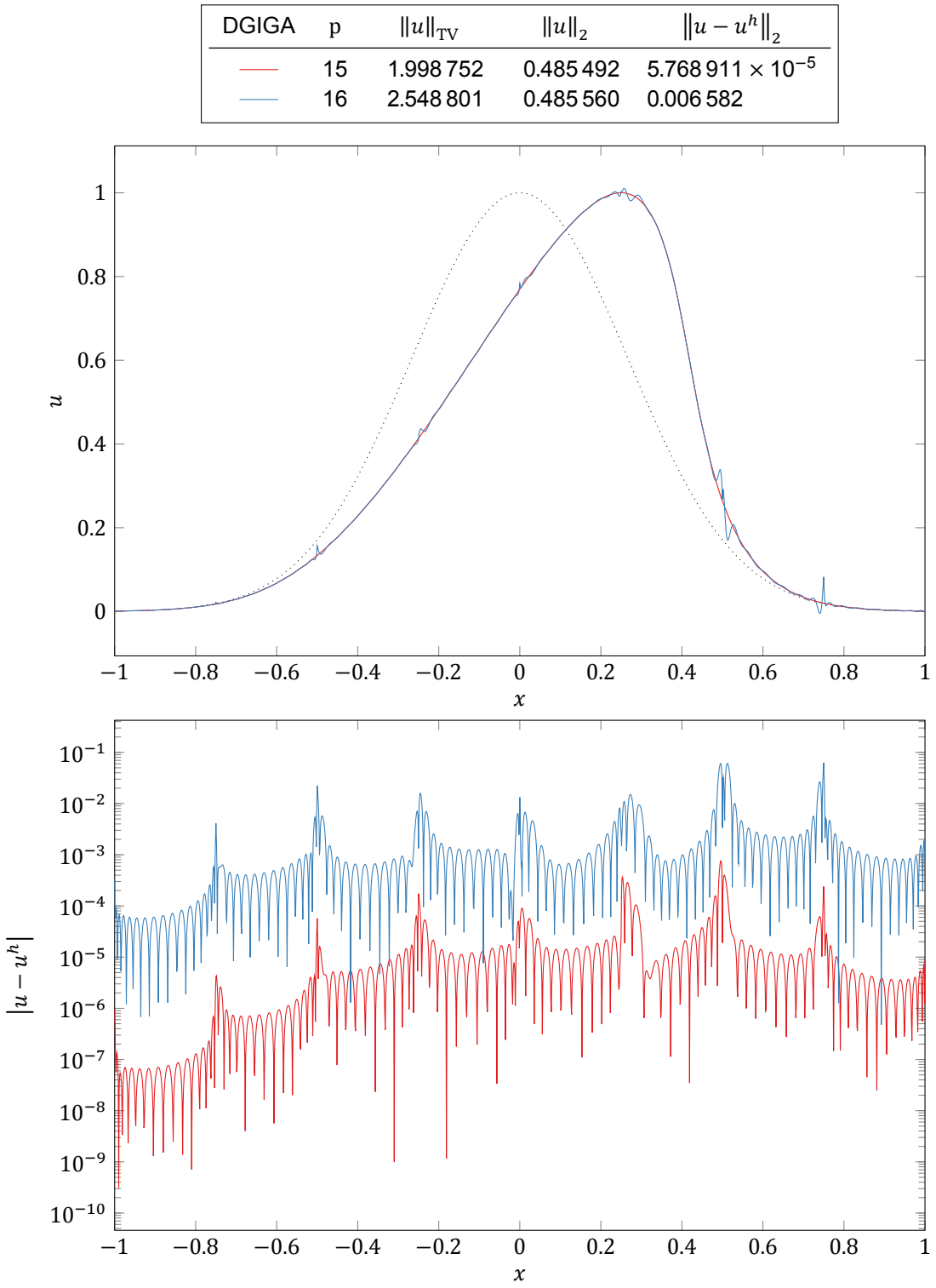


Figure 10.33: DGIGA, selected runs of figure 10.22 (batch 106).

DGIGA	p	$\|u\|_{\mathrm{TV}}$	$\|u\|_2$	$\|u-u^h\|_2$
	5	1.998 286	0.485 492	$9.499\,574 \times 10^{-8}$
	6	1.998 289	0.485 492	$8.255\,893 \times 10^{-7}$

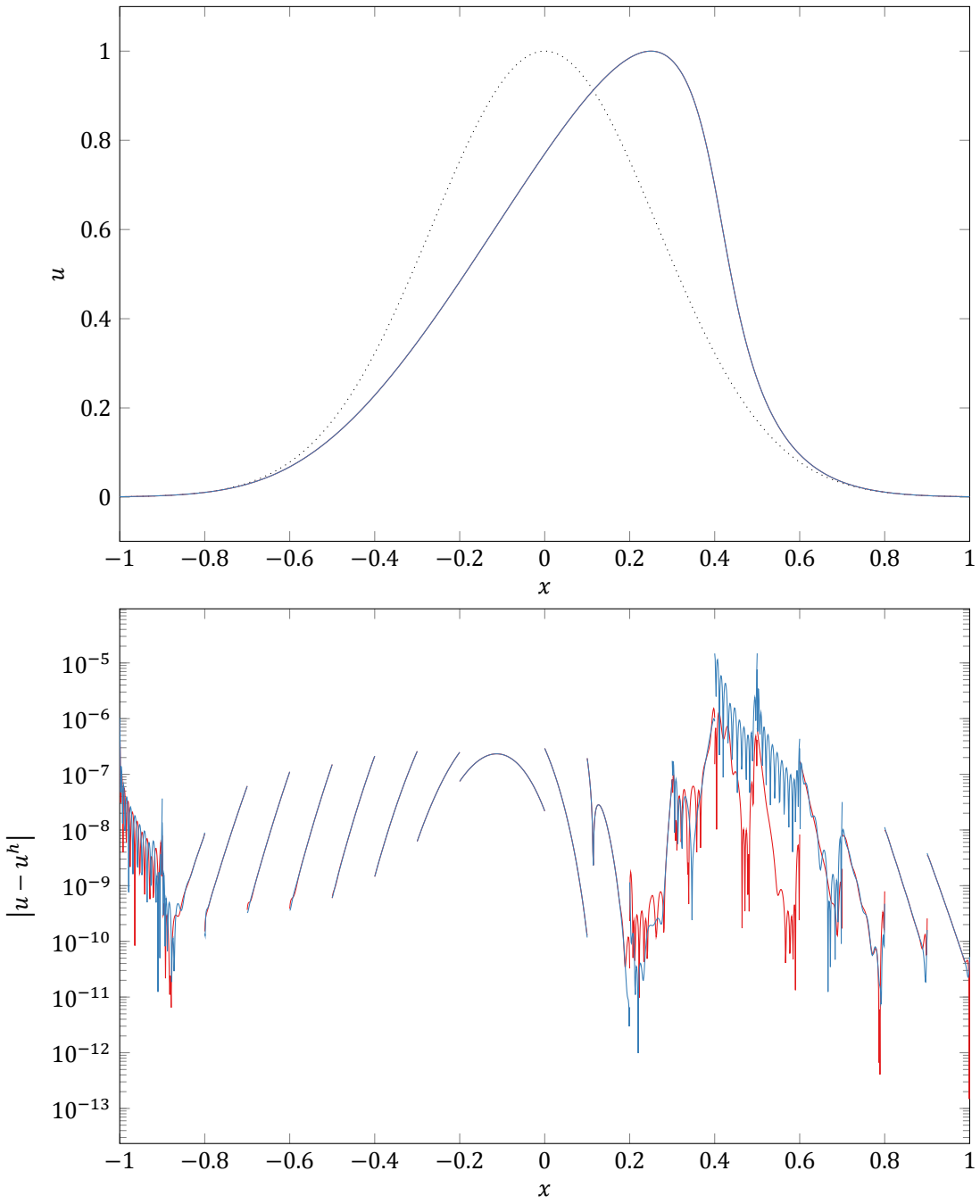


Figure 10.34: DGIGA, selected runs of figure 10.23 (batch 114).

11

Dispersion, Dissipation and Linear Stability

Equation (2.19), the linear scalar hyperbolic conservation law, allows the characterization of *dissipative*, *dispersive* and *linear stability* properties of a given (linear) spatial discretization method in an analytical, a priori, approach (all details are given in appendix A). The main motivation behind these results is to identify regions in the parameter space of each method under study which result in promising discretizations for a scale-resolving context (to be placed under further scrutiny in subsequent chapters).

In all figures, the exact dispersion/dissipation relation (§A.2.1) is shown as a thick, solid line. Also, in the complex plane graphs, the boundary of the stability region of SSP-RK3(3) is shown in black. All Fourier footprints shown are scaled by the largest linearly stable[1] Courant number; these are listed in each figure, for every basis analyzed.

11.1. DGSEM

A priori modified wavenumber results are for $\Delta x \to 0$. DGSEM discretizations, therefore, have only one free parameter: the polynomial degree, p. For this method, spurious eigenmodes seem to always retain exactly the same shape as the physical one—their footprints in the complex plane coincide, see e.g. figures 11.1, 11.2 and 11.3. Focusing only on the physical eigenmode, figures 11.4 and 11.5 show a clear trend: the higher the degree, the more wavenumbers *per basis function* that are *accurately* resolved. In all cases, there is an overshoot in the modified dispersion relation. It becomes stronger as p increases. This occurs at the same wavenumbers for which dissipation rate increases rather sharply; this onset of numerical dissipation becomes both stronger and more sudden with increasing p. These results match those found in the literature, e.g. [116, §8.1.5] and [38].

11.2. FR/CPR

Flux reconstruction adds a new parameter, the correction function. Just like in DGSEM, the spurious eigenmodes of all[2] FR/CPR discretizations appear to correspond to phase shifts applied to the physical one—evidence of that is provided in figures 11.6–11.15. Figures 11.16–11.20 focus on the behavior of the physical eigenmode as p increases, for each correction function separately. Reciprocally, various correction functions are compared to each other at a fixed degree in figures 11.21–11.25. These figures show that the overshoot in the dispersion relation can be mitigated by using a stronger correction, for moderate degrees—this effect is very modest for high degrees, and increasing η to arbitrarily high values is not enough to reduce said overshoot further; also, this simultaneously pulls the onset of non-negligible numerical dissipation towards lower wavenumbers.

Vincent et al. [120] carried out a similar analysis, the results of which are consistent with those presented here. They performed a very detailed study on the optimal choice of the c or η parameters, and concluded that:

[1] See §A.4.

[2] Except for η_∞, for which one of the eigenmodes is constant and equal to zero—see 11.10 and 11.15.

- Values close to the extrema (η_- and η_∞), at high wavenumbers, admit large dispersion errors with little damping and, therefore, should be avoided.
- It is possible to maximize the range of wavenumbers over which the exact dispersion relation is approximated accurately, by using $\eta > 0$ (e.g. η_{Ga} is optimal in this sense for $\mathrm{p} = 3$).
- A second local optimum is in global order of accuracy per degree of freedom, and is achieved by η_{DG} (a justification can be found in [120]).
- A third local optimum exists in the form of the η that maximizes the maximum allowable time-step size for a fixed degree (refer to [120] for details).

11.3. DGIGA

A B-spline–based trial space has *three* free parameters able to influence its spectral response: the number of breakpoint spans per patch ($\mathrm{k} > 0$), the degree of the piecewise polynomials within each span ($\mathrm{p} \geq 0$), and the continuity class of its basis functions across spans ($C^\varkappa : 0 \leq \varkappa \leq \mathrm{p} - 1$). A crucial difference that I have observed DGIGA has with DGSEM and (most) FR/CPR cases is the tendency—as the number of basis functions per patch, J, increases—of its Fourier footprint to split into more than one contour (compare figures 11.26 and 11.27). This process, which seems to occur for $\mathrm{k} > 1$ once J (the number of basis functions per patch) is high enough, is reminiscent to how bubbles are made when blowing onto a soap film. It starts with the main (and only, up to that point) contour developing a pair of protrusions (one in each side of the real axis); these elongate such that the region that connects them with the main contour gets thinner, until it eventually closes, resulting in three distinct contours: the main one, traced by the the physical eigenmode and its "resonant eigenmodes", and two isolated "bubbles"—traced, in every single case encountered, by a single (spurious) eigenmode each. This process continues as J keeps increasing, with additional "bubbles" forming on each of the new contours, and so on.

The influence of the degree is shown first, in figures 11.28–11.35. Figures 11.36–11.43 do the same for k. These reveal an interesting aspect of the $\mathrm{k} = 1$ discretizations: their dispersion/dissipation relations and Fourier footprint are identical to those of DGSEM with equal degree[3]. Therefore—in the linear case, at least—this particular version of DGIGA is essentially equivalent to DGSEM[4]. The effect of the smoothness of the basis functions is explored for various break span numbers at $\mathrm{p} = 4$ in figures 11.44–11.46. Lastly, in figure 11.47, various combinations of parameters resulting in $\mathrm{J} = 22$ are compared to each other.

To my knowledge, no extensive exploration of the spectral properties of the DGIGA discretization such as the present one exists in the literature. There is some degree of overlap, however, with [21, figure 22a]. The aforementioned is consistent with my results—specifically, figure 11.42.

[3] Differences start being non-negligible at sufficiently high degree ($\mathrm{p} > 25$, not shown); I hypothesize this to be a numerical artifact, due to the ill-conditioning of such DGIGA bases resulting in large errors in the computation of their modified wavenumbers.

[4] Notice that the finite dimensional space spanned by these basis functions coincides with the trial and test spaces of DGSEM—see §6.2.2.

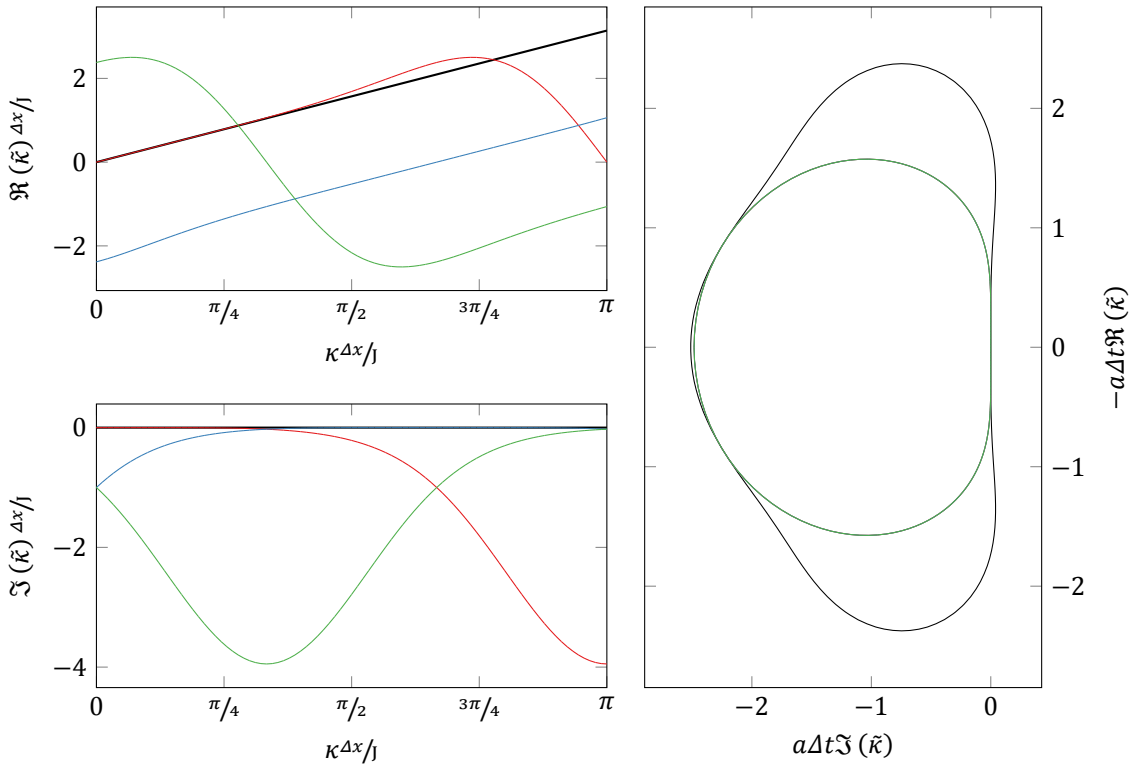


Figure 11.1: Modified wavenumbers (all eigenmodes); DGSEM, $\mathrm{p} = 2$.

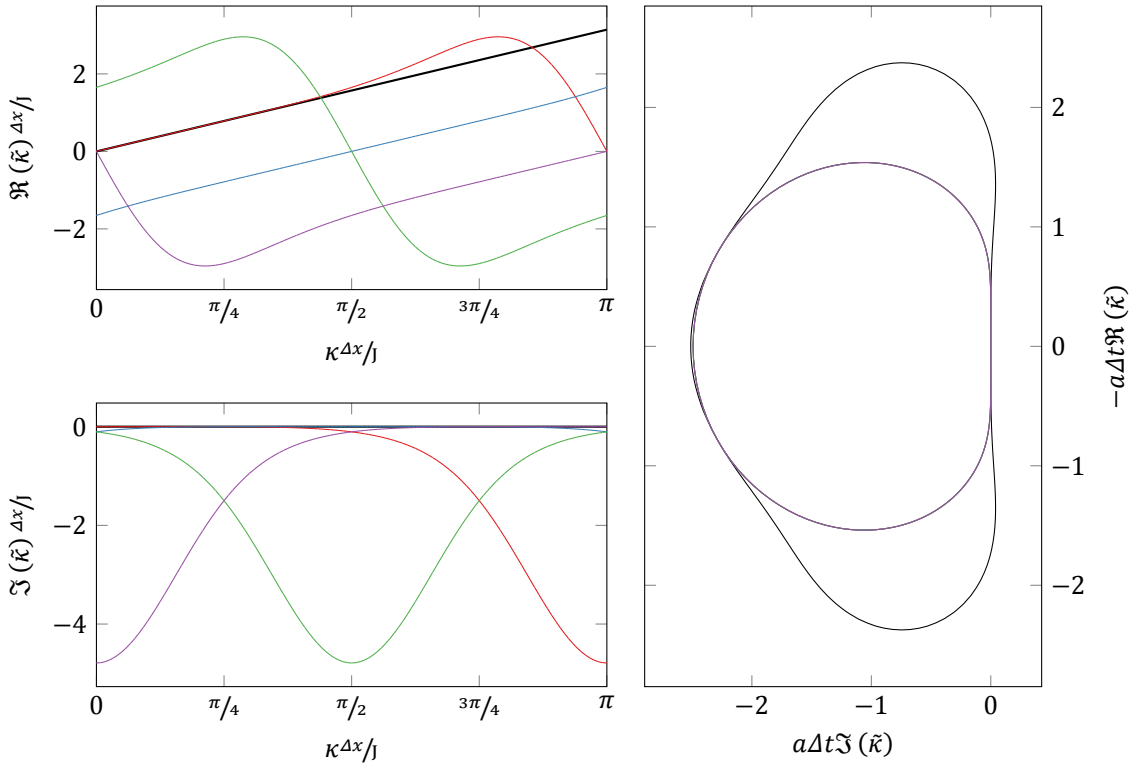


Figure 11.2: Modified wavenumbers (all eigenmodes); DGSEM, $\mathrm{p} = 3$.

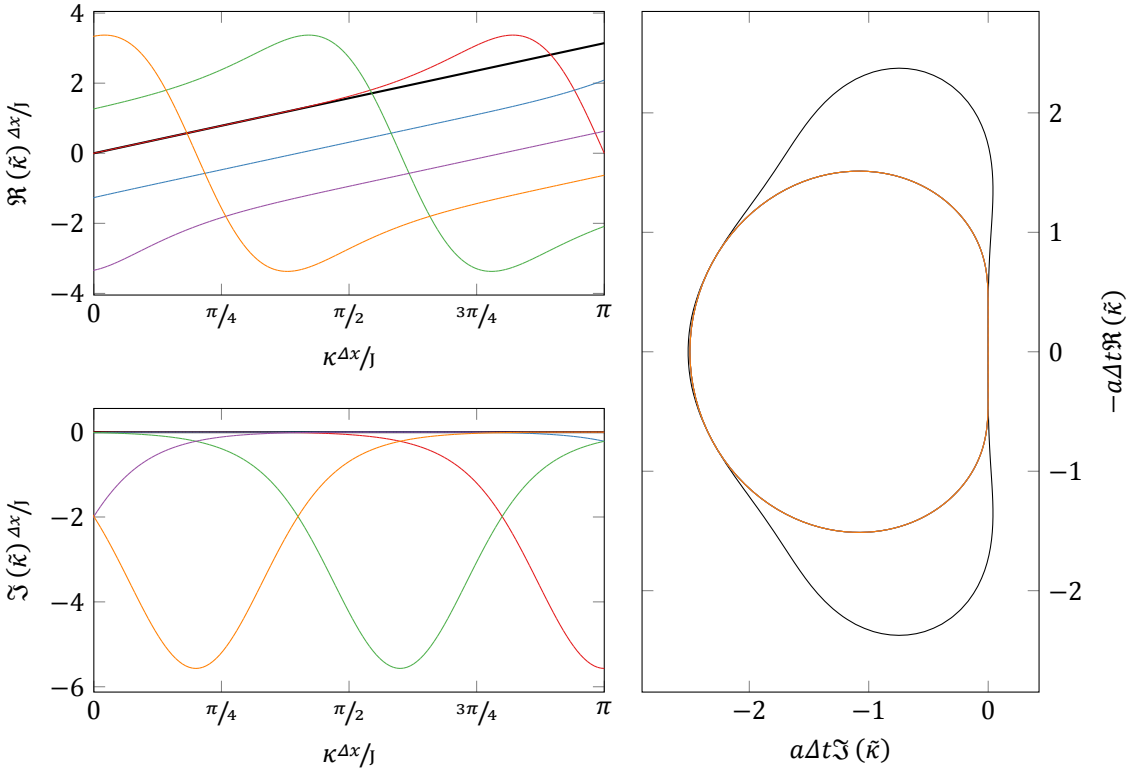


Figure 11.3: Modified wavenumbers (all eigenmodes); DGSEM, $\mathrm{p} = 4$.

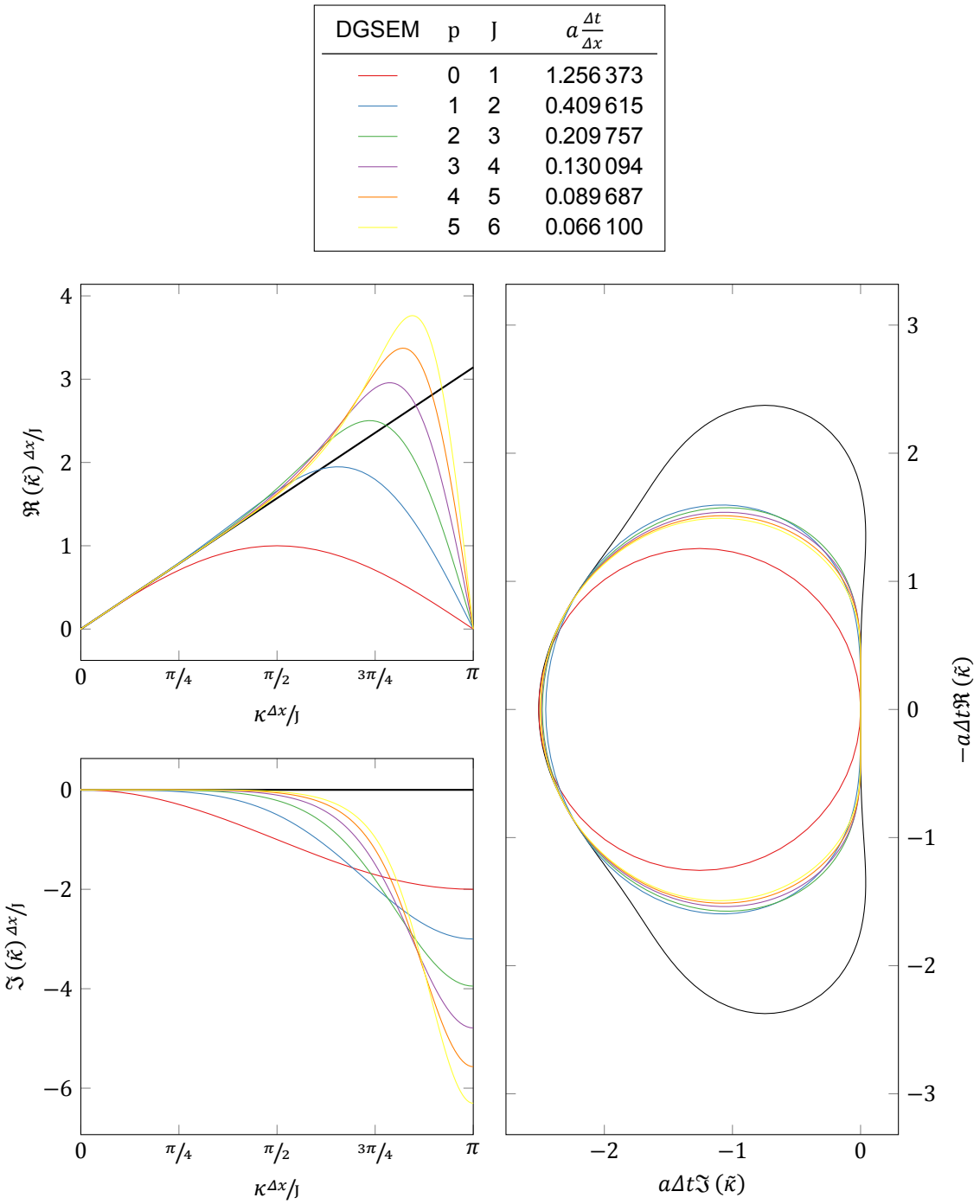


Figure 11.4: Modified wavenumbers (physical eigenmode only) of various DGSEM bases at low to moderate degrees.

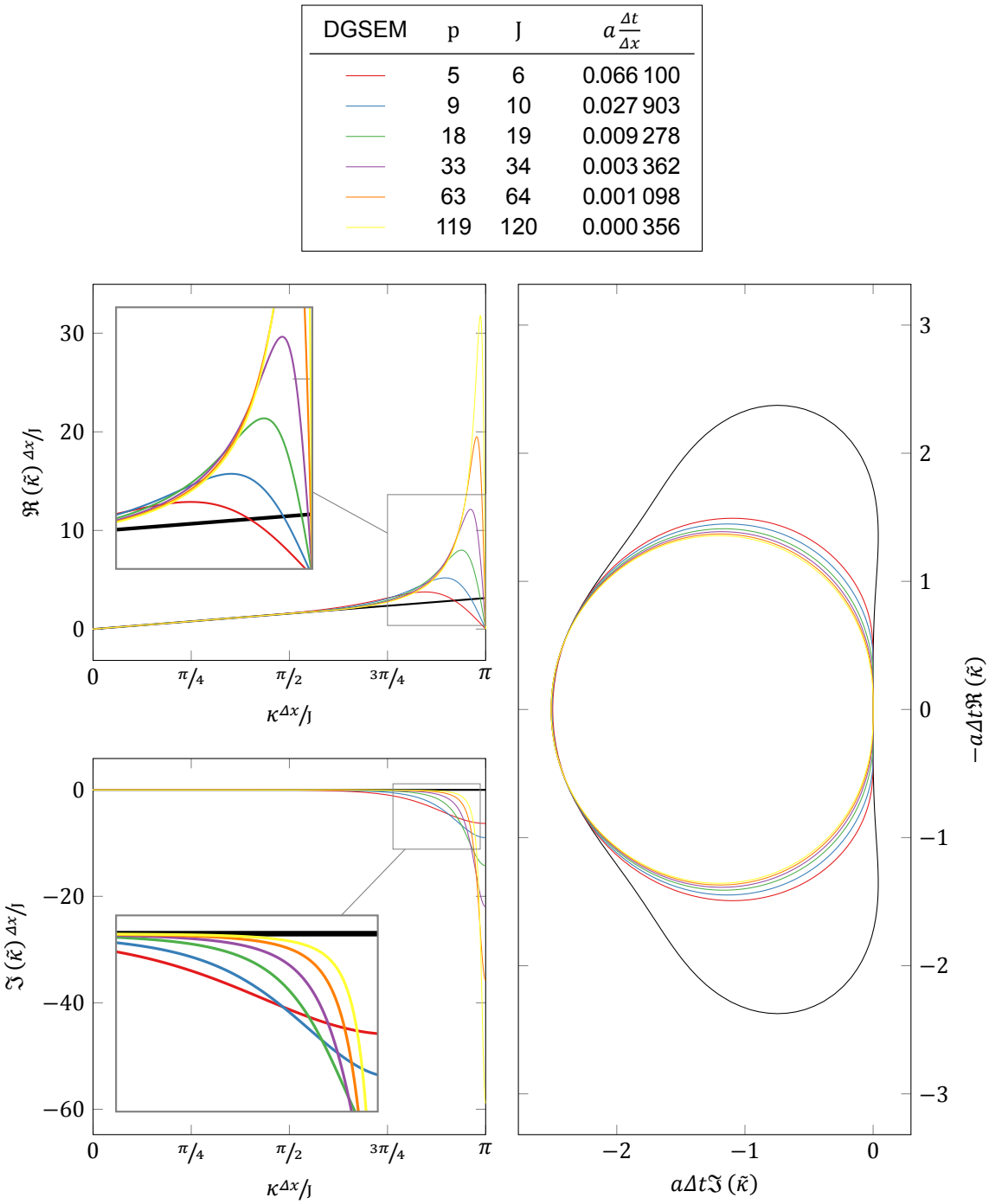


Figure 11.5: Modified wavenumbers (physical eigenmode only) of various DGSEM bases at moderate to high degrees.

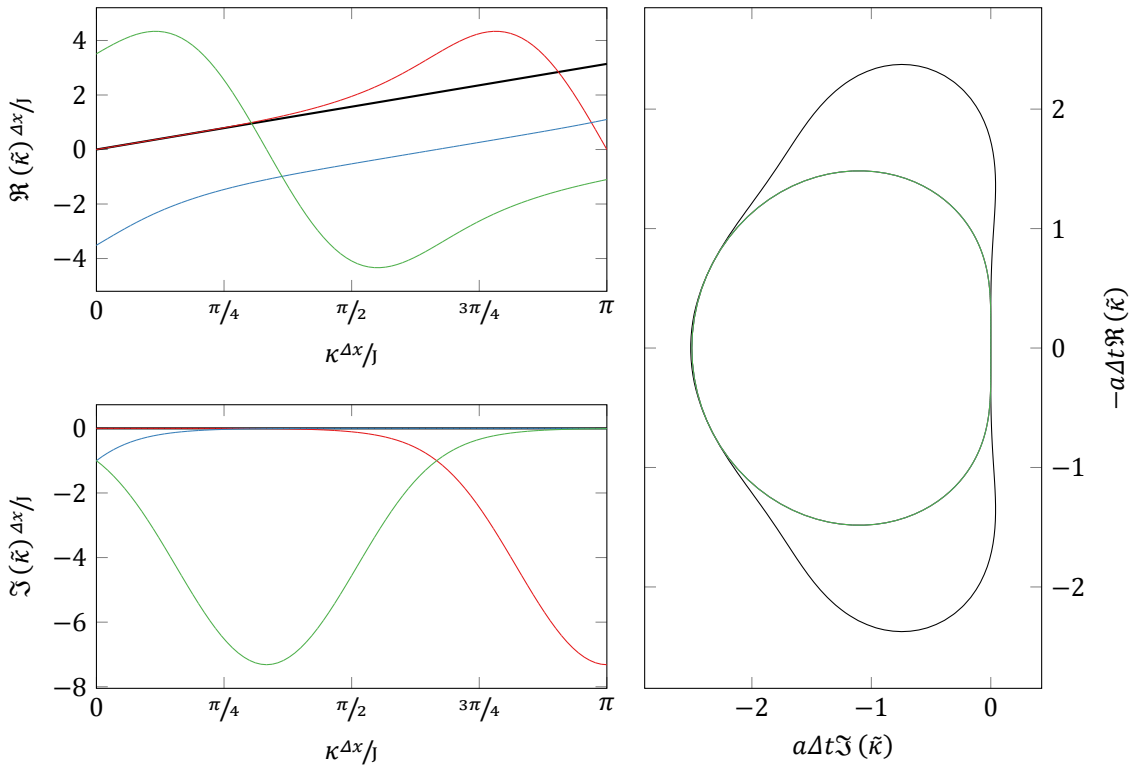


Figure 11.6: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 2$, $\frac{\eta_-}{2}$.

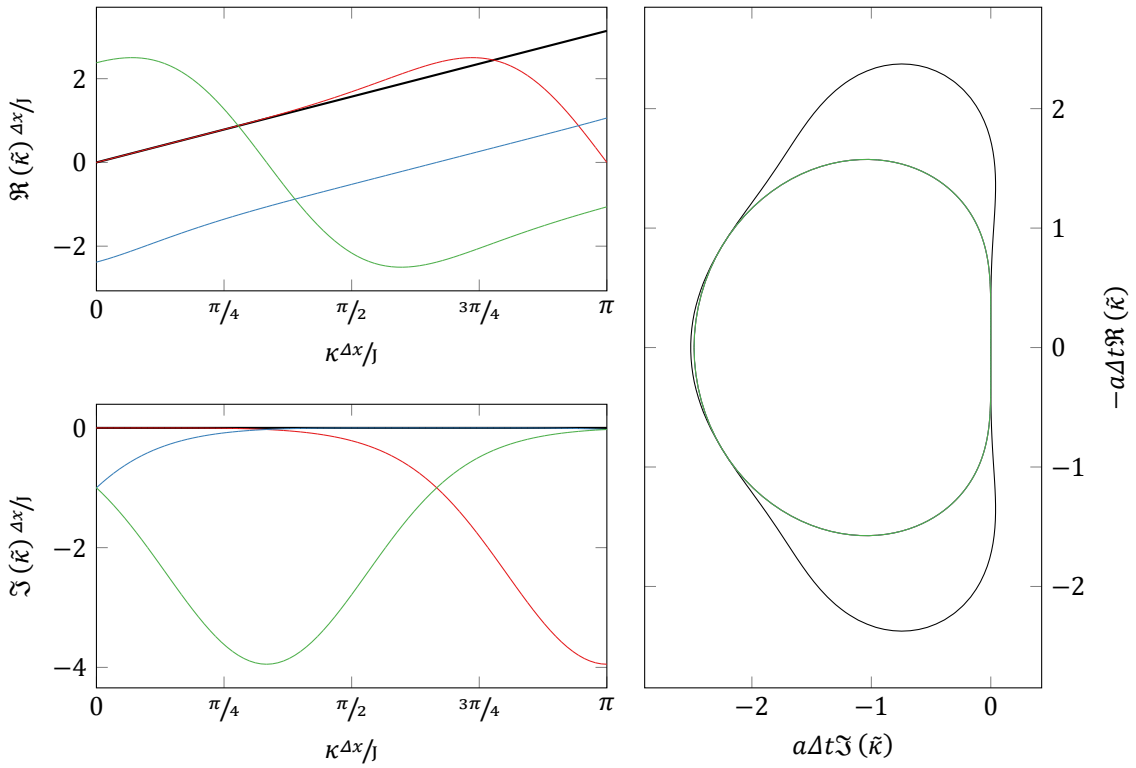


Figure 11.7: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 2$, η_{DG}.

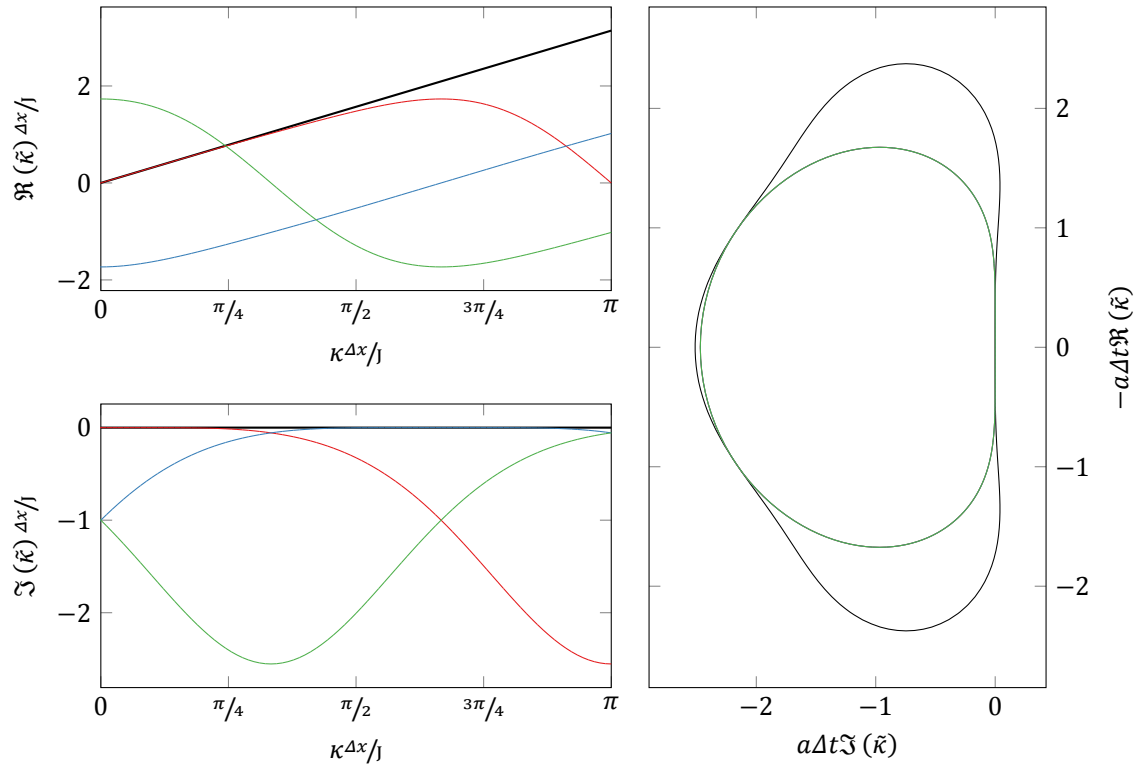


Figure 11.8: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 2$, η_{Ga}.

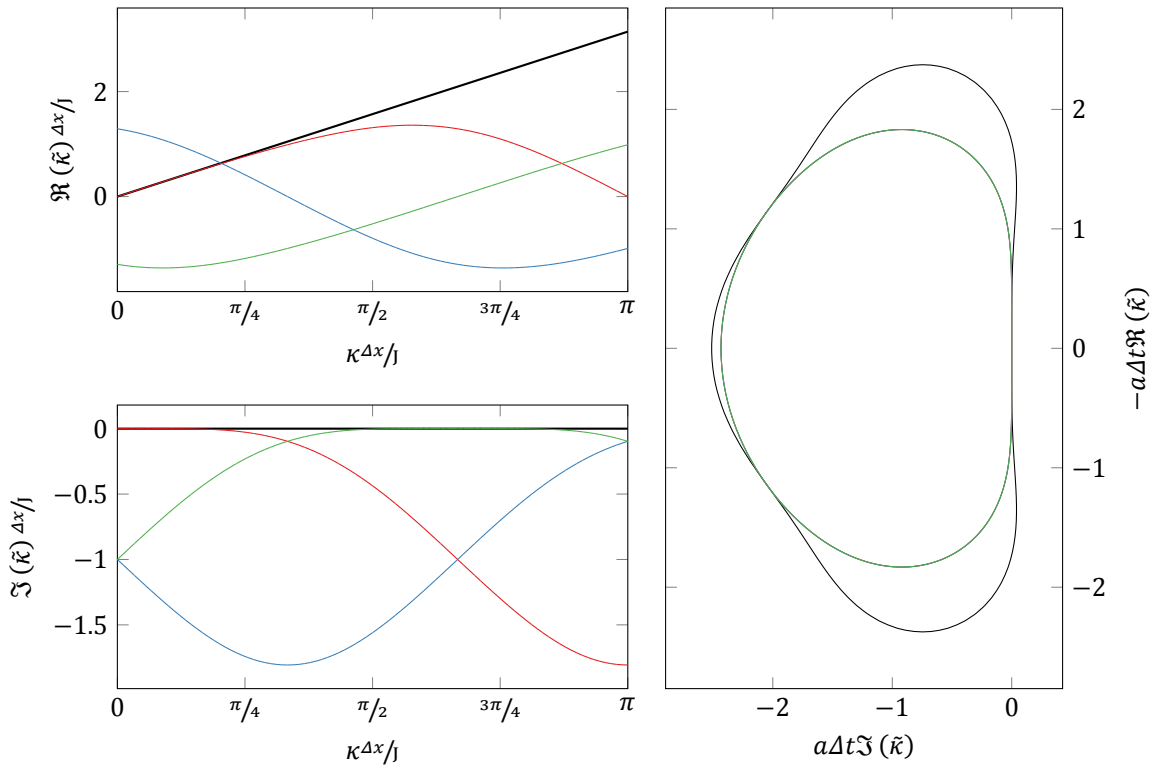


Figure 11.9: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 2$, η_2.

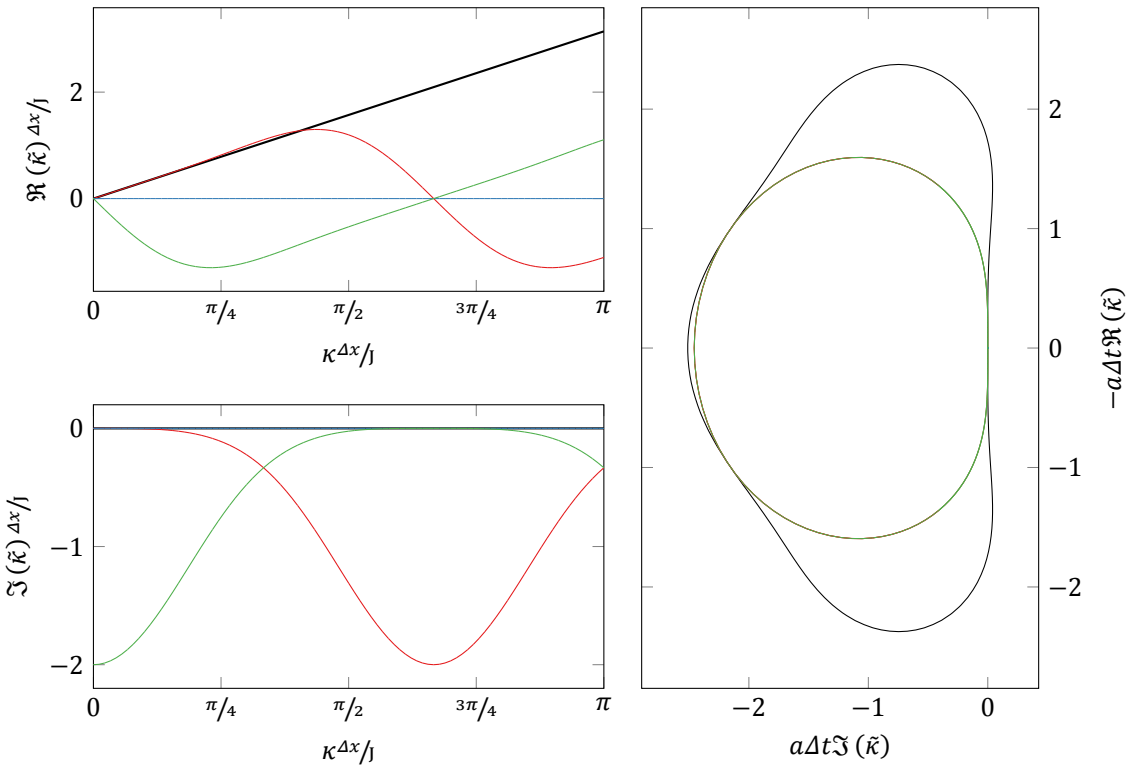


Figure 11.10: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 2$, η_∞.

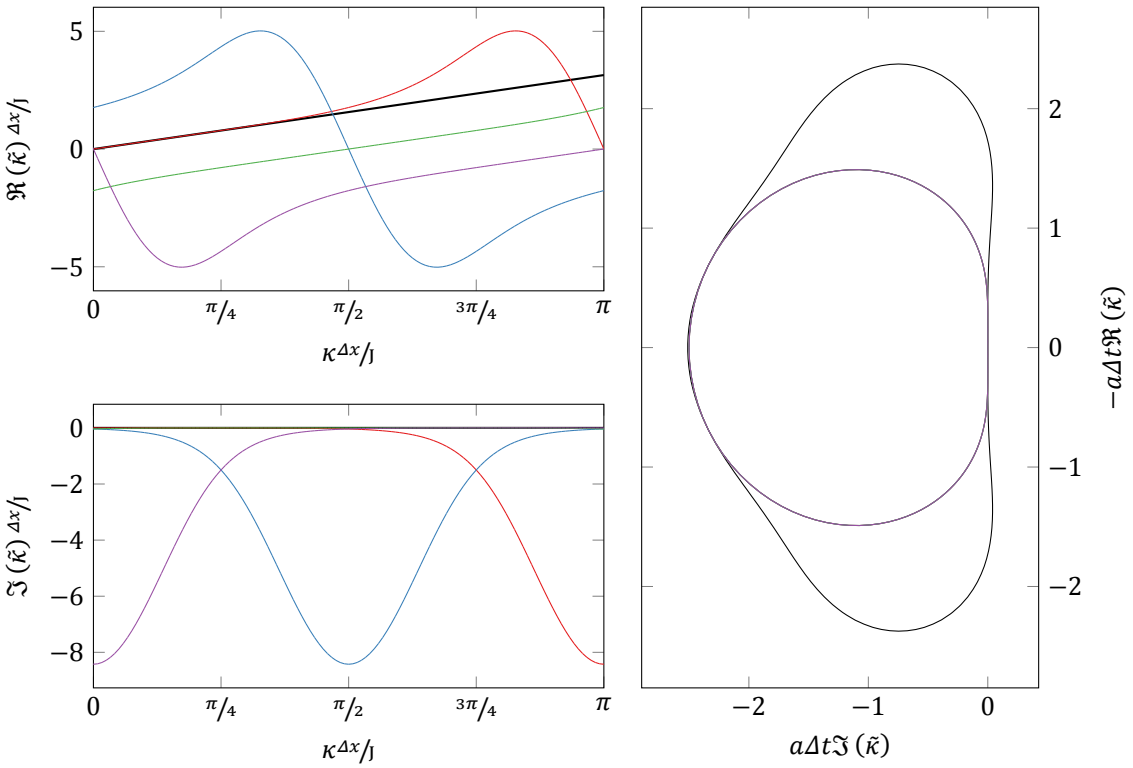


Figure 11.11: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 3$, $\frac{\eta_-}{2}$.

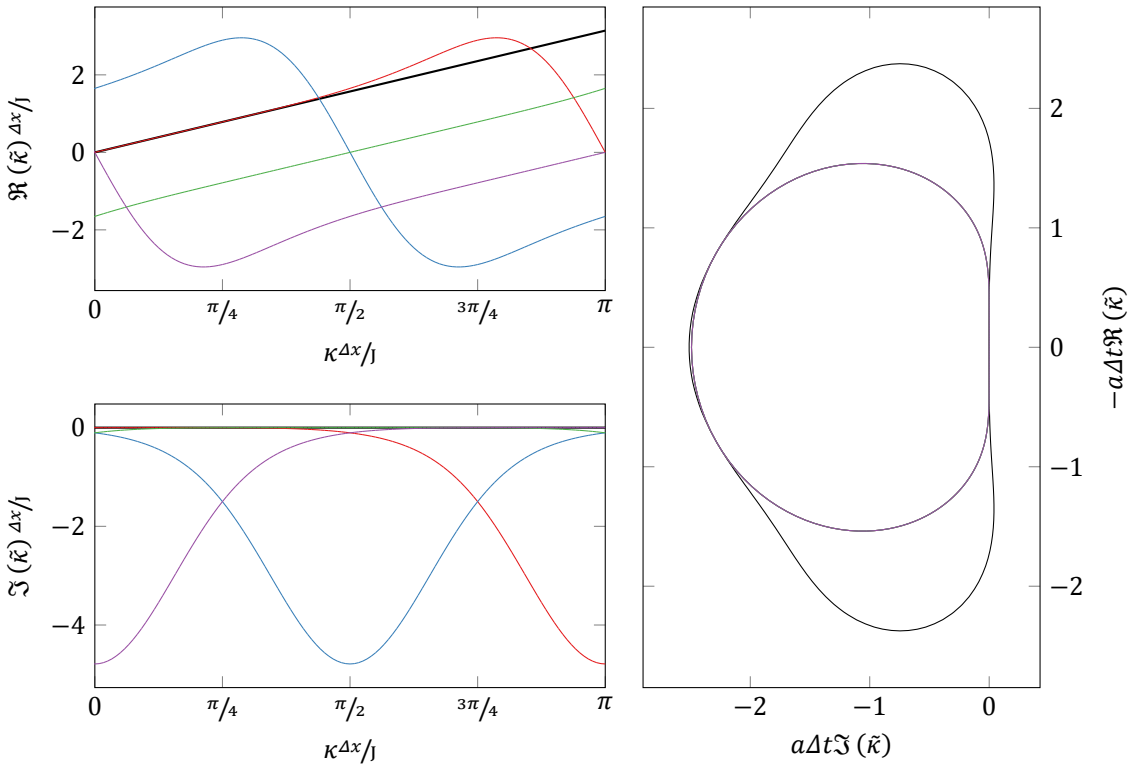


Figure 11.12: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 3$, η_{DG}.

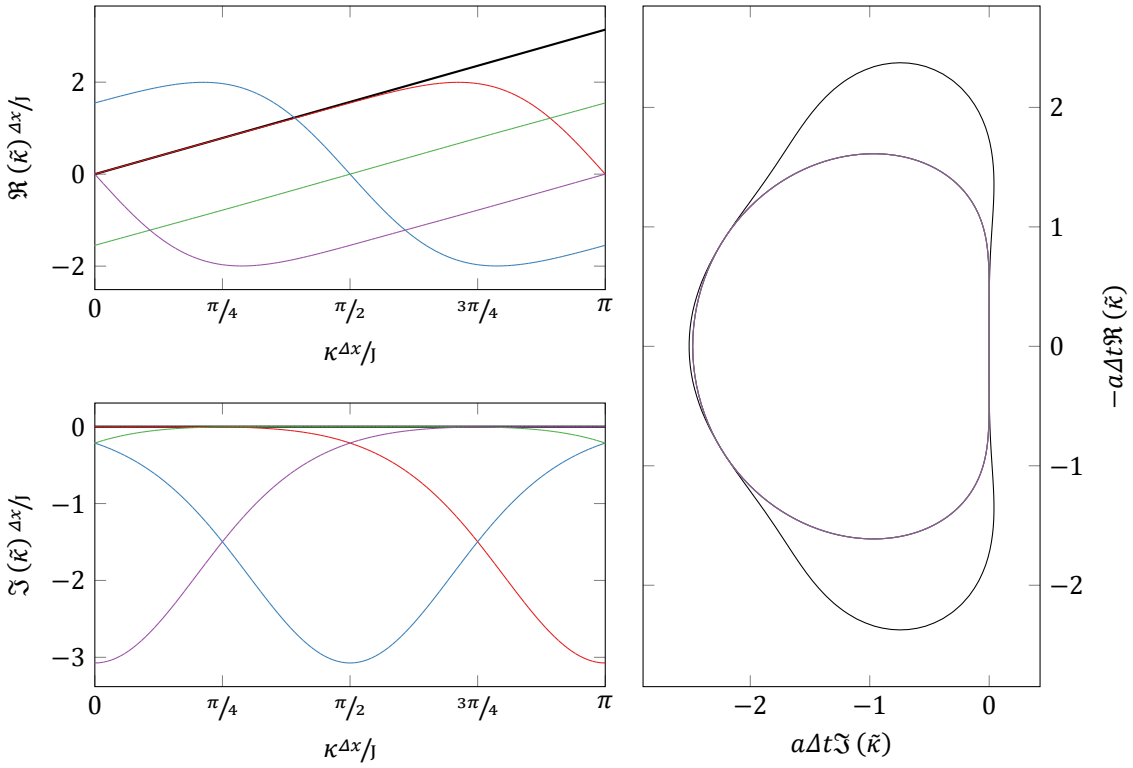


Figure 11.13: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 3$, η_{Ga}.

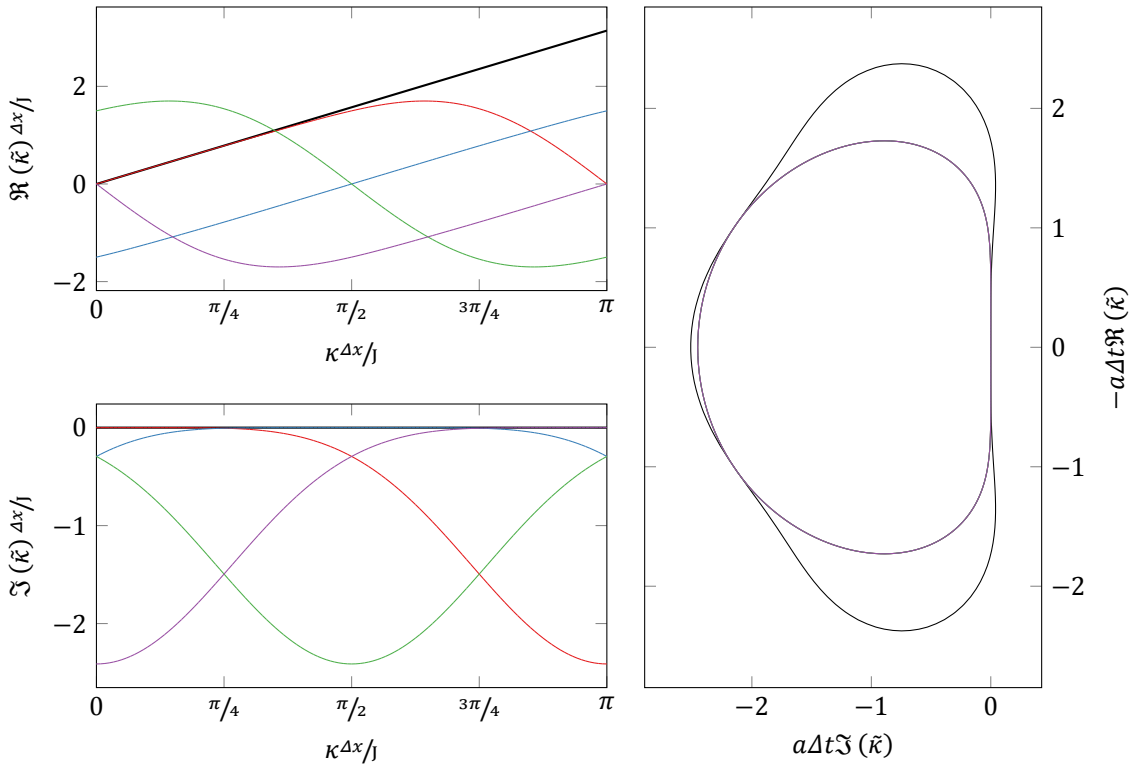


Figure 11.14: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 3$, η_2.

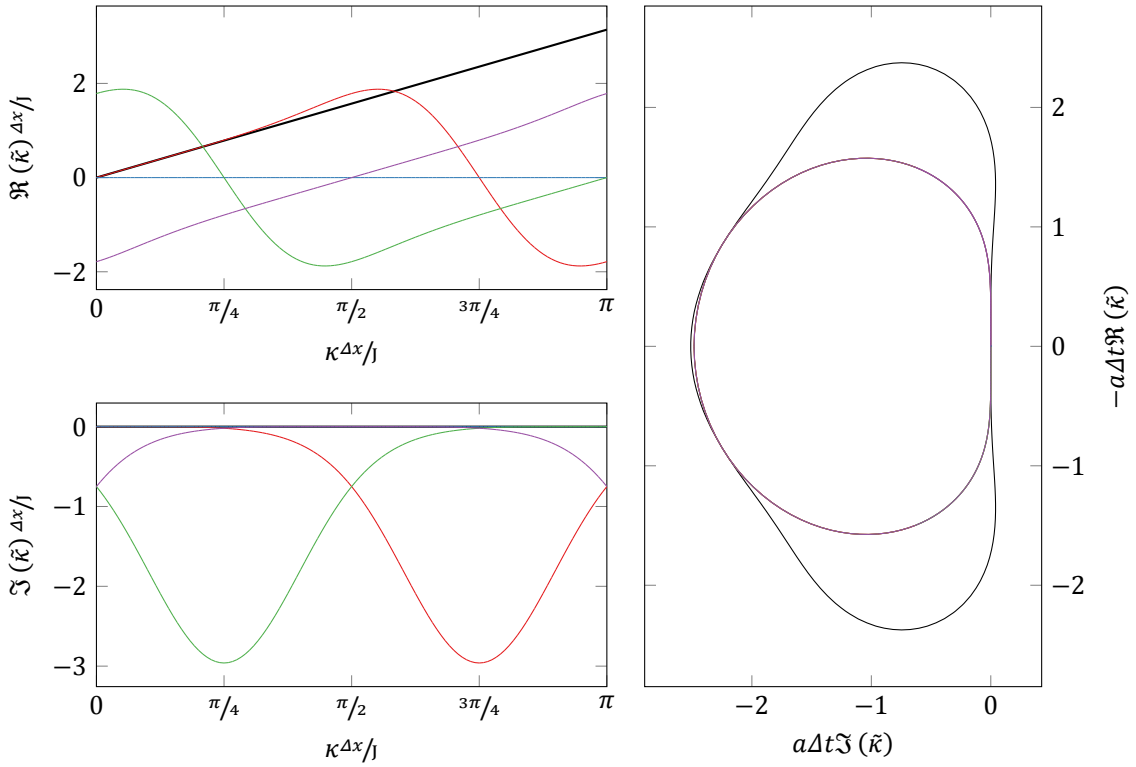


Figure 11.15: Modified wavenumbers (all eigenmodes); FR/CPR, $\mathrm{p} = 3$, η_∞.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	−0.500 000	3	0.114 004
	3	−0.500 000	4	0.074 218
	4	−0.500 000	5	0.053 041
	5	−0.500 000	6	0.040 194

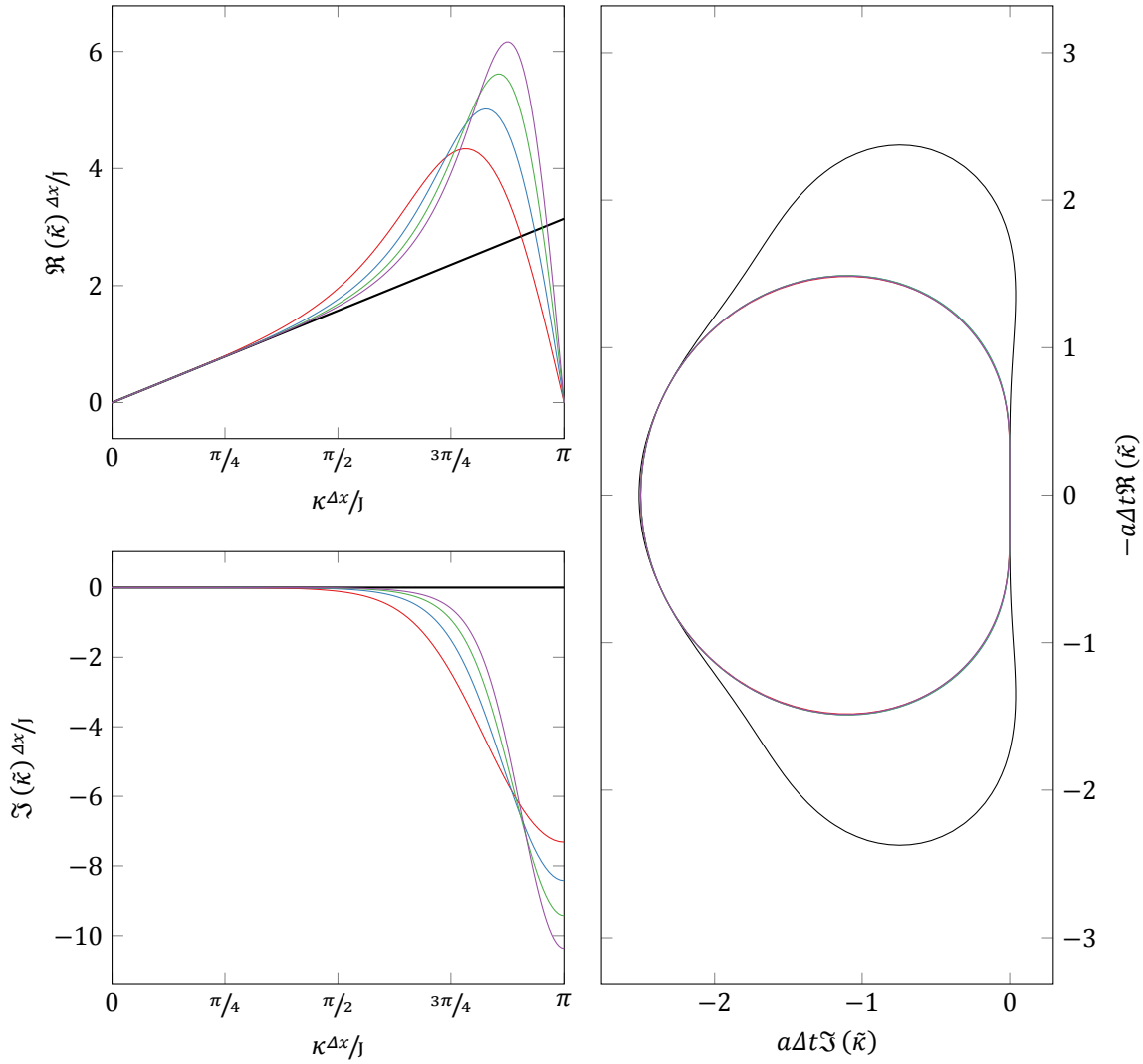


Figure 11.16: Modified wavenumbers (physical eigenmode) of FR/CPR at various degrees, with $\eta = \frac{\eta_-}{2}$.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	0.000 000	3	0.209 757
	3	0.000 000	4	0.130 094
	4	0.000 000	5	0.089 687
	5	0.000 000	6	0.066 100

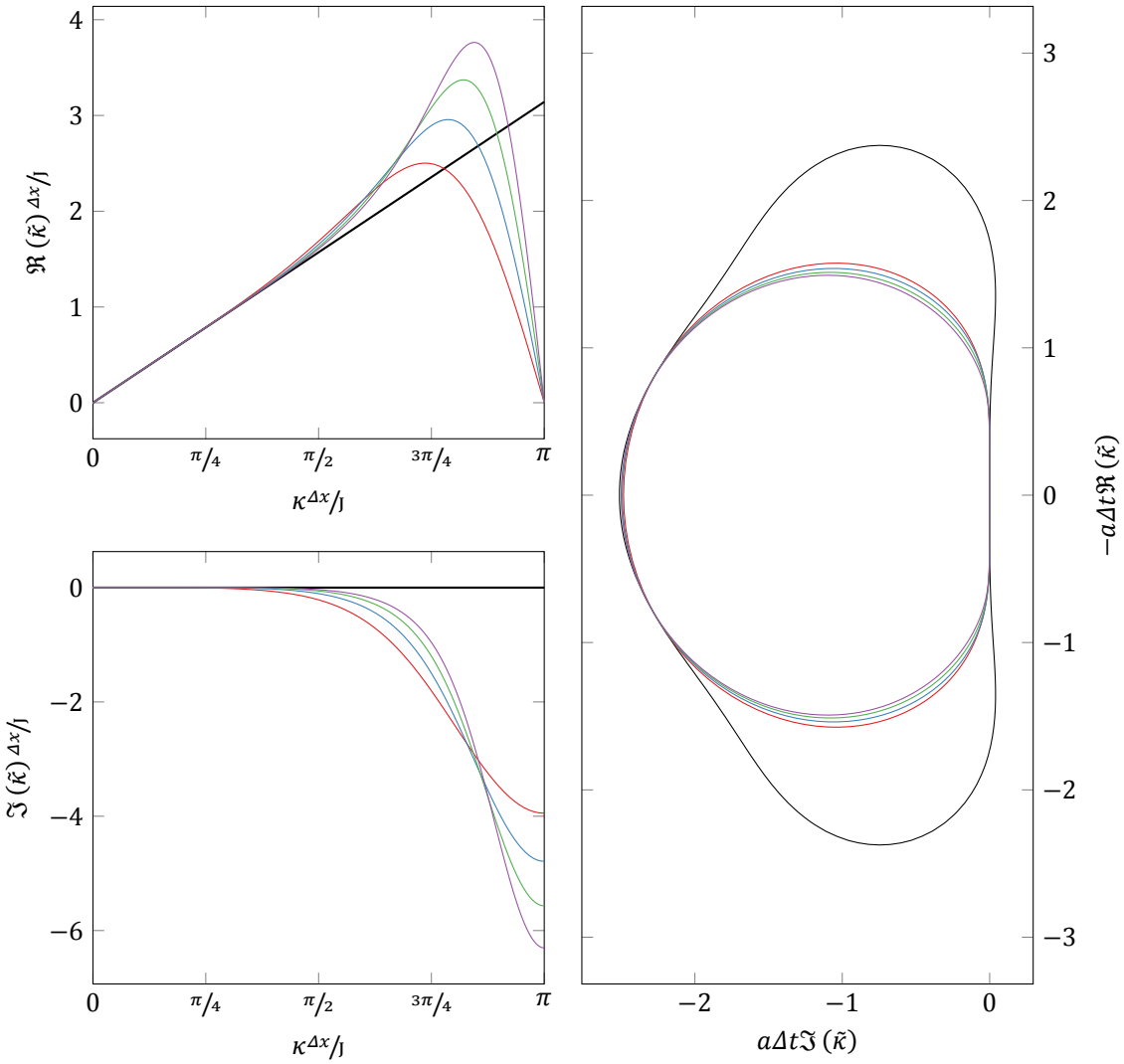


Figure 11.17: Modified wavenumbers (physical eigenmode) of FR/CPR at various degrees, with $\eta = \eta_{\mathrm{DG}}$.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	0.666 666	3	0.322 228
	3	0.750 000	4	0.201 876
	4	0.800 000	5	0.139 791
	5	0.833 333	6	0.103 338

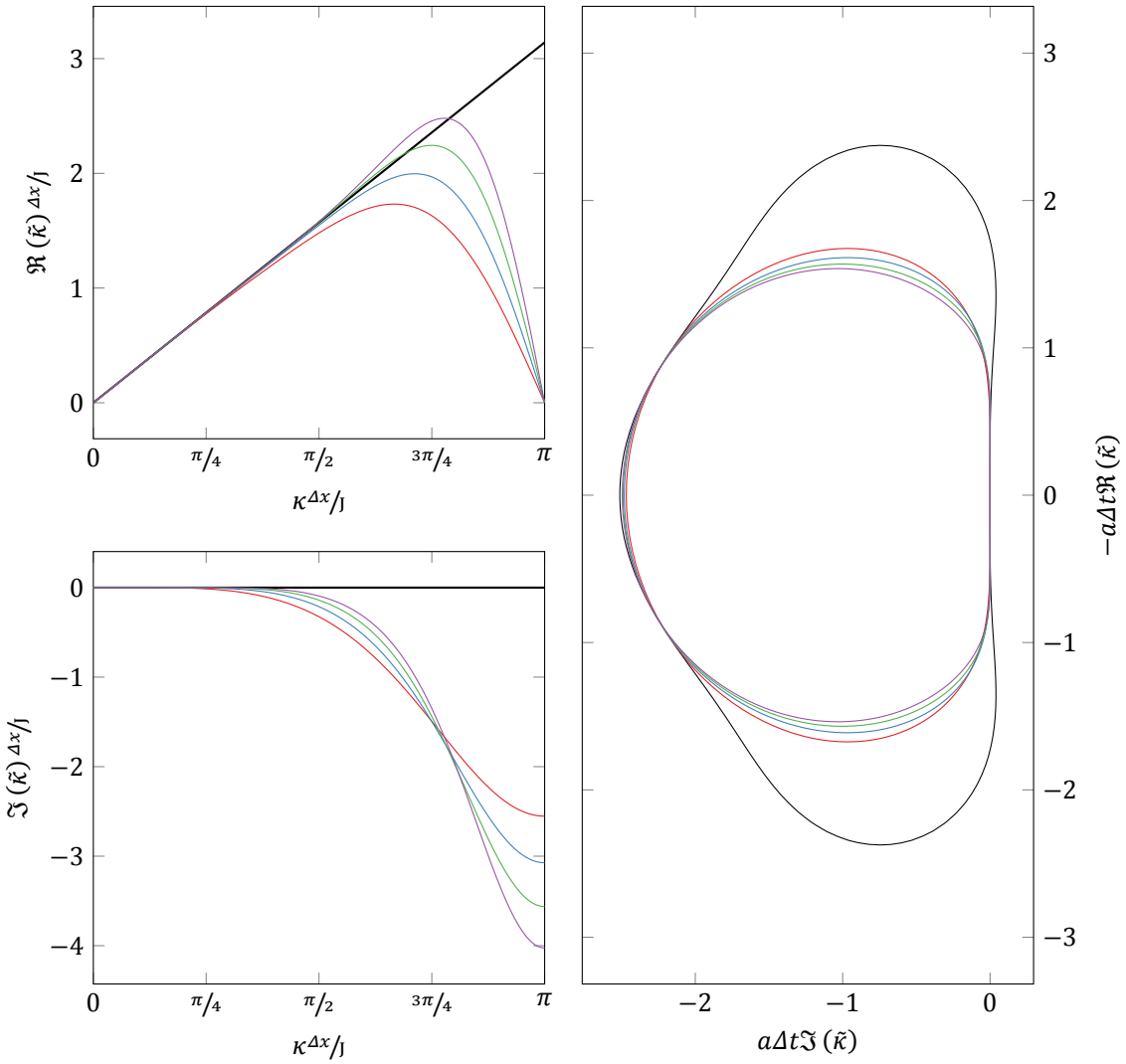


Figure 11.18: Modified wavenumbers (physical eigenmode) of FR/CPR at various degrees, with $\eta = \eta_{\mathrm{Ga}}$.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	1.500 000	3	0.449 069
	3	1.333 333	4	0.254 283
	4	1.250 000	5	0.167 583
	5	1.200 000	6	0.120 301

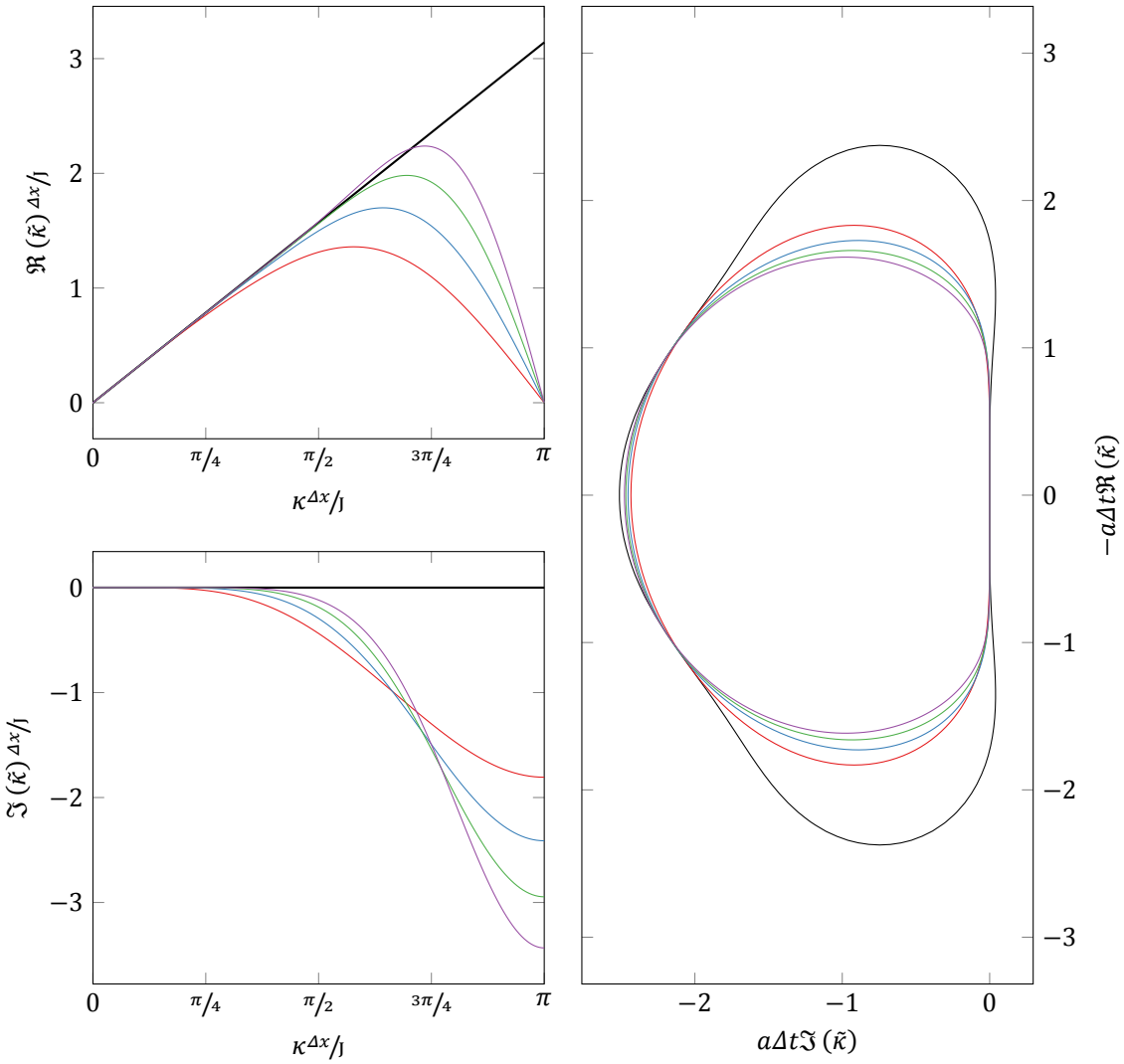


Figure 11.19: Modified wavenumbers (physical eigenmode) of FR/CPR at various degrees, with $\eta = \eta_2$.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	1×10^{16}	3	0.409 594
	3	1×10^{16}	4	0.209 754
	4	1×10^{16}	5	0.130 094
	5	1×10^{16}	6	0.089 687

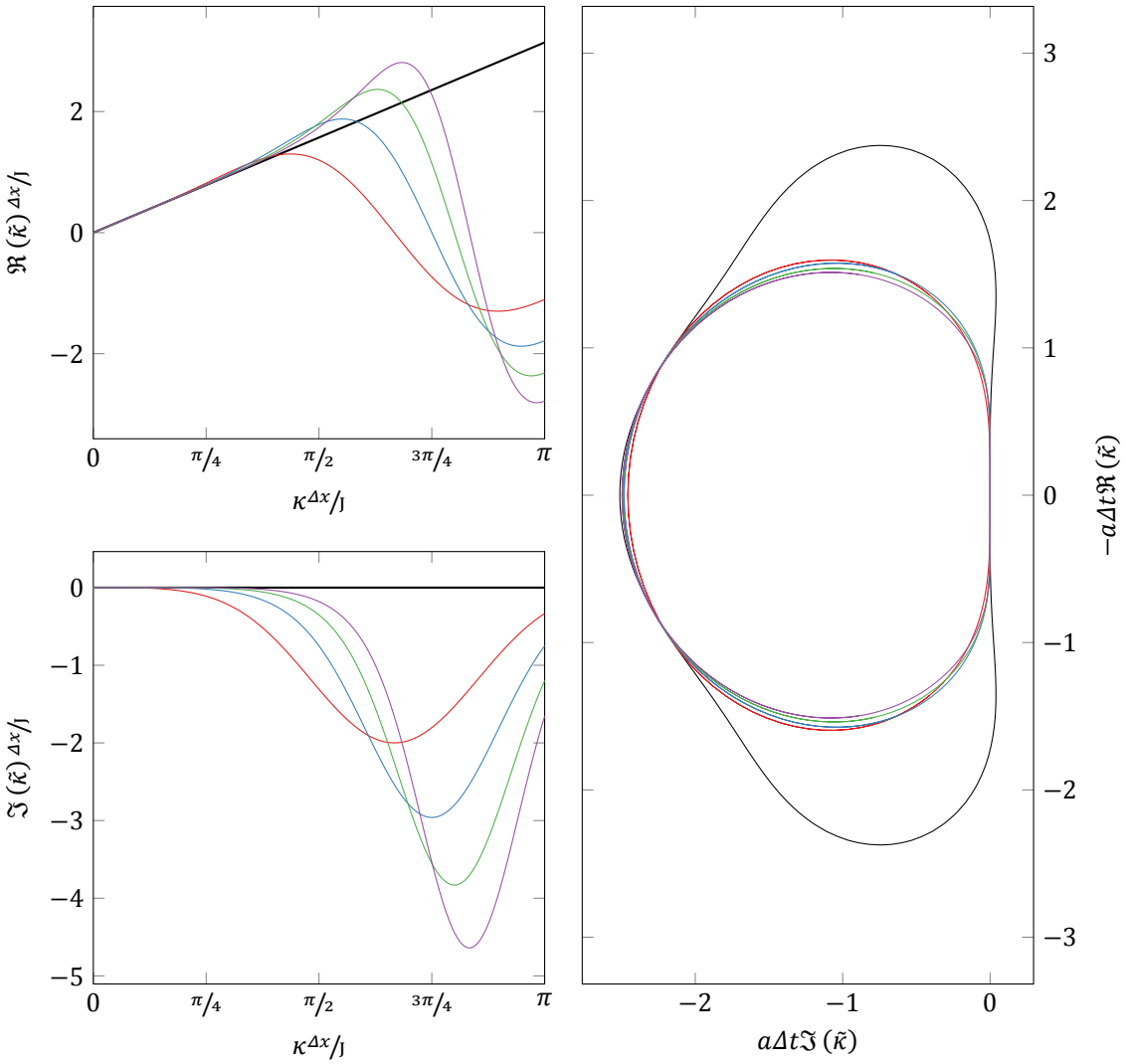


Figure 11.20: Modified wavenumbers (physical eigenmode) of FR/CPR at various degrees, with $\eta = \eta_\infty$.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	2	$\eta_{-/2}$	3	0.114 004
	2	η_{DG}	3	0.209 757
	2	η_{Ga}	3	0.322 228
	2	η_2	3	0.449 069
	2	η_∞	3	0.409 594

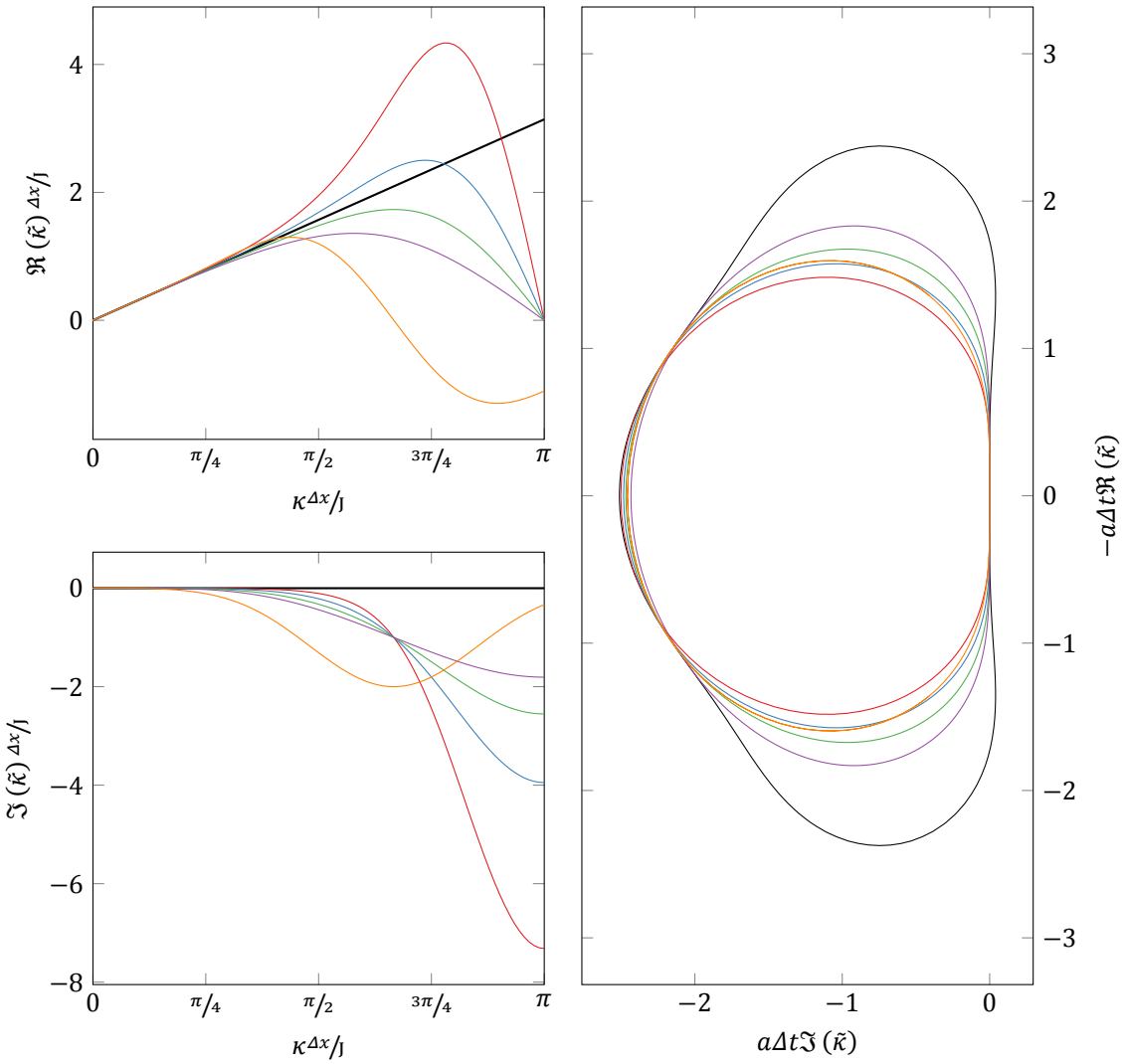


Figure 11.21: Modified wavenumbers (physical eigenmode) of 3rd order FR/CPR, for several correction functions.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	3	$\eta_{-/2}$	4	0.074 218
	3	η_{DG}	4	0.130 094
	3	η_{Ga}	4	0.201 876
	3	η_2	4	0.254 283
	3	η_∞	4	0.209 754

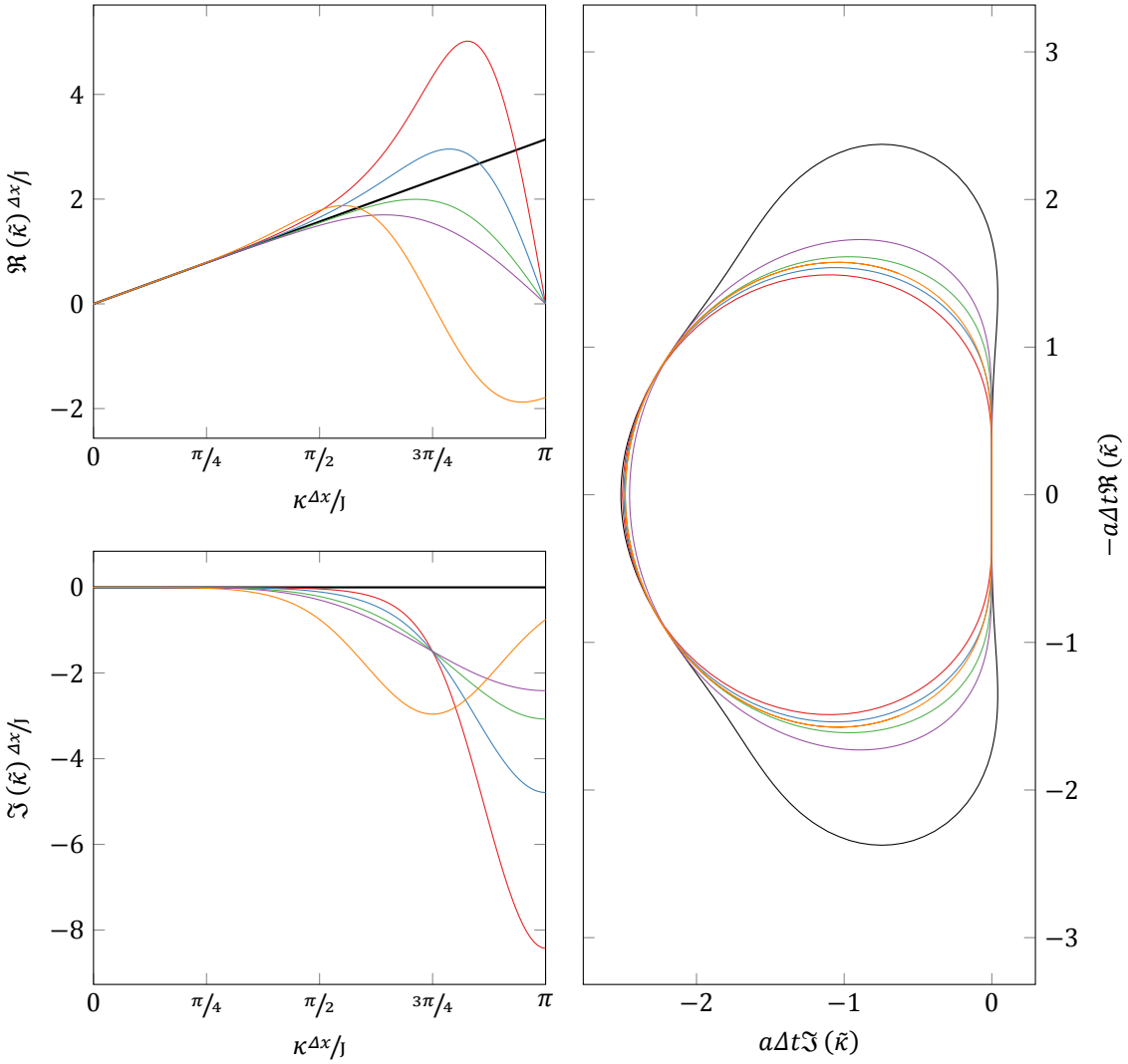


Figure 11.22: Modified wavenumbers (physical eigenmode) of 4th order FR/CPR, for several correction functions.

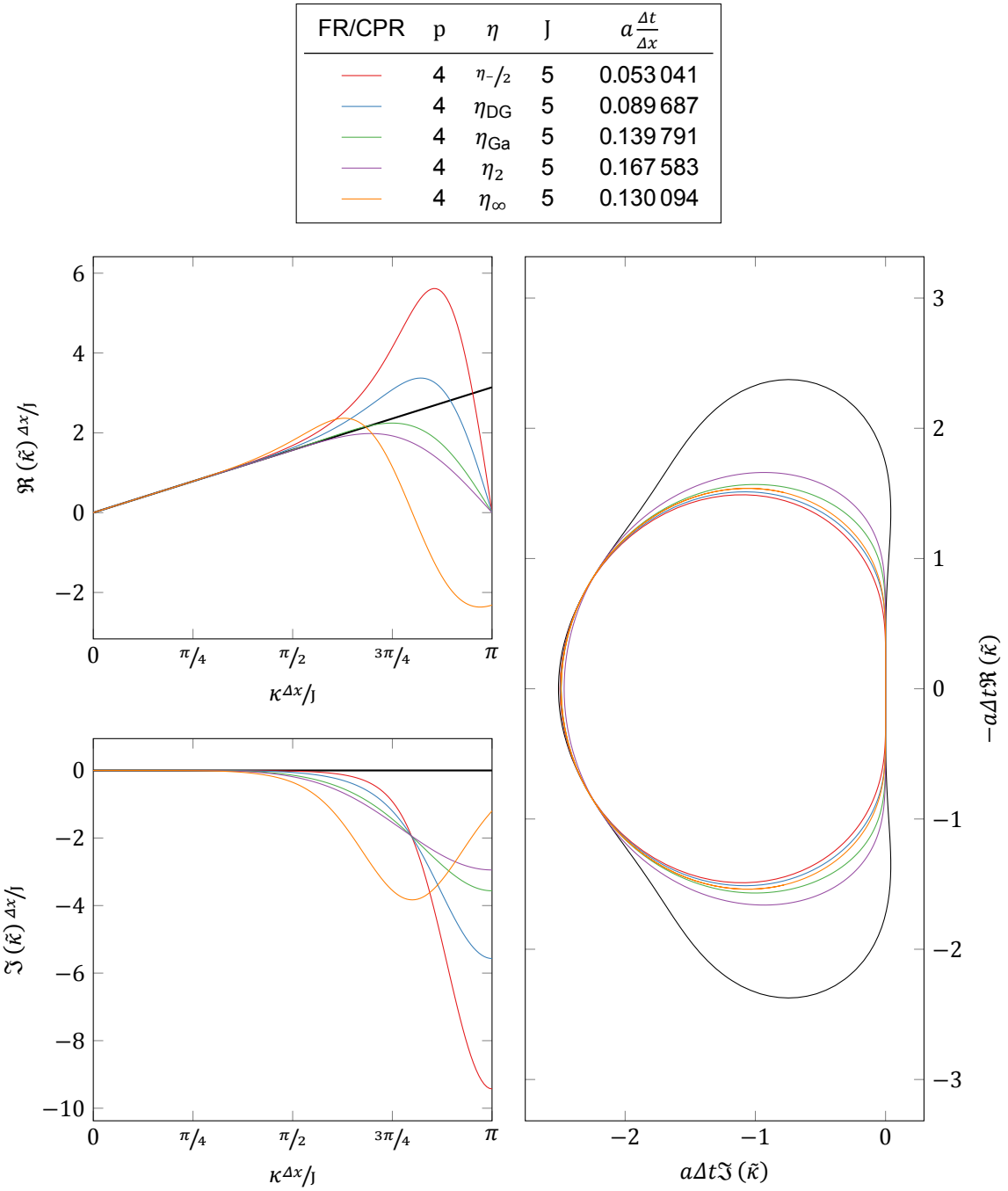


Figure 11.23: Modified wavenumbers (physical eigenmode) of 5th order FR/CPR, for several correction functions.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	5	$\eta_{-/2}$	6	0.040 194
	5	η_{DG}	6	0.066 100
	5	η_{Ga}	6	0.103 338
	5	η_2	6	0.120 301
	5	η_∞	6	0.089 687

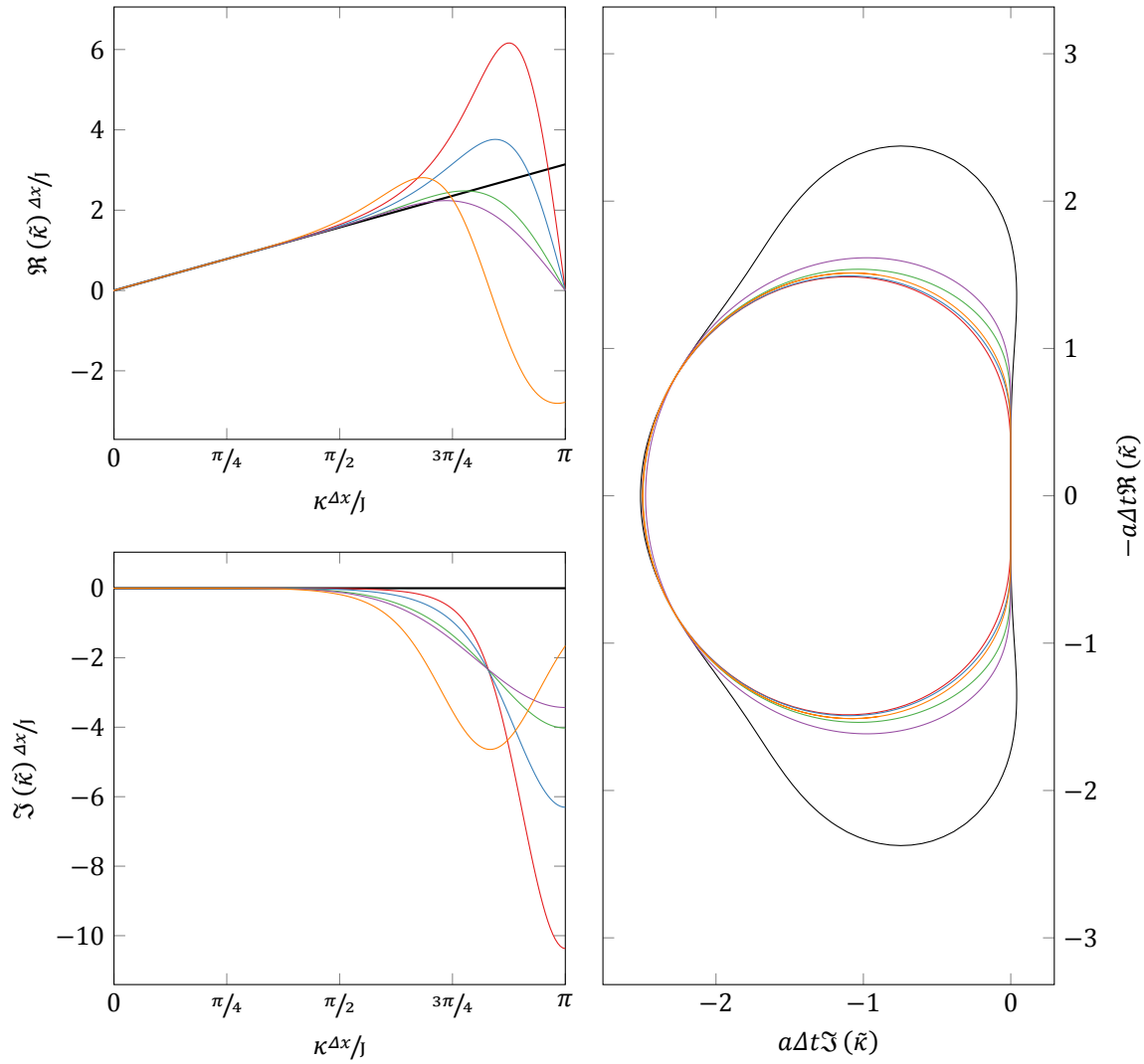


Figure 11.24: Modified wavenumbers (physical eigenmode) of 6th order FR/CPR, for several correction functions.

FR/CPR	p	η	J	$a\frac{\Delta t}{\Delta x}$
	119	$\eta_{-}/_{2}$	120	0.000 279
	119	η_{DG}	120	0.000 356
	119	η_{Ga}	120	0.000 603
	119	η_{2}	120	0.000 620
	119	η_{∞}	120	0.000 361

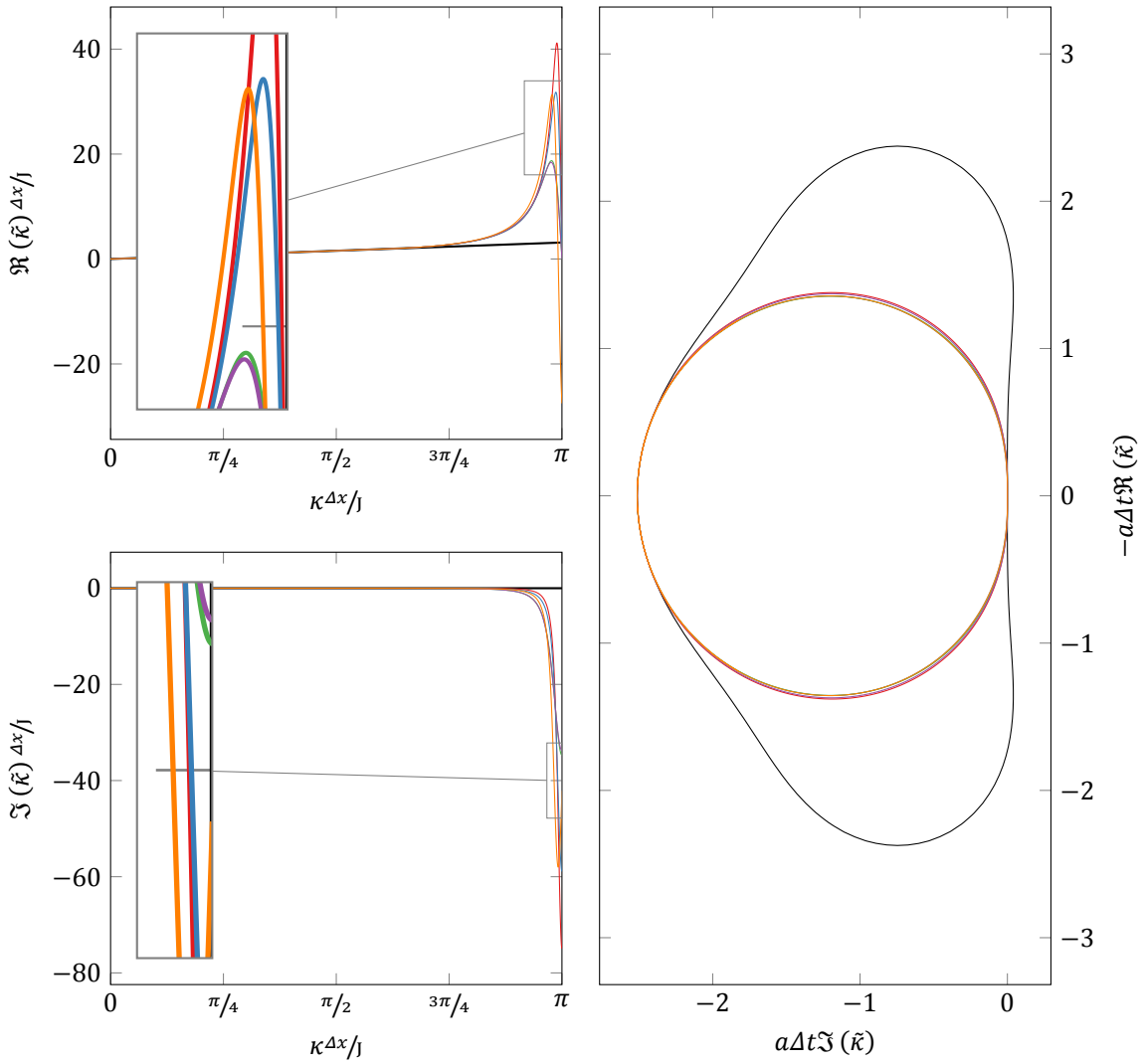


Figure 11.25: Modified wavenumbers (physical eigenmode) of high order FR/CPR, for several correction functions.

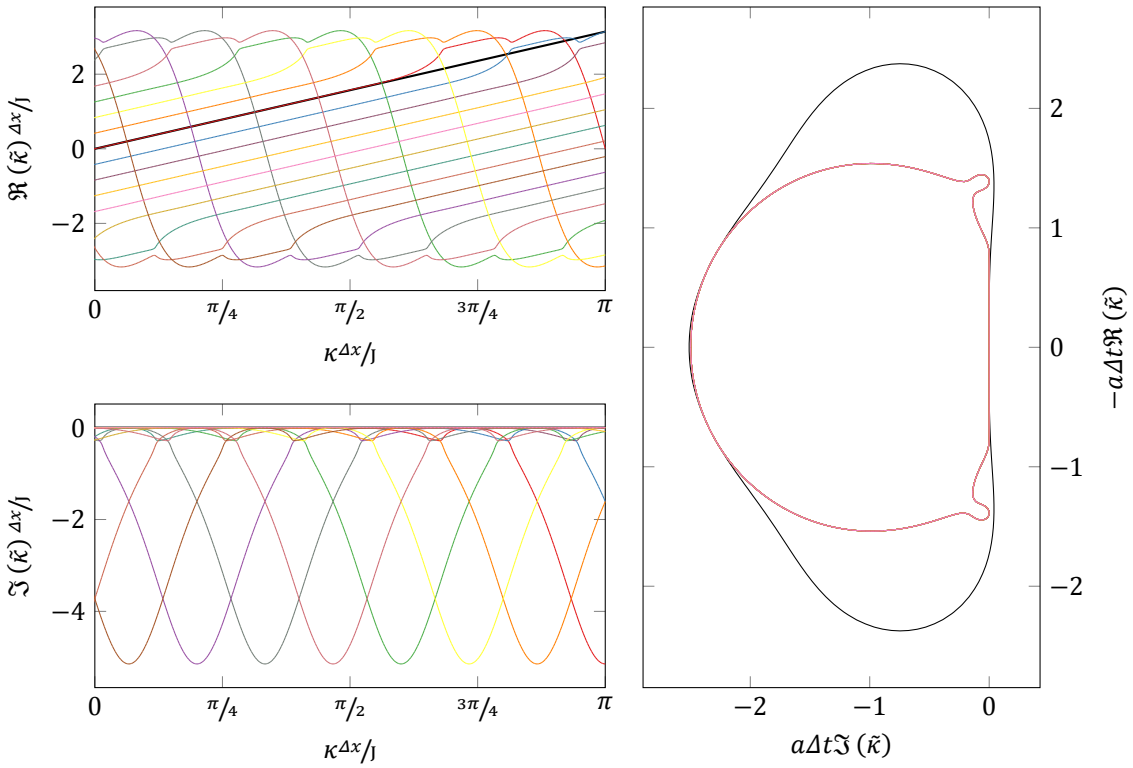


Figure 11.26: Physical (1) and spurious (14) eigenmodes of DGIGA, $\mathrm{k} = 2$, $\mathrm{p} = 7$, C^0 (cf. 11.27).

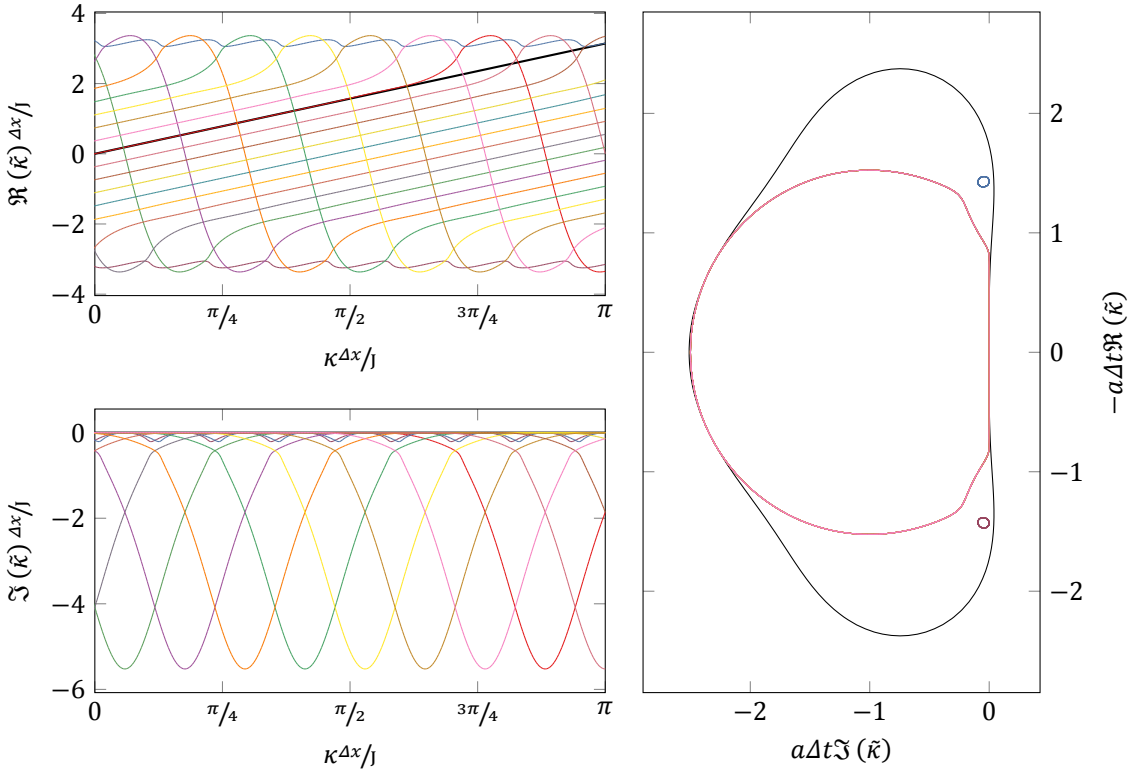


Figure 11.27: Physical (1), “bubble” (2) and spurious (14) eigenmodes of DGIGA, $\mathrm{k} = 2$, $\mathrm{p} = 8$, C^0 (cf. 11.26).

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	2	2	0	5	0.163 094
	2	3	0	7	0.101 465
	2	4	0	9	0.070 291
	2	5	0	11	0.052 068

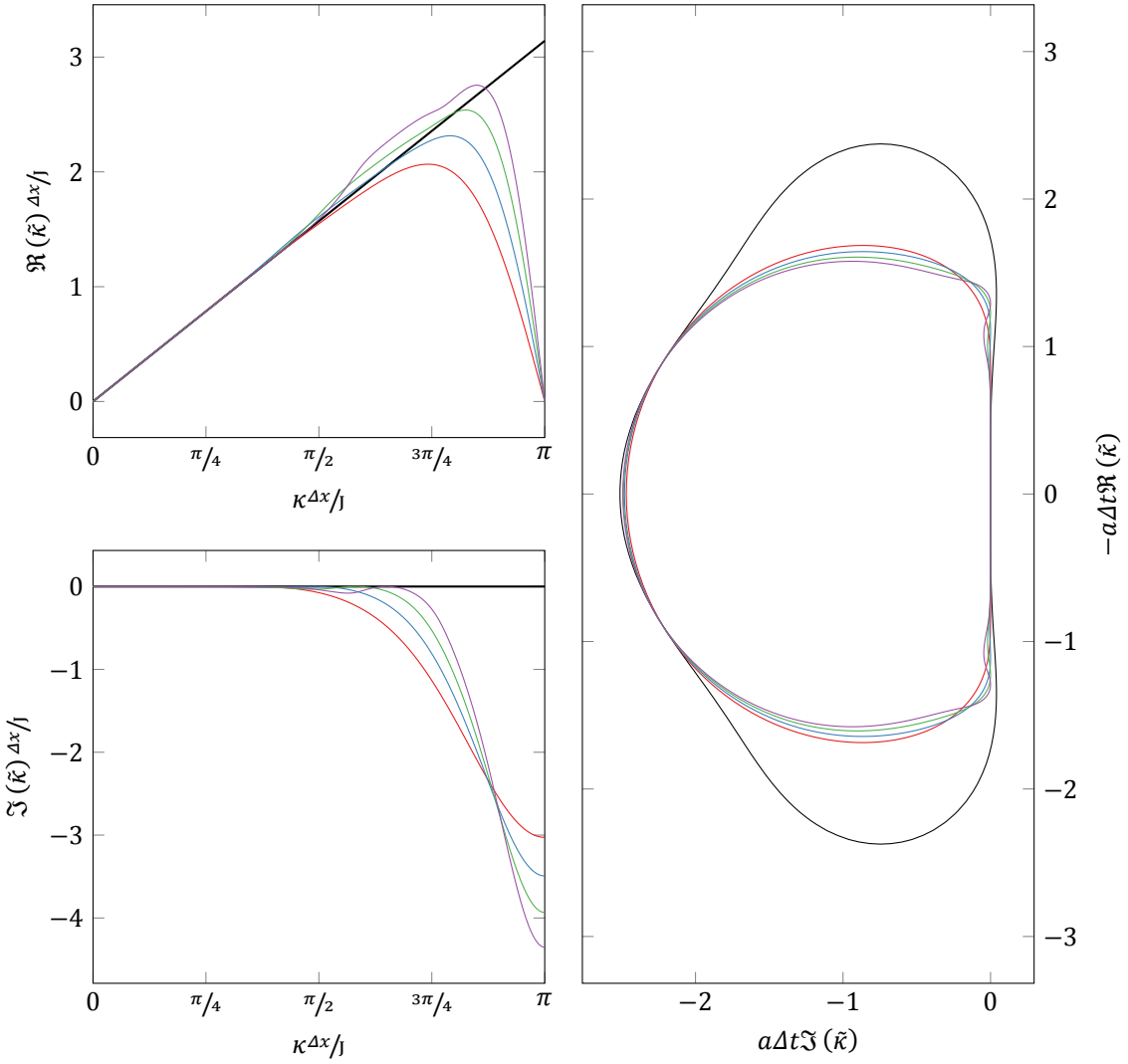


Figure 11.28: Effect of the degree to the modified wavenumbers (physical eigenmode) in DGIGA, for 2 breakpoint spans per patch and minimal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	4	2	0	6	0.125 265
	4	3	0	7	0.079 422
	4	4	0	8	0.056 268
	4	5	0	9	0.042 636

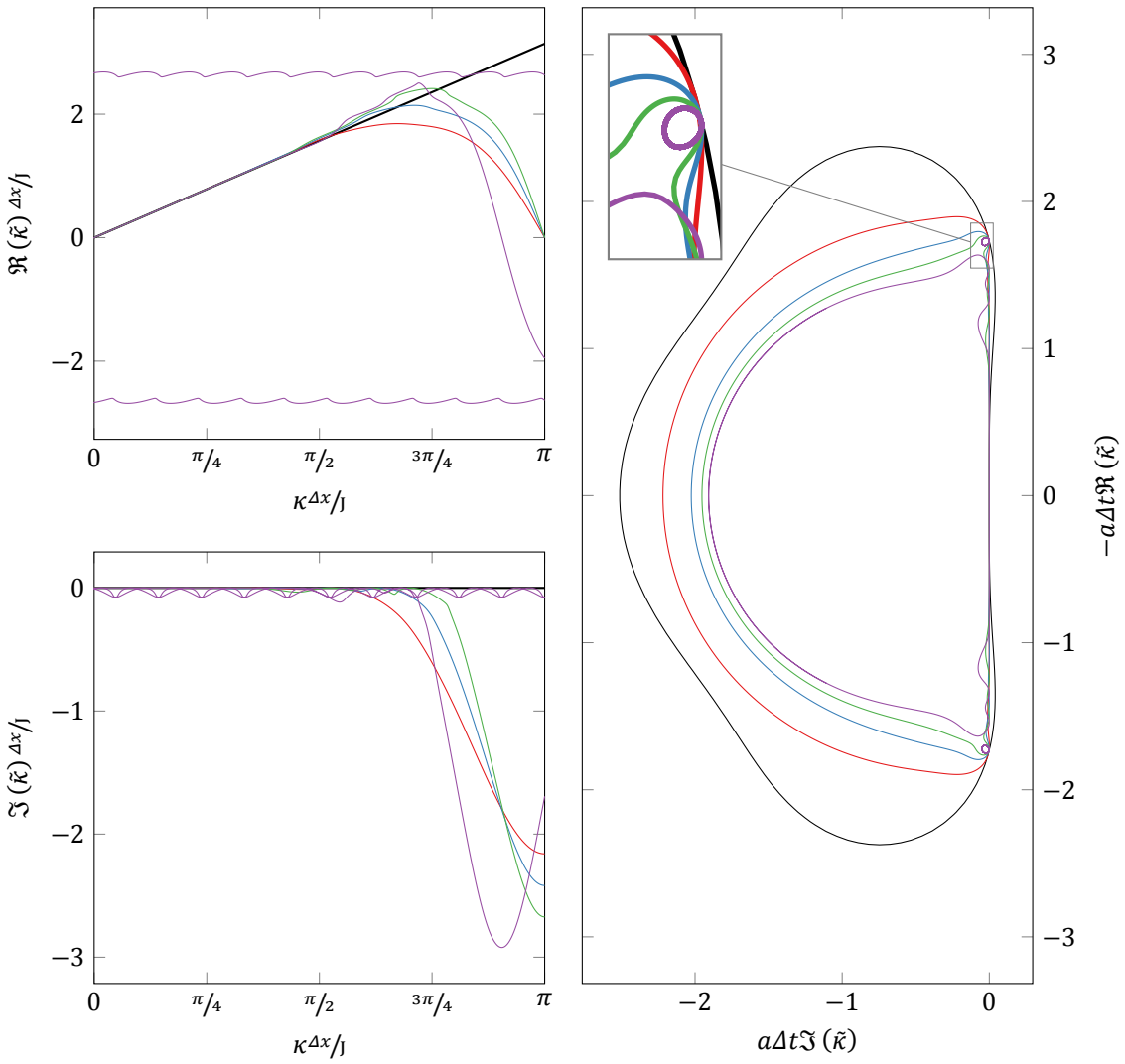


Figure 11.29: Effect of the degree to the modified wavenumbers (physical and “bubble” eigenmodes) in DGIGA, for 4 breakpoint spans per patch and minimal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	8	2	0	17	0.052 704
	8	3	0	25	0.030 632
	8	4	0	33	0.020 679
	8	5	0	41	0.015 060

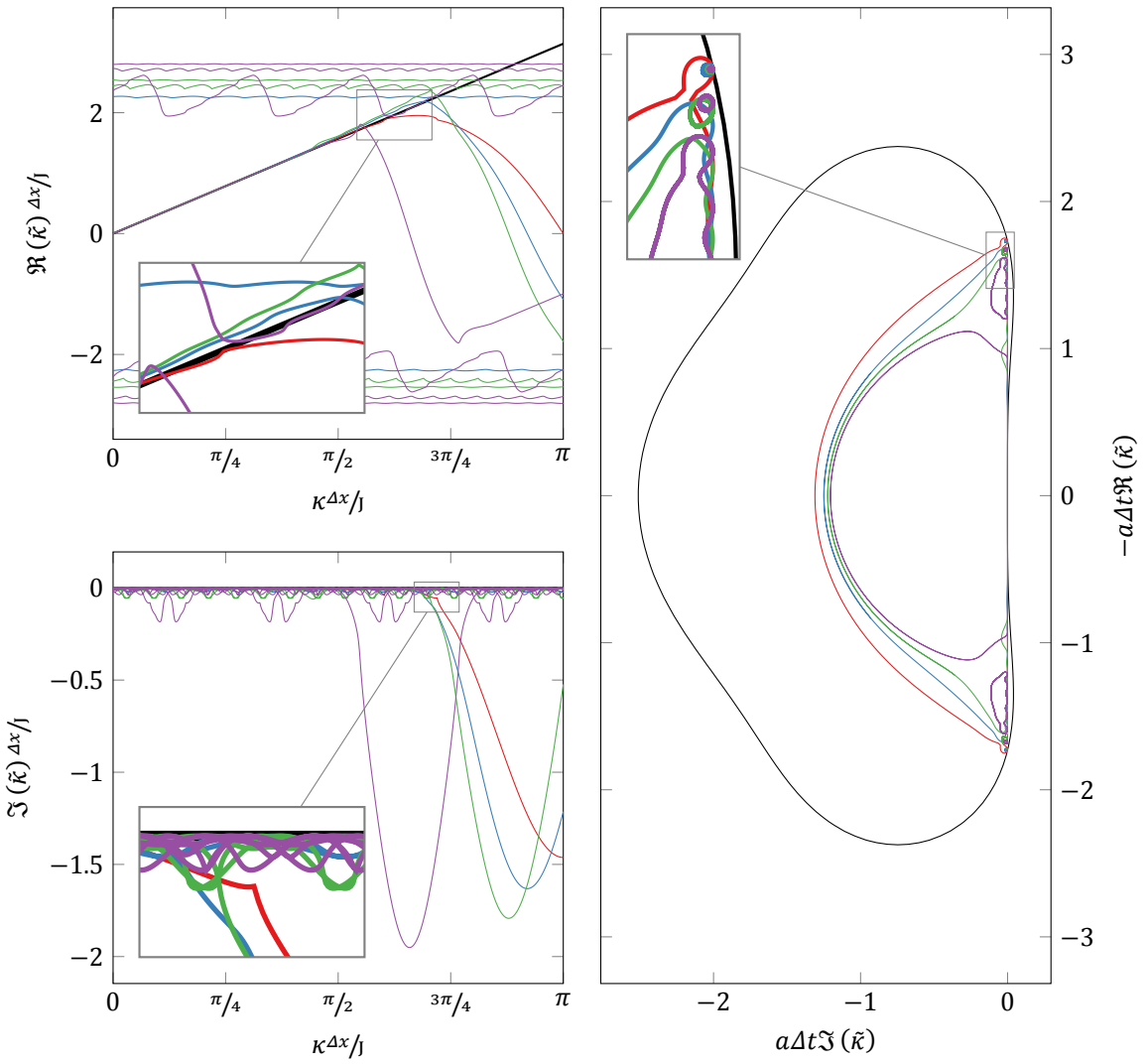


Figure 11.30: Effect of the degree to the modified wavenumbers (physical and “bubble” eigenmodes) in DGIGA, for 8 breakpoint spans per patch and minimal smoothness.

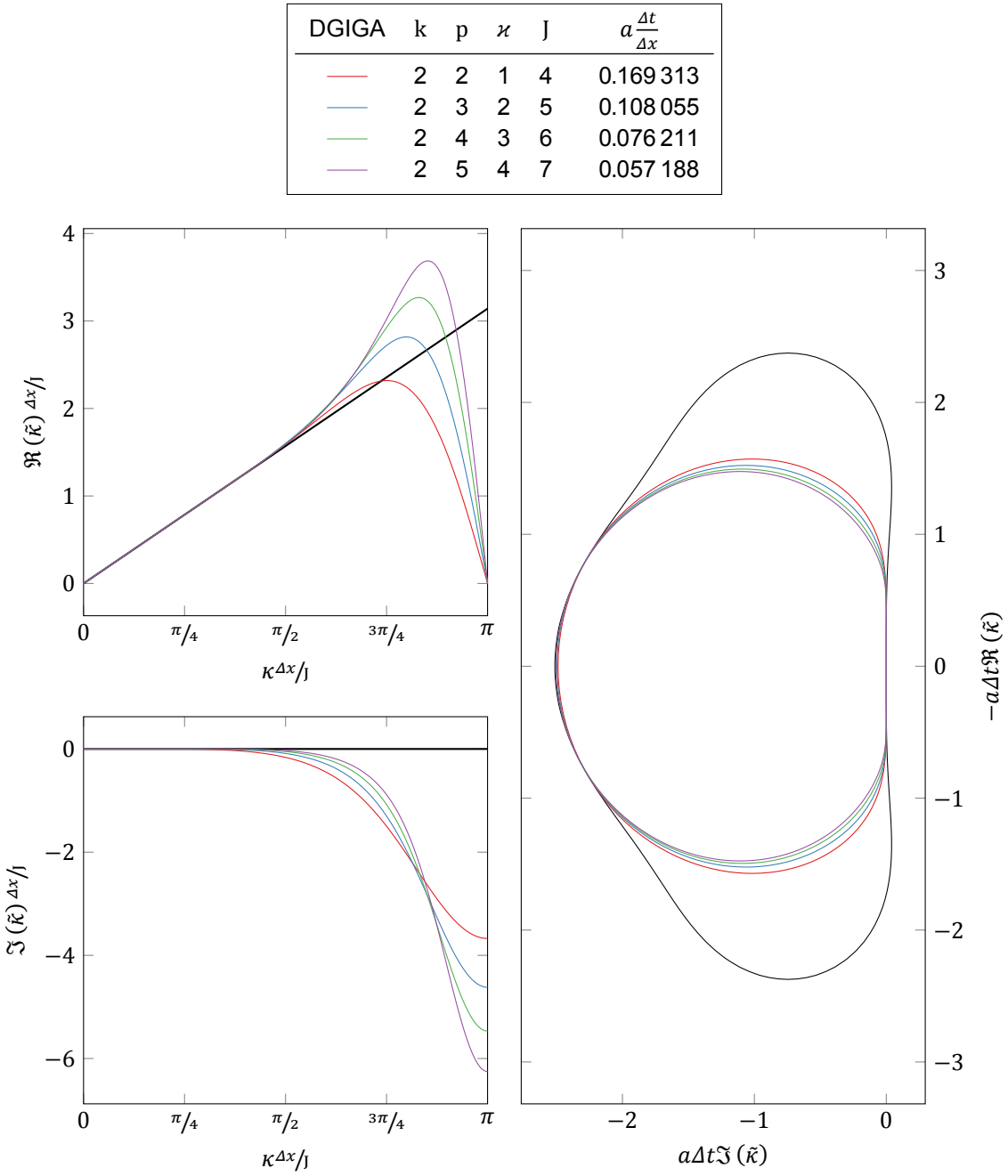


Figure 11.31: Effect of the degree to the modified wavenumbers (physical eigenmode) in DGIGA, for 2 breakpoint spans per patch and maximal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	4	2	1	9	0.114 097
	4	3	2	13	0.064 463
	4	4	3	17	0.042 983
	4	5	4	21	0.031 095

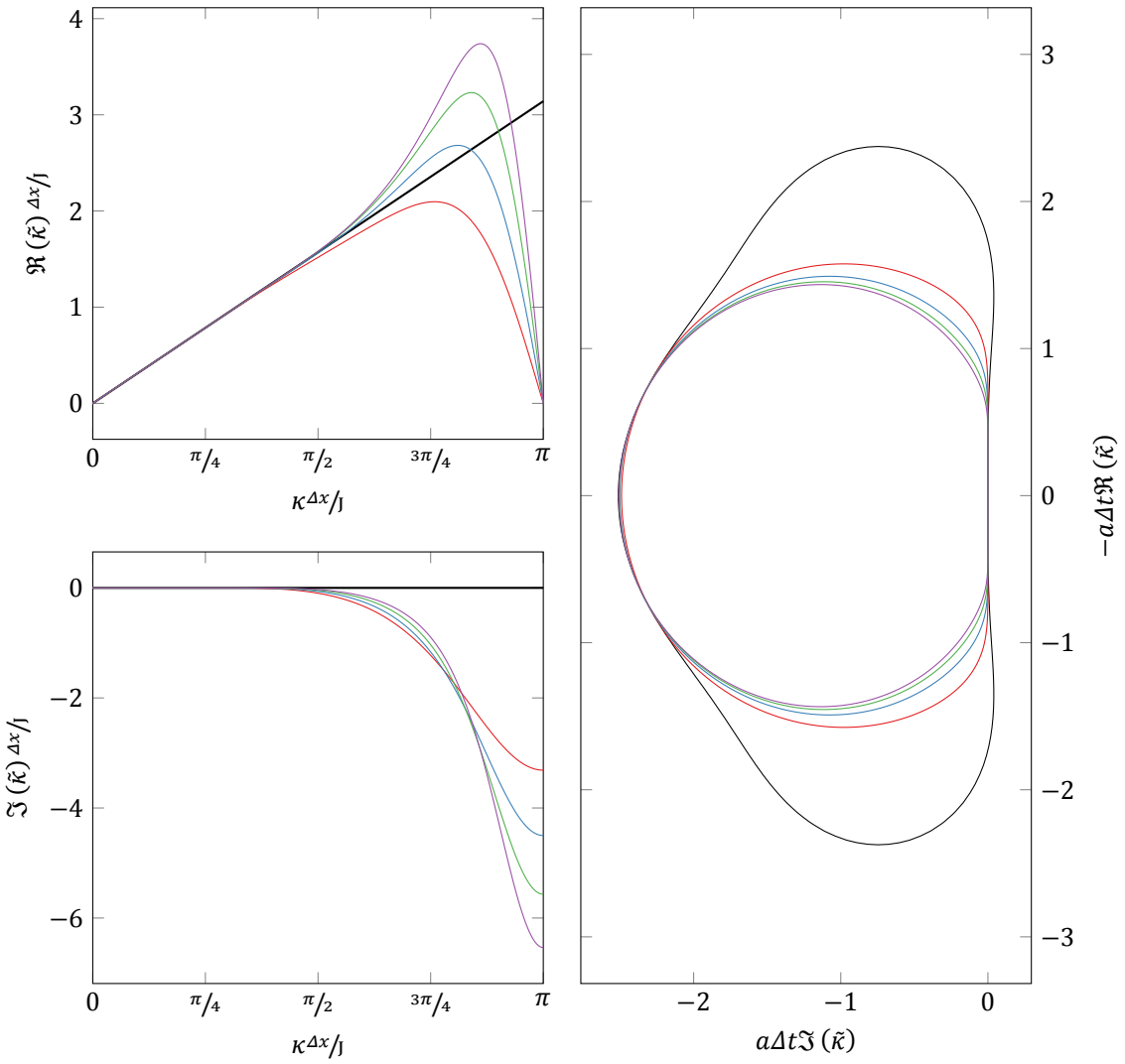


Figure 11.32: Effect of the degree to the modified wavenumbers (physical eigenmode) in DGIGA, for 4 breakpoint spans per patch and maximal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	8	2	1	10	0.094 508
	8	3	2	11	0.058 830
	8	4	3	12	0.040 717
	8	5	4	13	0.030 295

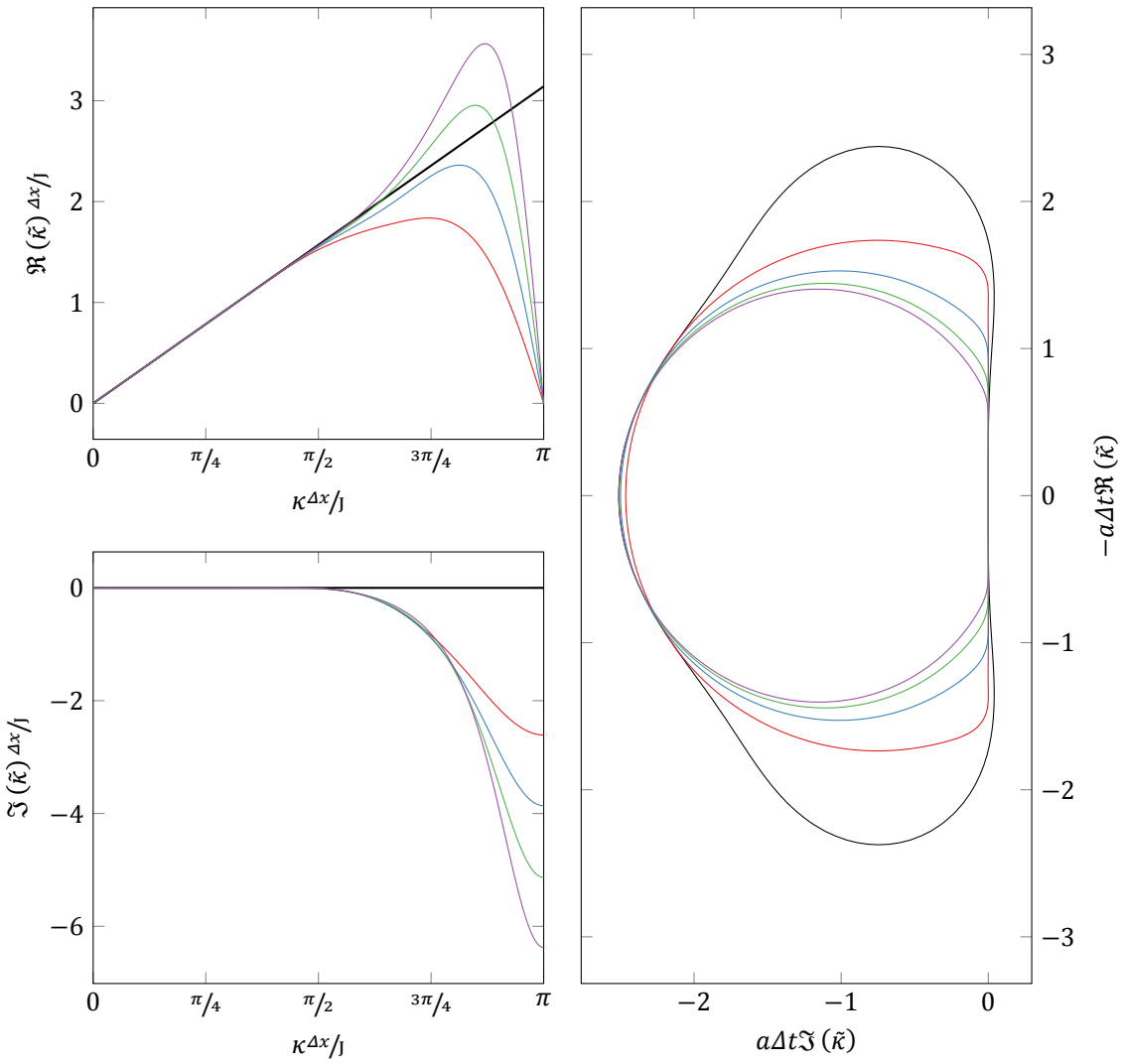


Figure 11.33: Effect of the degree to the modified wavenumbers (physical eigenmode) in DGIGA, for 8 breakpoint spans per patch and maximal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	16	2	1	18	0.053 805
	16	3	2	19	0.045 109
	16	4	3	20	0.031 103
	16	5	4	21	0.022 768

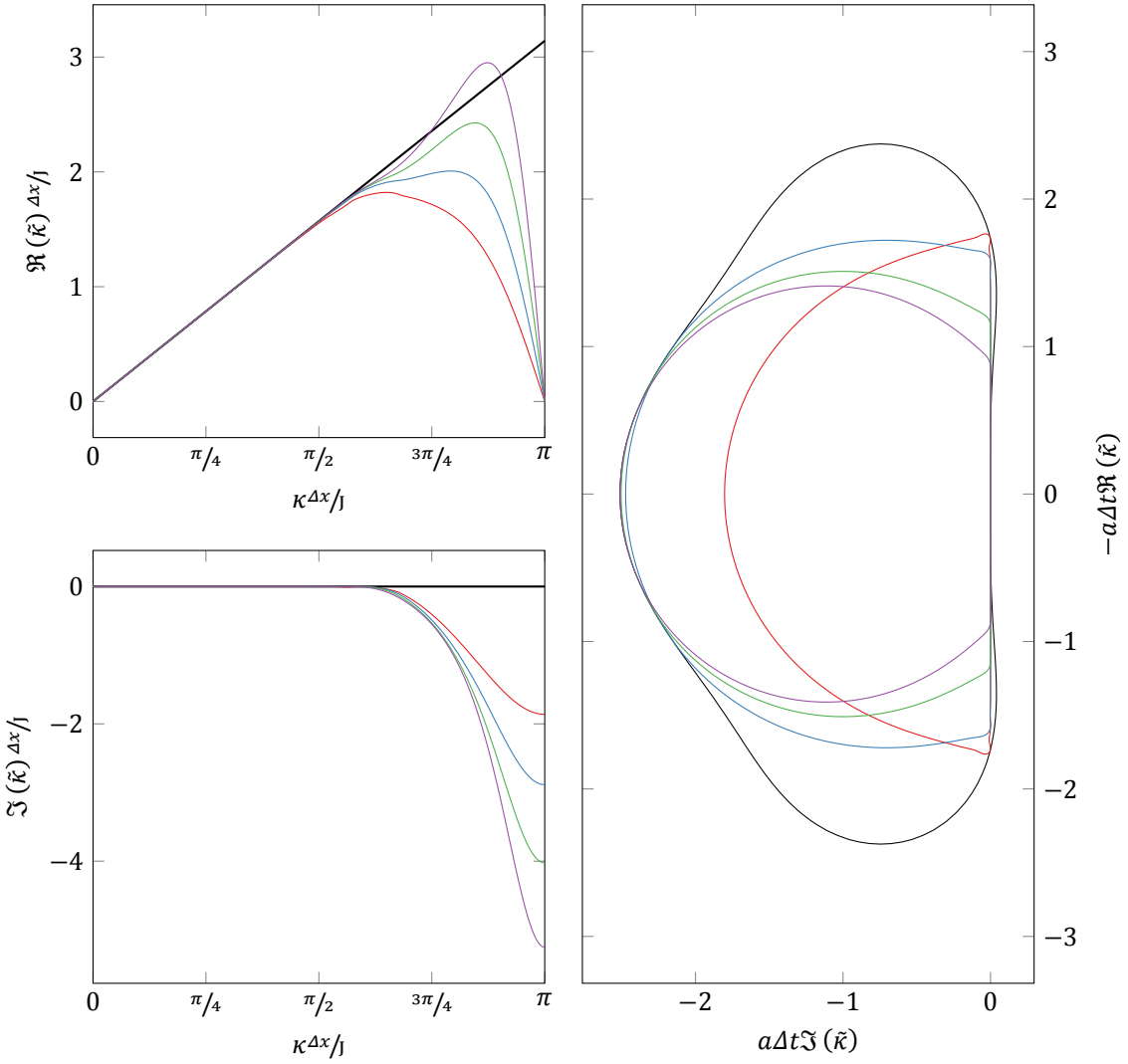


Figure 11.34: Effect of the degree to the modified wavenumbers (physical eigenmode) in DGIGA, for 16 breakpoint spans per patch and maximal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	32	2	1	34	0.026 265
	32	3	2	35	0.023 982
	32	4	3	36	0.022 706
	32	5	4	37	0.017 762

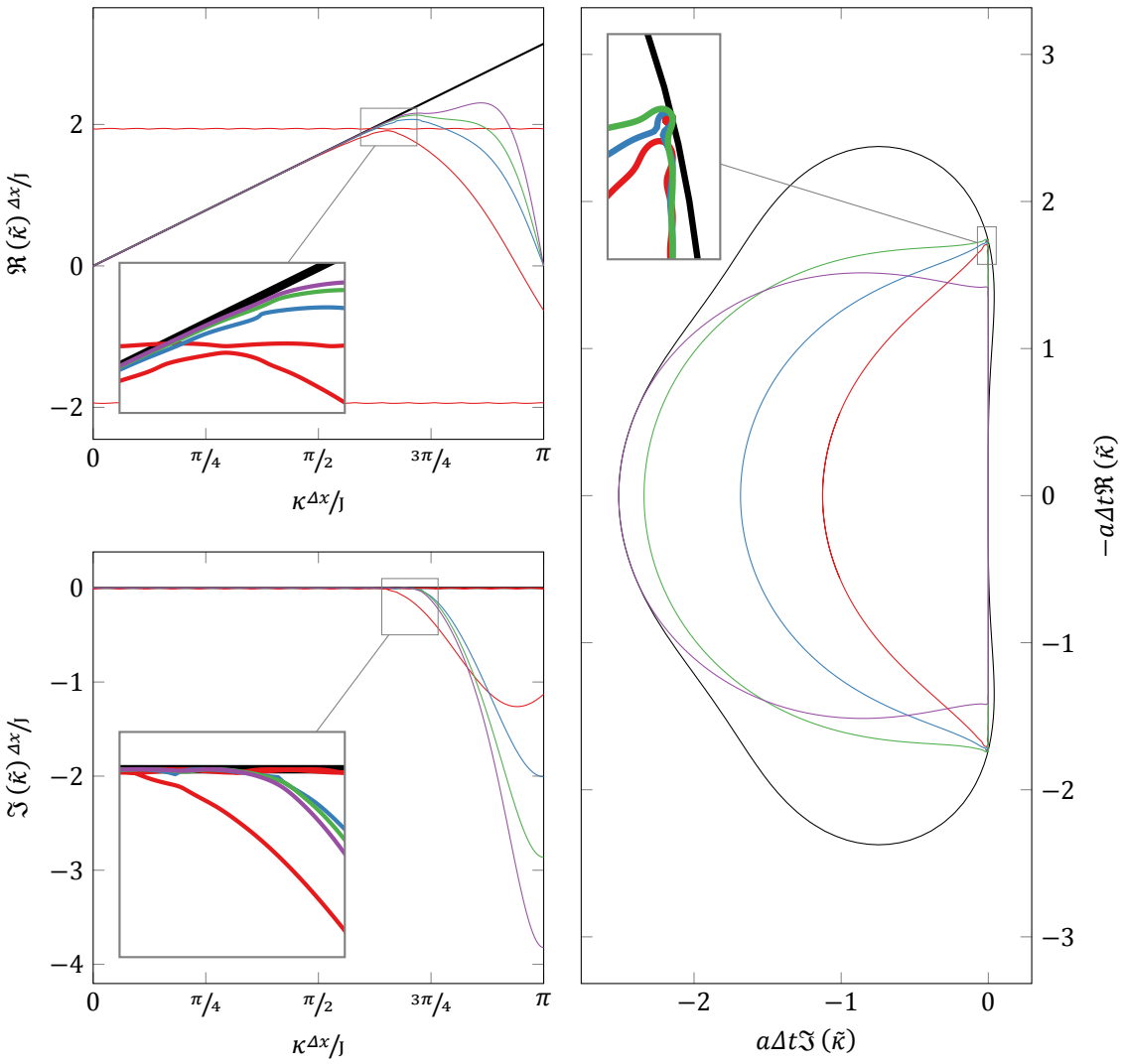


Figure 11.35: Effect of the degree to the modified wavenumbers (physical and “bubble” eigenmodes) in DGIGA, for 32 breakpoint spans per patch and maximal smoothness.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	2	∞	3	0.209 757
	2	2	0	4	0.163 094
	4	2	0	9	0.114 097
	8	2	0	17	0.052 704

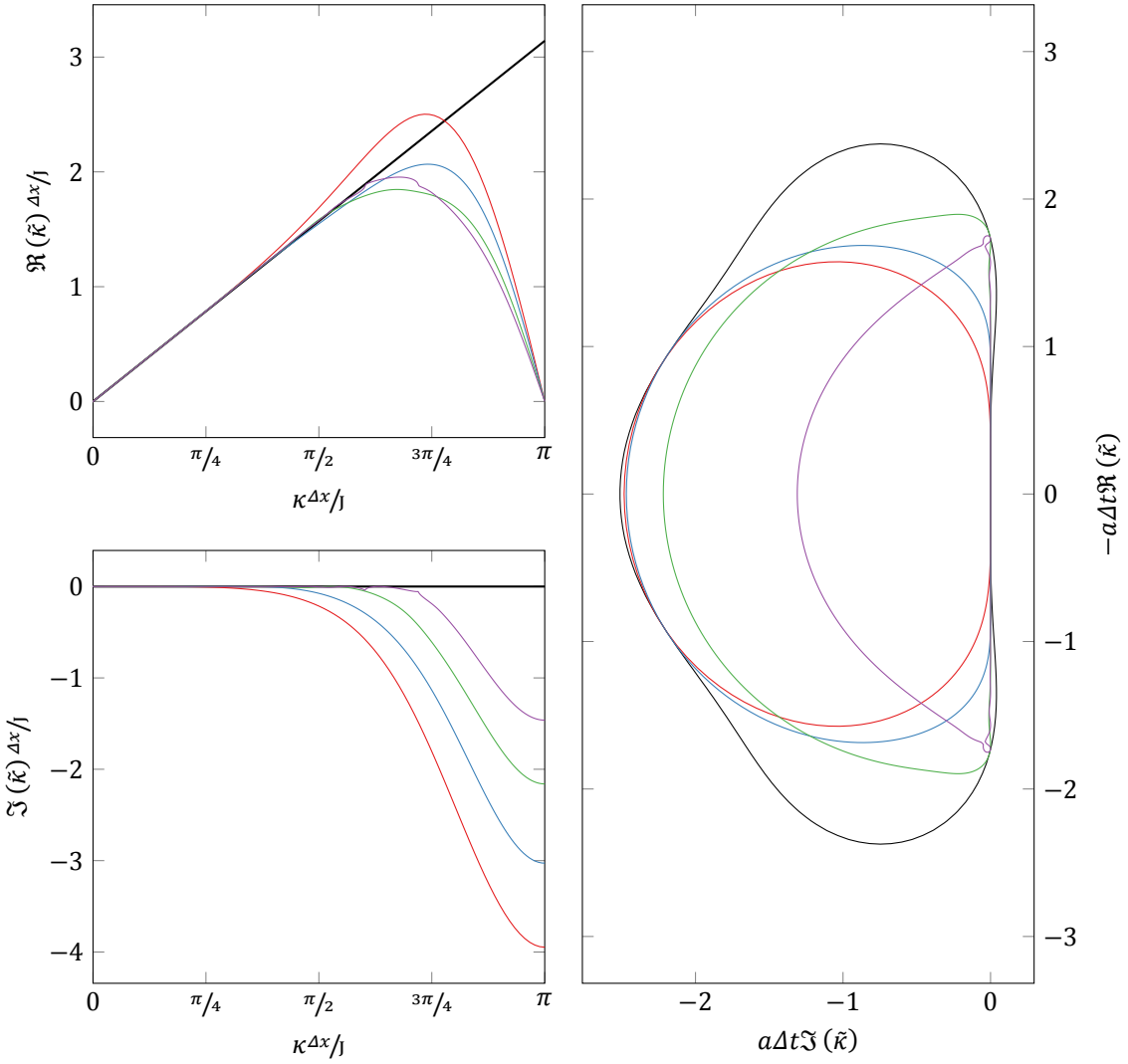


Figure 11.36: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical eigenmode) in 3rd order, C^0, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	3	∞	4	0.130 094
	2	3	0	7	0.101 465
	4	3	0	13	0.064 463
	8	3	0	25	0.030 632

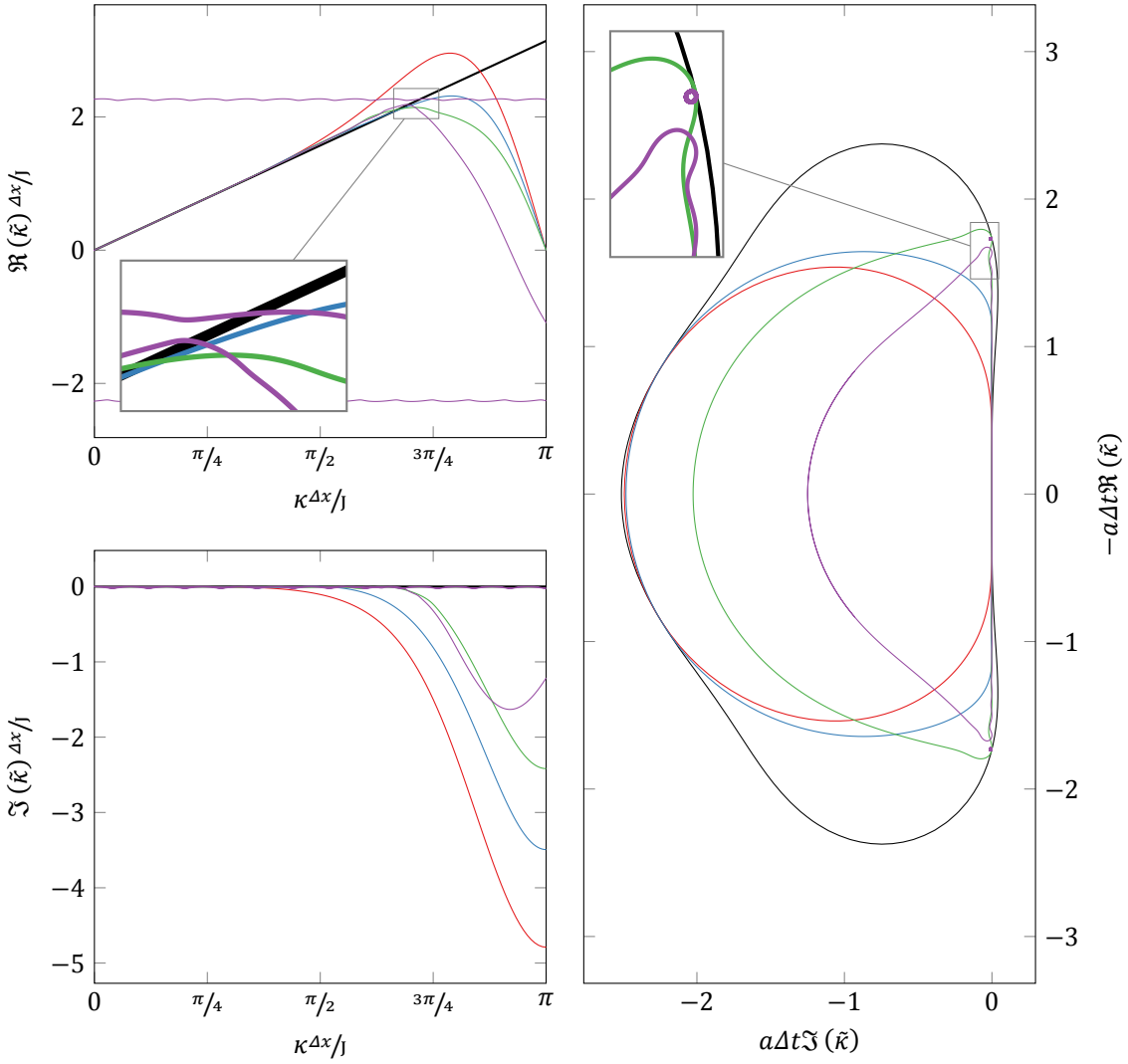


Figure 11.37: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical and "bubble" eigenmodes) in 4th order, C^0, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	4	∞	5	0.089 687
	2	4	0	9	0.070 291
	4	4	0	17	0.042 983
	8	4	0	33	0.020 679

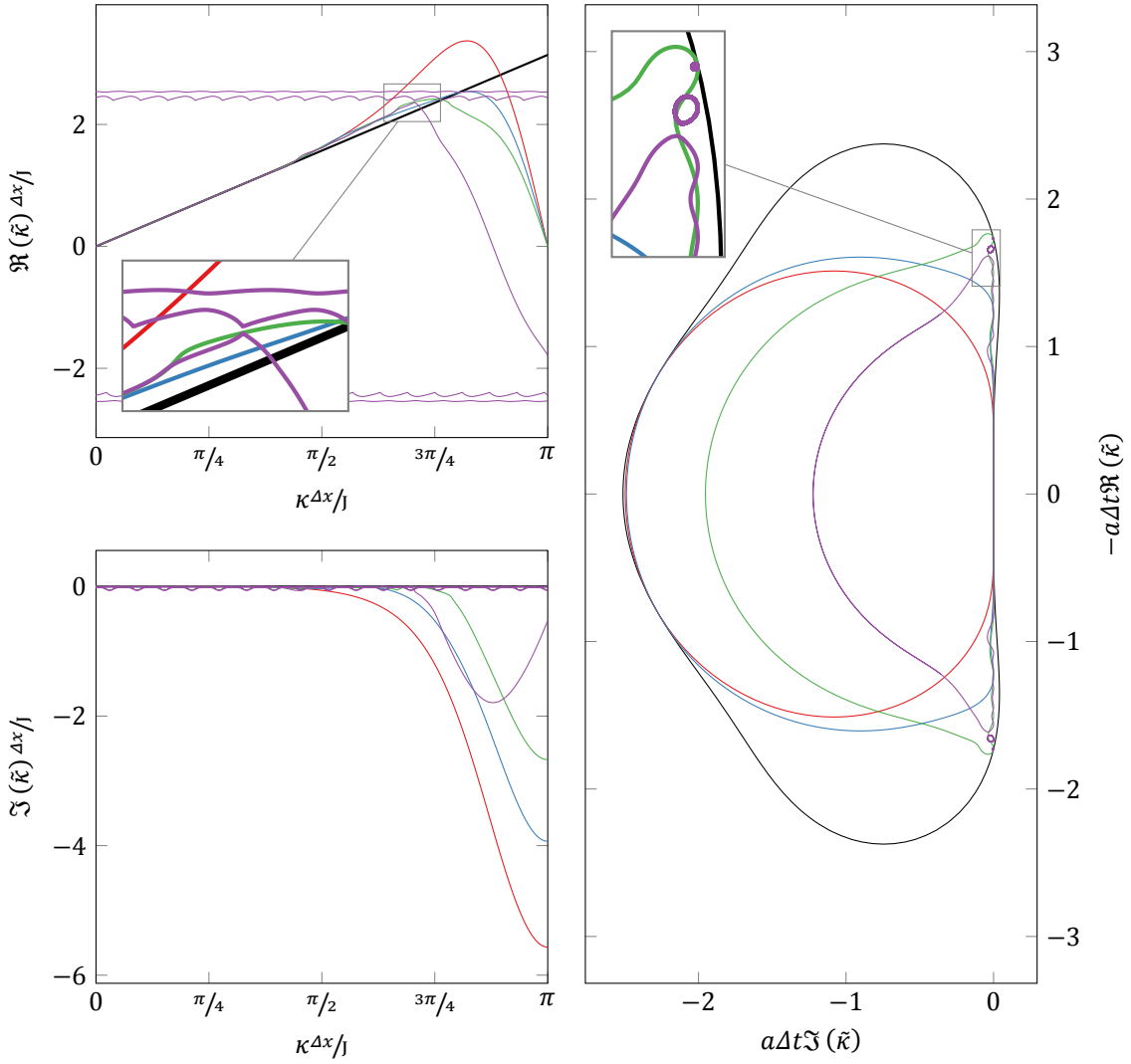


Figure 11.38: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical and "bubble" eigenmodes) in 5th order, C^0, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	5	∞	6	0.066 100
	2	5	0	11	0.052 068
	4	5	0	21	0.031 095
	8	5	0	41	0.015 060

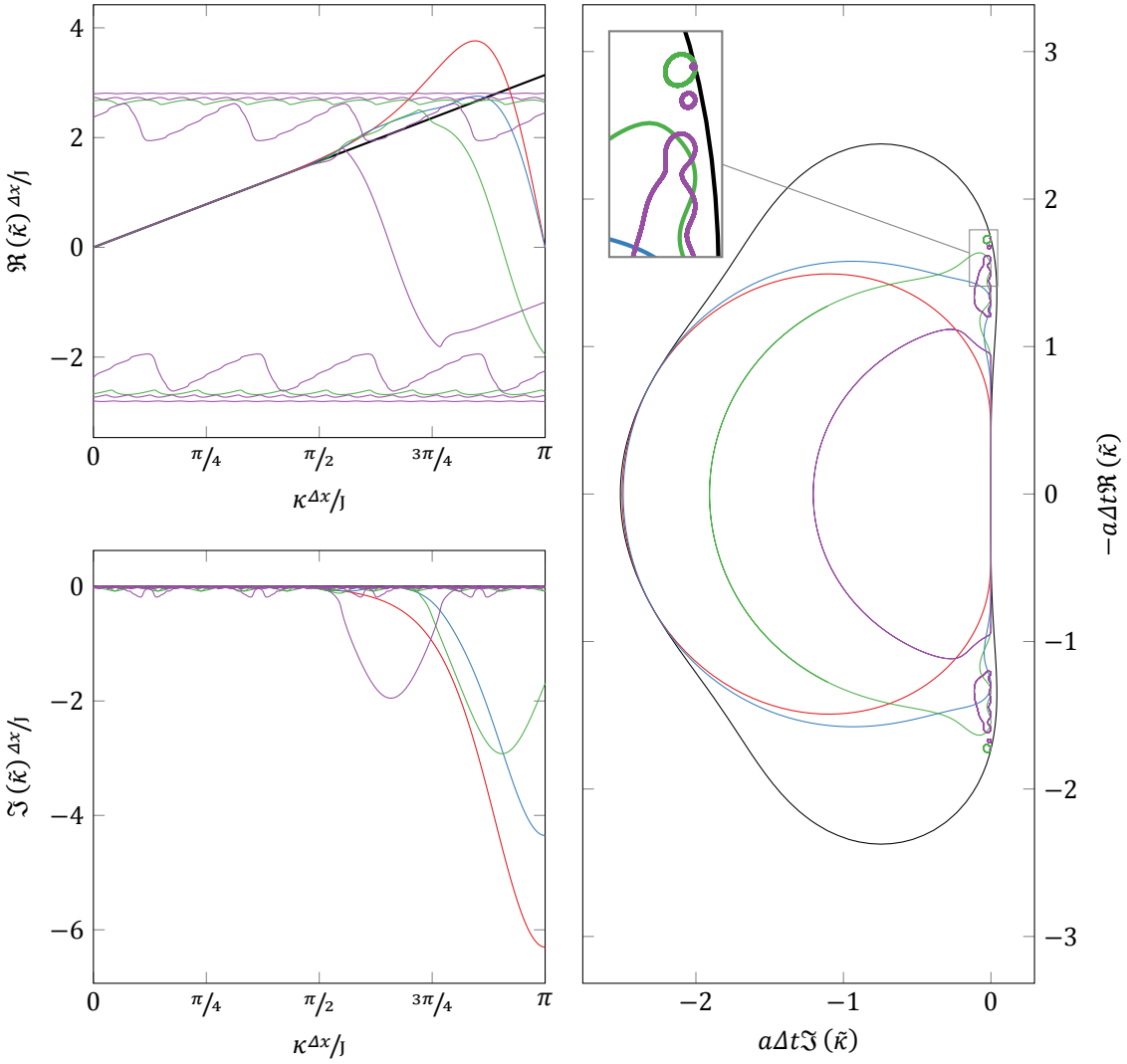


Figure 11.39: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical and "bubble" eigenmodes) in 6th order, C^0, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	2	∞	3	0.209 757
	2	2	1	4	0.169 313
	4	2	1	6	0.125 265
	8	2	1	10	0.094 508
	16	2	1	18	0.053 805
	32	2	1	34	0.026 265

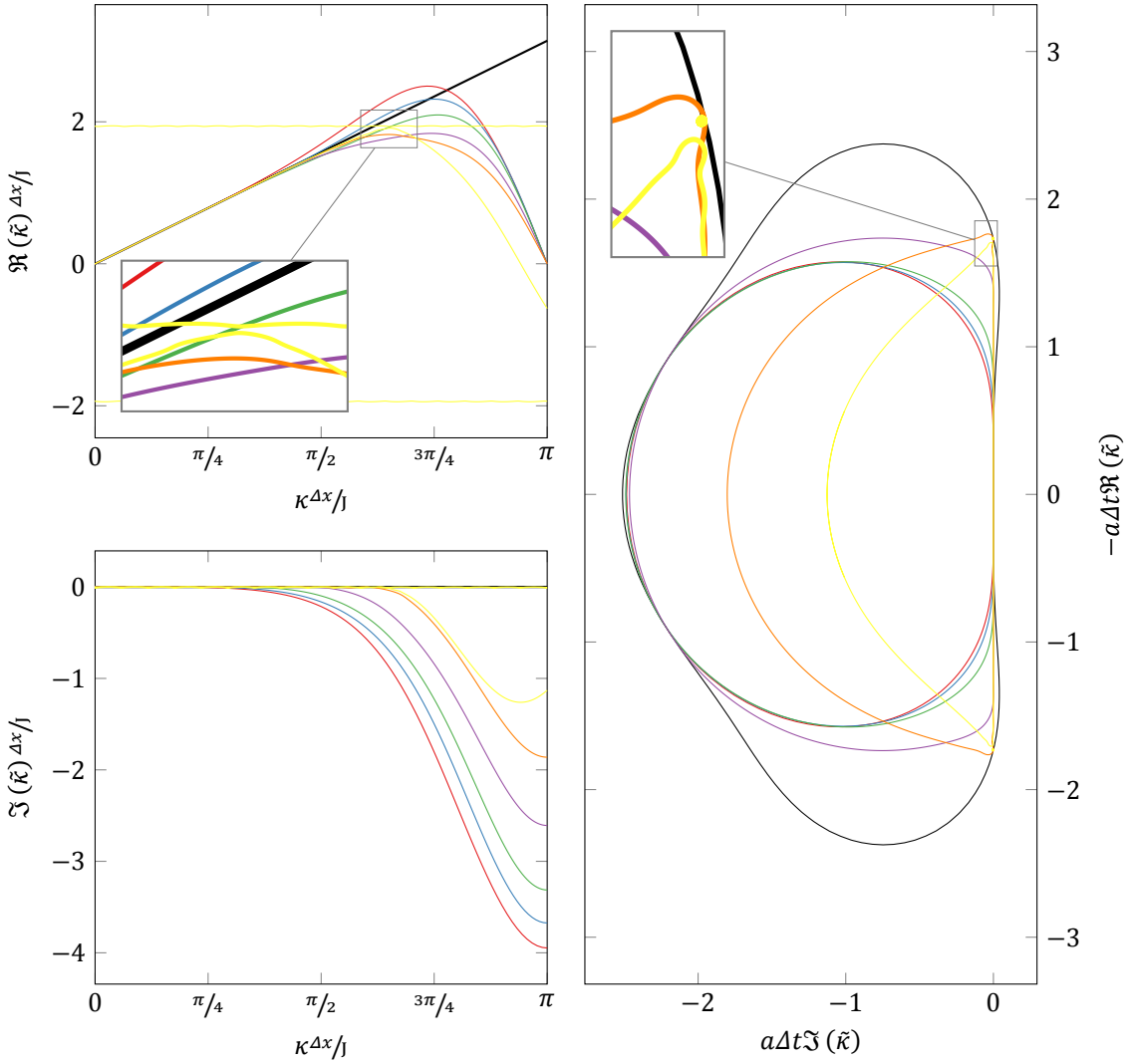


Figure 11.40: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical and "bubble" eigenmodes) in 3rd order, C^1, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	3	∞	4	0.130 094
	2	3	2	5	0.108 055
	4	3	2	7	0.079 422
	8	3	2	11	0.058 830
	16	3	2	19	0.045 109
	32	3	2	35	0.023 982

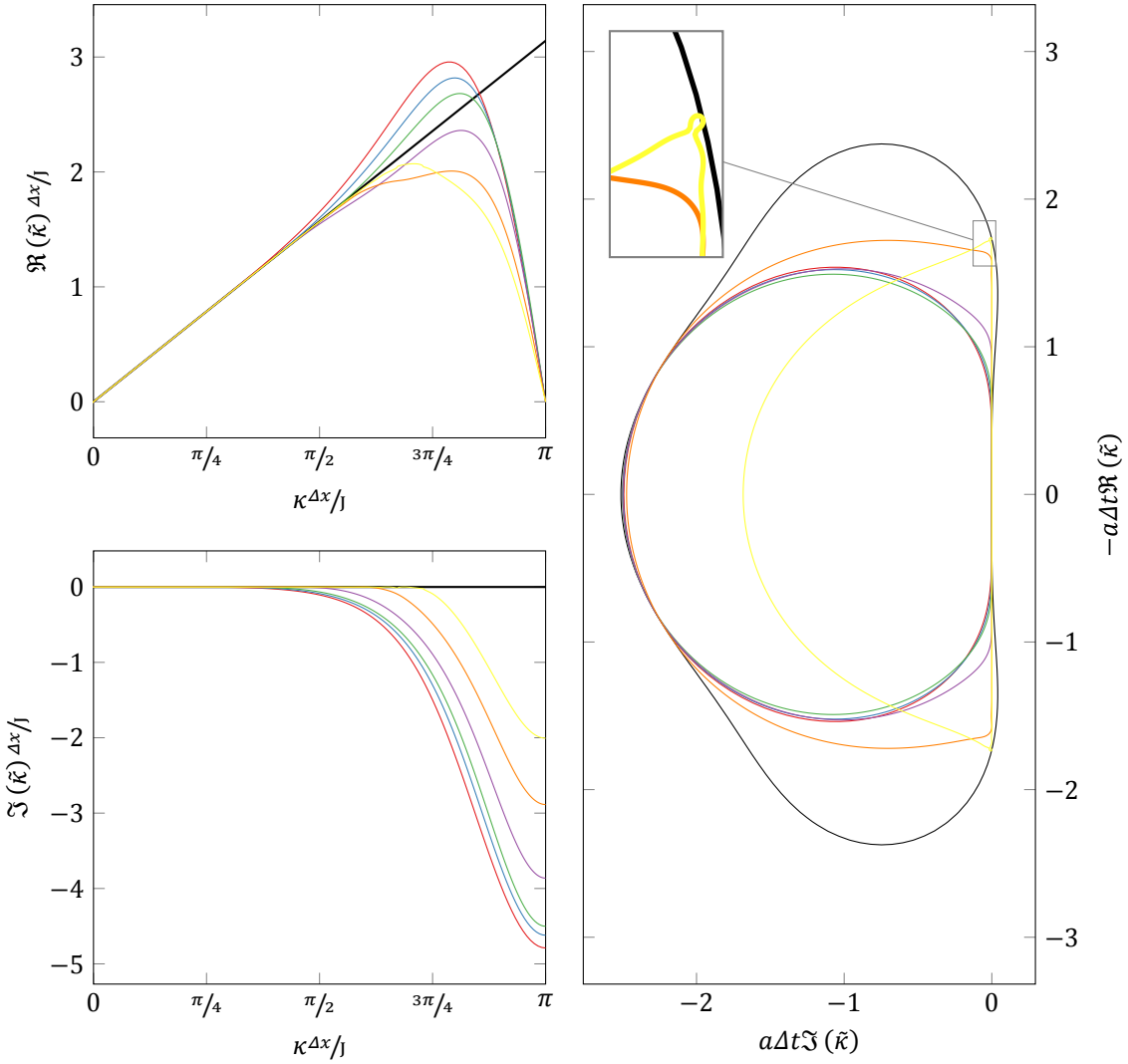


Figure 11.41: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical eigenmode) in 4th order, C^2, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	4	∞	5	0.089 687
	2	4	3	6	0.076 211
	4	4	3	8	0.056 268
	8	4	3	12	0.040 717
	16	4	3	20	0.031 103
	32	4	3	36	0.022 706

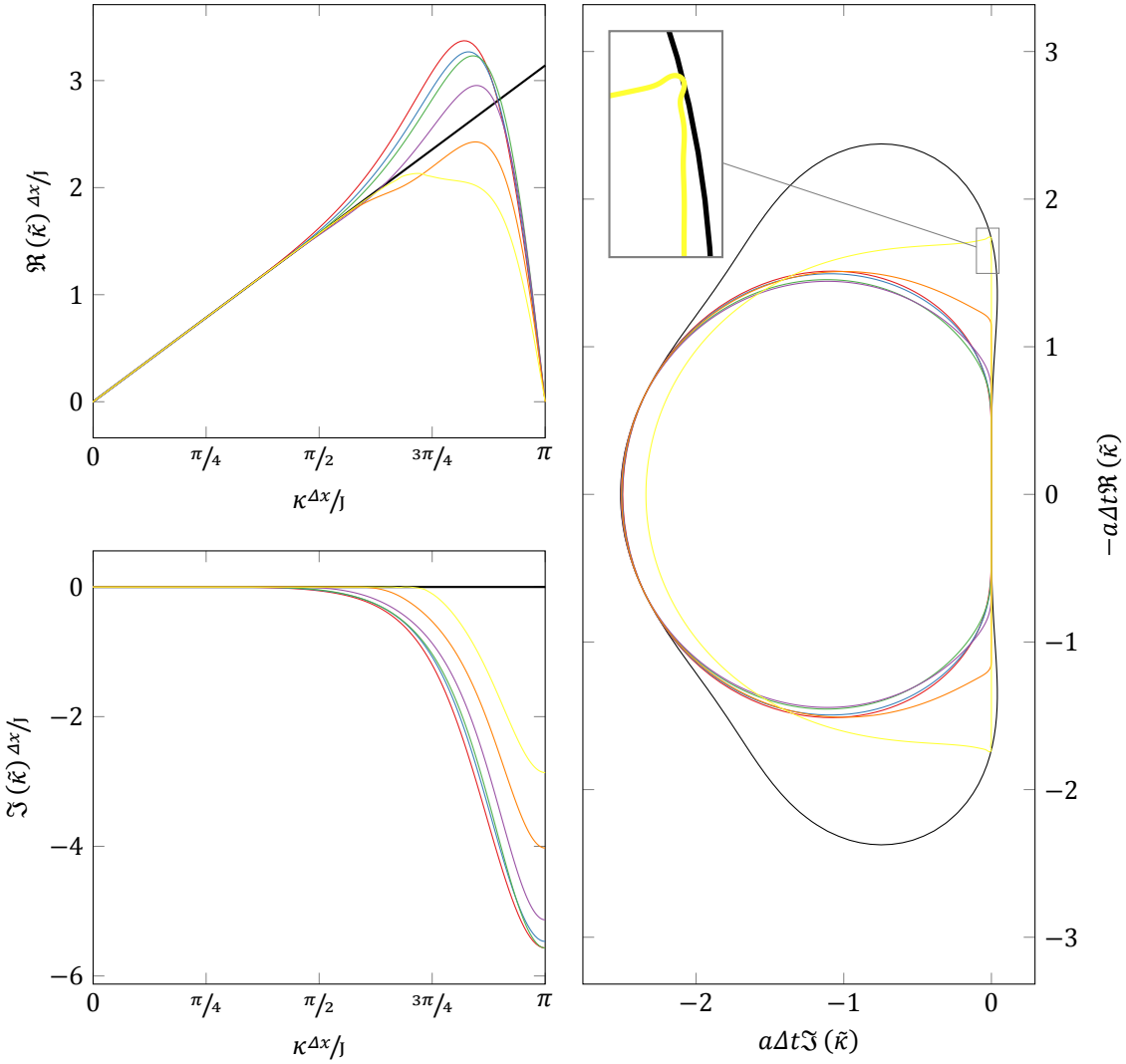


Figure 11.42: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical eigenmode) in 5th order, C^3, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	5	∞	6	0.066 100
	2	5	4	7	0.057 188
	4	5	4	9	0.042 636
	8	5	4	13	0.030 295
	16	5	4	21	0.022 768
	32	5	4	37	0.017 762

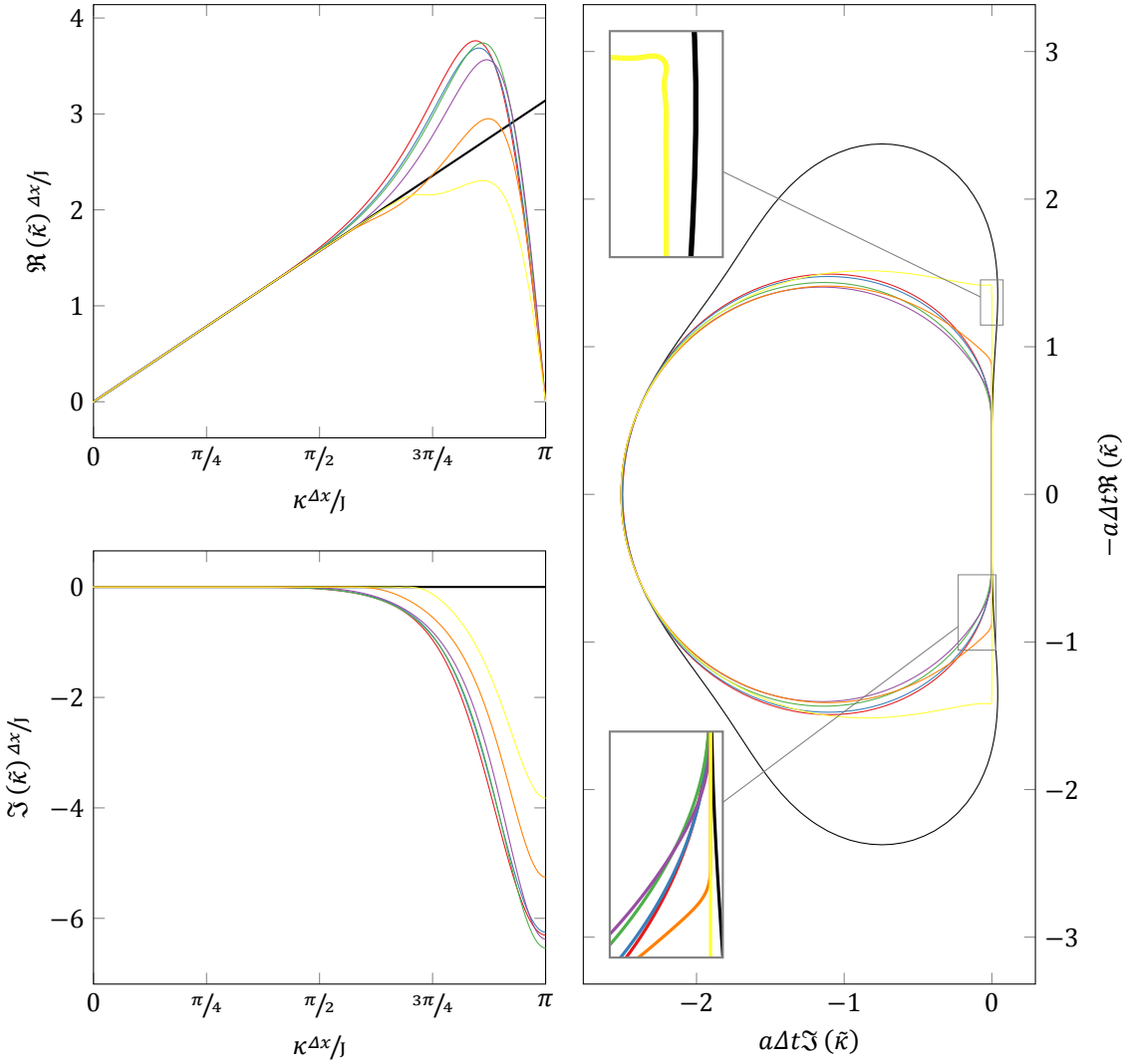


Figure 11.43: Influence of the number of breakpoint spans per patch to the modified wavenumbers (physical eigenmode) in 6th order, C^4, DGIGA.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	2	4	0	9	0.070 291
	2	4	1	8	0.069 147
	2	4	2	7	0.070 619
	2	4	3	6	0.076 211

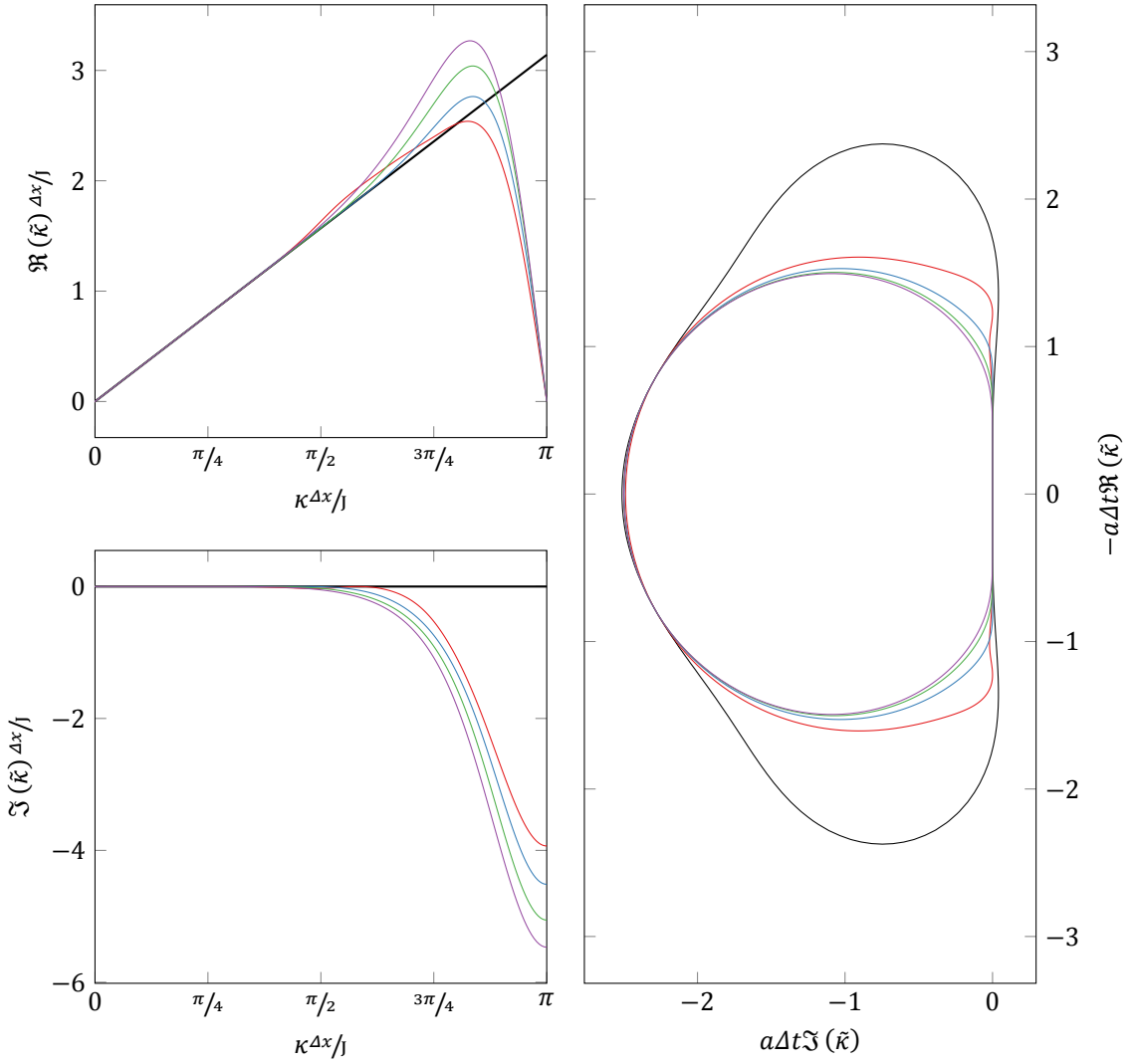


Figure 11.44: Fourth order DGIGA's modified wavenumbers (physical eigenmode), for 2 breakpoint spans per patch and all possible smoothnesses.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	4	4	0	17	0.042 983
	4	4	1	14	0.052 930
	4	4	2	11	0.052 557
	4	4	3	8	0.056 268

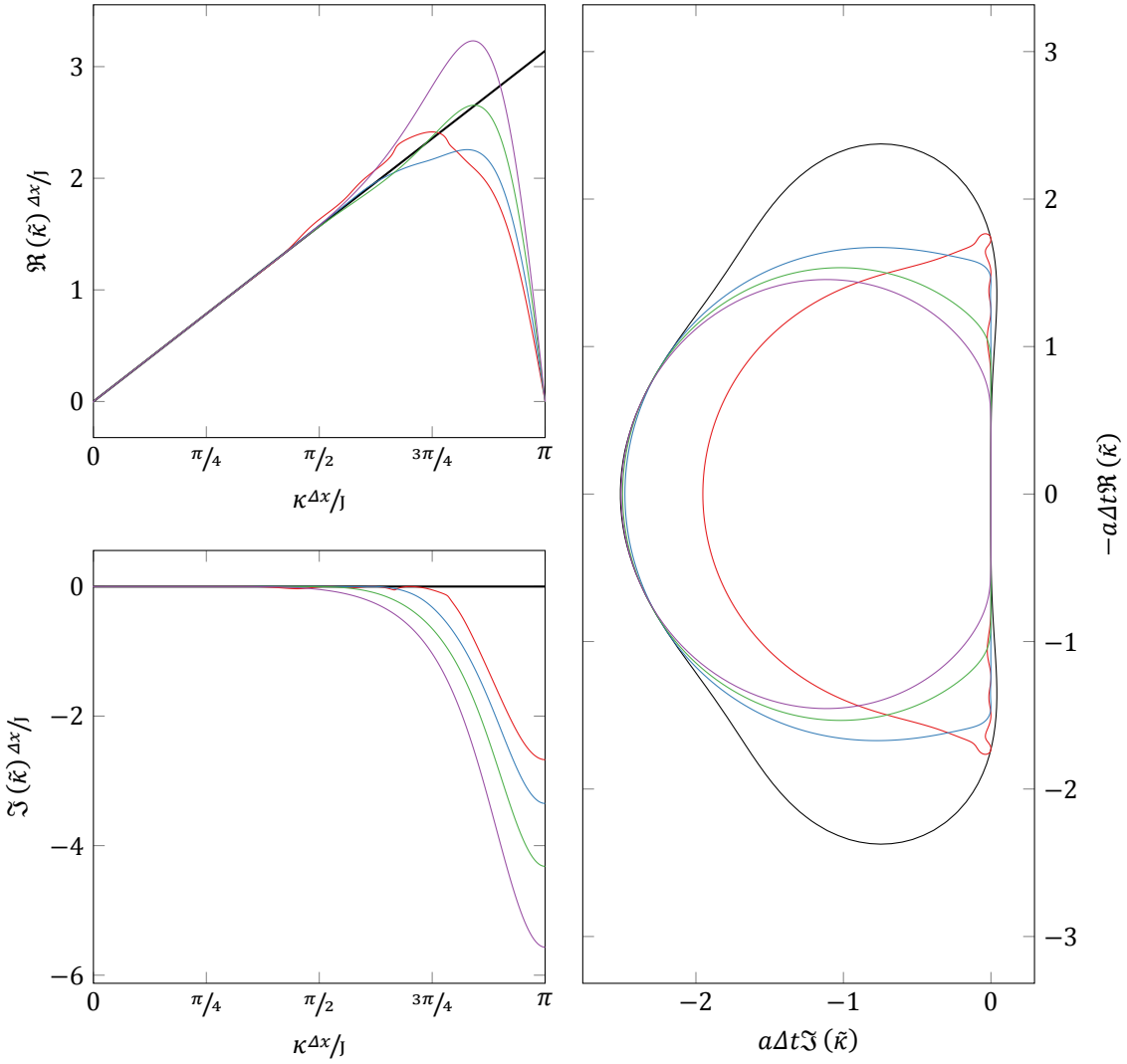


Figure 11.45: Fourth order DGIGA's modified wavenumbers (physical eigenmode), for 4 breakpoint spans per patch and all possible smoothnesses.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	8	4	0	33	0.020 679
	8	4	1	26	0.030 041
	8	4	2	19	0.040 337
	8	4	3	12	0.040 717

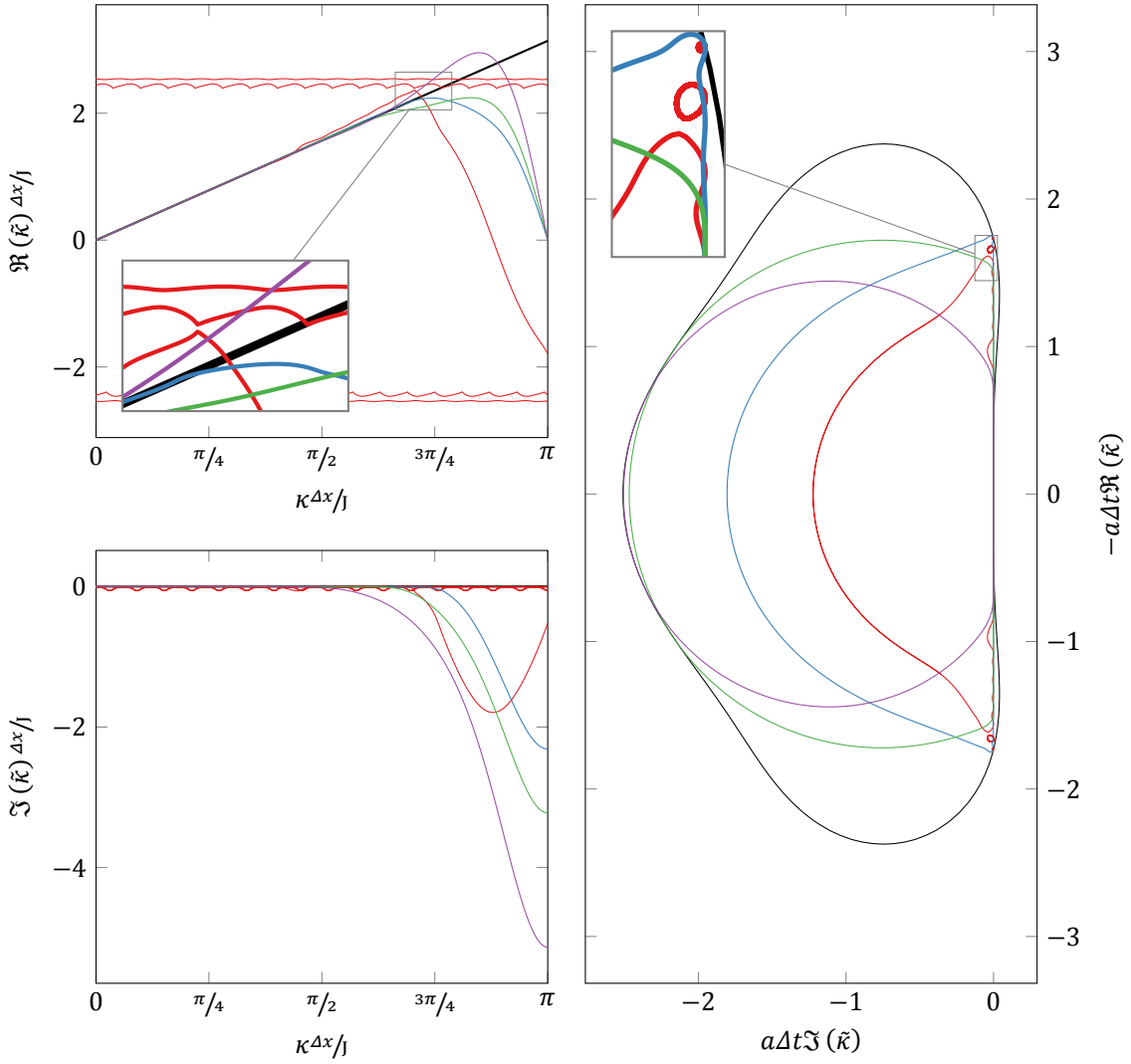


Figure 11.46: Fourth order DGIGA's modified wavenumbers (physical and "bubble" eigenmodes), for 8 breakpoint spans per patch and all possible smoothnesses.

DGIGA	k	p	$\varkappa$	J	$a\frac{\Delta t}{\Delta x}$
	1	21	∞	22	0.007 195
	20	2	1	22	0.042 563
	11	11	10	22	0.008 746
	7	3	0	22	0.035 205

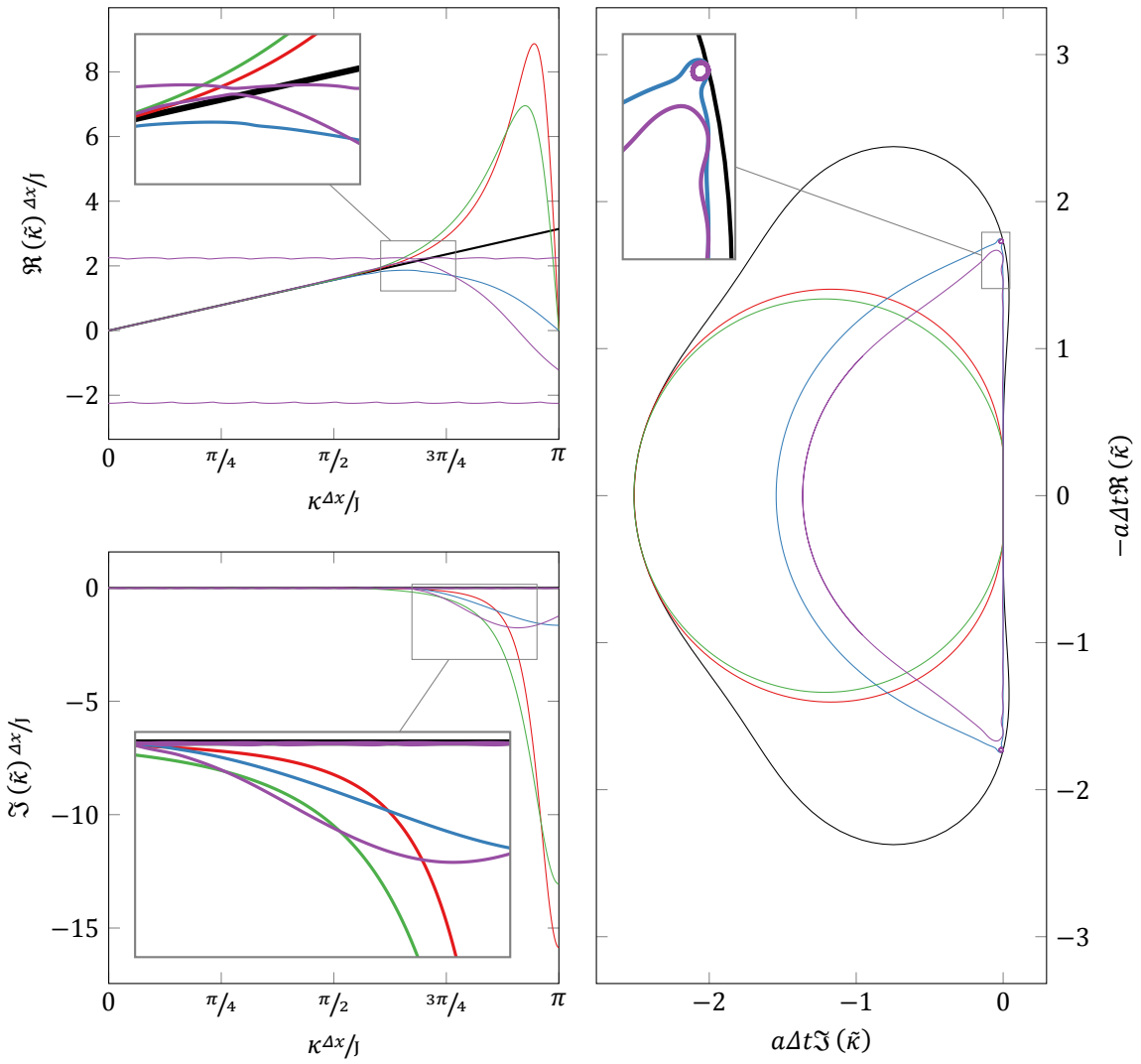


Figure 11.47: Modified wavenumbers (physical and “bubble” eigenmodes) of several combinations of parameters leading to DGIGA discretizations with 22 basis functions per patch.

12

Optimal FR/CPR and DGIGA Configurations

Recall that either DGSEM, FR/CPR or DGIGA possess the freedom to still allow an infinite number of semi-discretization variants each (a full breakdown is given §9.2). The spectral characteristics of the various methods under consideration have been explored in §11, showcasing the effect of each method's parameters in a more qualitative manner. My intention in this section is to go one step further and select the instance of each of these classes of schemes that, in some sense, is best suited to the resolution of turbulent flows.

Turbulent flows are characterized by a broad range of scales coexisting in a flow. In a scale-resolving context (LES or DNS), the goal is to compute the dynamics of a large number of these small-scale features accurately, while simultaneously solving the larger scales as well. The goal of this chapter is to identify promising configurations of the FR/CPR and DGIGA schemes for such applications. I shall do so in in the context of linear advection of a wave of given wavenumber[1], for which it is possible to predict resolution errors (dispersion and dissipation) analytically. Promising configurations of FR/CPR and DGIGA are first identified in §12.1. Then, the following additional aspects of the previously selected methods are studied:

- A priori dispersion and dissipation errors, as a function of time and wavenumber (§12.2)
- Potential suitability for implicit LES (§12.3)
- Estimated computational cost, given a largest-to-smallest resolved wavelength ratio (§12.4)

12.1. Identification of optimal configurations

I choose to treat this selection process as a an optimization problem. Such an approach naturally leads to a quantitative way of comparing any two methods (the value of the objective function), and removes any ambiguity in the selection process. The downside, however, is that the entire responsibility of the selection being meaningful falls on the choice of said objective function. I propose the following one, borrowed from Asthana and Jameson [7], but modified slightly with the use of combined-mode phase shift and amplification factors (see §A.2.5) instead of a weighted average of eigenmodal contributions (akin to that employed in §A.5.4).

12.1.1. Objective function

Consider, as the exact solution, a monochromatic complex-valued wave:

$$q(x,t,\kappa) = \mathrm{e}^{\mathrm{i}\kappa(x-at)}\,, \tag{12.1}$$

and let the following represent dimensionless versions of space, time and wavenumber:

$$x^* \coloneqq \frac{x}{\Delta x}\,, \qquad t^* \coloneqq \frac{a}{\Delta x}t\,, \qquad \kappa^* \coloneqq \kappa\Delta x\,, \tag{12.2}$$

[1] Admittedly, a rather crude approximation to an actual turbulent flow.

so that we may rewrite the previous as:

$$q(x^*, t^*, \kappa^*) = \mathrm{e}^{\mathrm{i}\kappa^*(x^*-t^*)} \,. \tag{12.3}$$

Let us assume, for the sake of this argument, that the numerical solution has a similar form, but is afflicted with:

- A reduced amplitude, $G(t^*, \kappa^*) \in [0, 1]$ (the effect of numerical dissipation)
- A modified phase angle $\Psi(t^*, \kappa^*) \coloneqq \kappa^* t^* - \Delta\Psi(t^*, \kappa^*)$, $\Delta\Psi$ being the phase angle by which the approximate solution is lagging behind the exact one (as a consequence of numerical dispersion)

These two quantities are defined by (A.36) and (A.35), respectively, for any specific semi-discretization and wavenumber as functions of t^*. The result is:

$$q^h(x^*, t^*, \kappa^*) = G(t^*, \kappa^*)\mathrm{e}^{\mathrm{i}(\kappa^* x^* - \Psi(t^*, \kappa^*))} \,. \tag{12.4}$$

After some simple algebra[2], the magnitude of the *combined-mode spectral error* between exact and numerical solutions, at any position x, simplifies to:

$$\left|q - q^h\right| = \left|1 - G(t^*, \kappa^*)\mathrm{e}^{\mathrm{i}\Delta\Psi(t^*, \kappa^*)}\right| \,. \tag{12.5}$$

From (12.5), I define an objective function, $F \colon \mathbb{R} \to \mathbb{R}$, as:

$$F(t^*) \coloneqq \frac{1}{\mathrm{J}\pi} \int_0^{\mathrm{J}\pi} \left|1 - G(t^*, \kappa^*)\mathrm{e}^{\mathrm{i}\Delta\Psi(t^*, \kappa^*)}\right| \mathrm{d}\kappa^* \,, \tag{12.6}$$

i.e. the L^1 norm of (12.5), taken over all (positive) wavenumbers up to the Nyquist limit (see §A.1 for more details), which I evaluate at a very large time, $t^* = 100$ (the time necessary for the exact solution to travel across 100 elements or patches), imitating [7]. I do so via MATLAB's built-in adaptive quadrature routine[3], with default tolerances (1×10^{-10} absolute, 1×10^{-6} relative).

For DGSEM, I have observed (not shown) that (12.5) is almost identical to the integrand used in [7], provided that $t^* \gg 1$. Regardless, based on [5, 117], I would argue that (12.6) can be expected to be more (or, at least, equally) representative of the spectral error of any given scheme, in general, than the aforementioned.

12.1.2. Optimization problems

An optimization problem can then be defined for each method, at a given number of degrees of freedom per patch or element, J. In the FR/CPR case, it is simply:

$$\min_{\eta} \qquad F(100) \,, \tag{12.7a}$$

$$\text{subject to} \qquad -1 < \eta \,, \tag{12.7b}$$

where the constraint on η restricts the search to linearly stable schemes. To solve (12.7), I use MATLAB's single-variable minimization over an interval built-in routine[4], with default settings and $\eta \leq 5$ as upper bound.

With DGIGA, in turn, I tackle the slightly more complicated:

$$\min_{\mathrm{k},\mathrm{p},\varkappa} \qquad F(100) \,, \tag{12.8a}$$

$$\text{subject to} \qquad 1 \leq \mathrm{k} \,, \tag{12.8b}$$

$$1 \leq \mathrm{p} \leq \mathrm{J} - 1 \,, \tag{12.8c}$$

$$0 \leq \varkappa \leq \mathrm{p} \,, \tag{12.8d}$$

$$\mathrm{J} = \mathrm{kp} + (1 - \mathrm{k})\varkappa + 1 \,. \tag{12.8e}$$

This is a discrete (nonlinear) optimization, since $\mathrm{k}, \mathrm{p}, \varkappa \in \mathbb{Z}$. To solve it, I resort to evaluating (12.6) for every single combination of these three parameters that satisfy the indicated constraints; each one of such combinations represents a candidate method, and that with the lowest $F(100)$ value is optimal.

[2] In particular, note that $\left|\mathrm{e}^{\mathrm{i}x}\right| = 1$ for any $x \in \mathbb{R}$.

[3] https://nl.mathworks.com/help/matlab/ref/integral.html

[4] https://nl.mathworks.com/help/matlab/ref/fminbnd.html

12.1.3. Optimization results and discussion

The results of the optimization problems (12.7) and (12.8), for a selection of J, are detailed in tables 12.2 and 12.3, respectively. Additionally, table 12.4 complements the latter by including all configurations (not only optimal ones) evaluated when solving (12.8). For added clarity, I report the characteristics of the optimized methods in terms of changes over the DGSEM scheme of equal J, which I shall consider the baseline methods; these are detailed in table 12.1. Besides the "badness" (lower is better) measure given by (12.6), I shall also judge the quality of each optimal semi-discretization through the following magnitudes (all of them are defined and detailed in appendix A):

- Theoretical order of convergence, A_T (A.65).
- Highest well-resolved wavenumber (dispersion plus dissipation, 1 % threshold), κ_f (A.66).
- Cutoff wavenumber (dissipation only, 1 % threshold), $\kappa_{1\%}$ (A.68).

In addition, the maximum allowable Courant number (as well as the condition number of its mass matrix, in the DGIGA case) of each optimum is also included in these results.

Table 12.2 shows that, for every J tested, the optimal FR/CPR discretization fails to achieve barely any improvement over DGSEM in terms of $F(100)$. Furthermore, all metrics other than linear stability indicate a slight *decrease* in performance with respect to the baseline, while the increase in maximum allowable Courant number—between 2 % and 8 %—is rather moderate in comparison to that obtained if optimizing FR/CPR for stability, which yields improvements in the 100 % to 200 % range [120] (results which I have verified, even though I do not show them). All in all, these results suggest that *FR/CPR methods of the VCJH class offer little to no intrinsic advantage over DGSEM* in terms of reduced spectral errors, theoretical order of convergence, resolution efficiency and stability. The same conclusion is reached in [7], although I deem it worth mentioning that I could not reproduce the improvements reported there for FR/CPR (my results seem to be more conservative).

For DGIGA (table 12.3), the three lowest J considered behave in the same way as just described for FR/CPR: the optimum corresponds to the scheme which reduces to DGSEM—that is, Bernstein-based DG (DGIGA with Bsplines made up of a single polynomial segment of degree $\mathrm{J}-1$). This is not surprising, as it results in the highest possible polynomial degree for a given number of degrees of freedom. Interestingly, however, this is not longer optimal for $\mathrm{J}=6$ and above[5]. From then on, all optima (up until $\mathrm{J}=20$, at least) correspond to bases that employ precisely two breakpoint spans per patch ($\mathrm{k}=2$). Moreover, and in contrast to FR/CPR, the optimized DGIGA schemes achieve significant improvements, 3 % to 12 % in $F(100)$ and 50 % to 30 % in $\varsigma_{\max}$, while *simultaneously* increasing (albeit only moderately) their resolving efficiency.

There is a considerable loss in theoretical order of accuracy, reaching up to 23 % (for which the optimum is of degree $\mathrm{p}=7$, in contrast to the baseline's $\mathrm{p}=10$). Although substantial, this reduction can be easily attributed to the lower B-spline degree. In fact, it is quite remarkable that *higher* spectral accuracy and resolving efficiency can be achieved with a significantly lower degree. This reduction of p for a fixed J, leads to the conditioning of the B-spline basis being greatly improved; additionally, the bandwidth of the gradient and residual matrices (which only depends on p for both DGSEM and DGIGA) is thus lower in the optimum than in the baseline. The fact that $\mathrm{k}=2$ is optimal for all tested $\mathrm{J}>5$ could be due to it representing the minimum number of breakpoints (a single one) across which the smoothness of the basis, $C^{\varkappa}$, can be varied—any additional breakpoint would force an even lower degree (since J needs to be preserved), while one breakpoint fewer would result in C^{∞} smoothness within the patch (and, hence, no possibility of balancing degree and smoothness). Table 12.4 shows that, for sufficiently high J, there are still multiple DGIGA configurations superior to the baseline, some of them only slightly worse than the optimum. It is possible, therefore, to exploit the benefits of a reduced B-spline degree even further, should it be necessary or desirable to do so, by selecting a slightly less optimal basis of even lower degree.

12.2. Combined-mode dispersion and dissipation errors

Complementing the error norm that is (12.6), figures 12.1 to 12.5 show the semi-discrete combined-mode behavior in terms of phase and amplitude errors (between the result of using a given scheme and

[5] I have verified that Bernstein-based DGIGA and DGSEM give almost identical results for all magnitudes shown in the tables, up to $\mathrm{J}=20$. This means that the ill-conditioning of the former is not sufficient to compromise the optimization procedure (Bernstein is the worst-conditioned of all DGIGA bases of a given J).

Table 12.1: Comparison, according to several criteria (higher is better), between DGSEM schemes of varying degree. The relative increase in "goodness" (negated objective function value) is evaluated with respect to that of the first row, $\mathrm{p} = 0$ (first order upwind FVM).

J	p	$-F(100)$	$-\frac{\Delta F(100)}{F(100)}$	A_T	$\frac{\Delta x}{\mathrm{J}\pi}\kappa_f$	$\frac{\Delta x}{\mathrm{J}\pi}\kappa_{1\%}$	$\varsigma_{\max}$
1	0	-0.96	$+0$	1	0.006	0.045	1.256
2	1	-0.867	$+0.097$	2.999	0.145	0.179	0.41
3	2	-0.796	$+0.171$	4.999	0.263	0.278	0.21
4	3	-0.744	$+0.225$	6.988	0.34	0.345	0.13
5	4	-0.706	$+0.265$	8.962	0.392	0.392	0.09
6	5	-0.675	$+0.297$	10.948	0.429	0.428	0.066
8	7	-0.631	$+0.342$	14.787	0.479	0.478	0.041
11	10	-0.589	$+0.386$	20.348	0.523	0.524	0.024
15	14	-0.553	$+0.425$	28.452	0.557	0.561	0.014
20	19	-0.523	$+0.455$	34.08	0.581	0.589	0.008

Table 12.2: FR/CPR semi-discretizations that, for each number of basis functions per element (J), minimize the objective function (12.6). The only free parameter is η (or, equivalently, c). Relative changes are increments over the DGSEM method of the same degree (table 12.1); in all magnitudes, a positive increment represents an improvement.

J	p	η	c	$-\frac{\Delta F(100)}{F(100)}$	$\frac{\Delta A_T}{A_T}$	$\frac{\Delta \kappa_f}{\kappa_f}$	$\frac{\Delta \kappa_{1\%}}{\kappa_{1\%}}$	$\frac{\Delta \varsigma_{\max}}{\varsigma_{\max}}$
3	2	0.023	0.001	$+0$	-0.029	-0.002	-0.007	$+0.019$
4	3	0.035	$4.457 \cdot 10^{-5}$	$+0.001$	-0.034	-0.002	-0.01	$+0.027$
5	4	0.047	$9.492 \cdot 10^{-7}$	$+0.001$	-0.037	-0.002	-0.011	$+0.035$
6	5	0.058	$1.181 \cdot 10^{-8}$	$+0.001$	-0.039	-0.002	-0.012	$+0.041$
8	7	0.079	$5.745 \cdot 10^{-13}$	$+0.002$	-0.03	-0.002	-0.014	$+0.051$
11	10	0.103	$2.286 \cdot 10^{-20}$	$+0.002$	-0.033	-0.002	-0.016	$+0.062$
15	14	0.128	$1.944 \cdot 10^{-31}$	$+0.003$	-0.041	-0.002	-0.017	$+0.073$
20	19	0.152	$1.161 \cdot 10^{-46}$	$+0.003$	-0.037	-0.002	-0.017	$+0.081$

Table 12.3: DGIGA semi-discretizations that minimize (12.6). Each optimal scheme has been selected among all possible combinations of $\mathrm{k}, \mathrm{p}, \varkappa$ (number of breakpoint spans, degree, smoothness) that result in a given J (number of basis functions per patch). Relative changes are increments with respect to the DGSEM method of equal J in table 12.1; positive increments imply improvements over the baseline.

J	k	p	$\varkappa$	$\mathrm{cond}\left(\widetilde{\boldsymbol{\mathcal{M}}}\right)$	$-\frac{\Delta F(100)}{F(100)}$	$\frac{\Delta A_T}{A_T}$	$\frac{\Delta \kappa_f}{\kappa_f}$	$\frac{\Delta \kappa_{1\%}}{\kappa_{1\%}}$	$\frac{\Delta \varsigma_{\max}}{\varsigma_{\max}}$
3	1	2	1	10	$-7 \cdot 10^{-15}$	$+6 \cdot 10^{-6}$	$-1 \cdot 10^{-14}$	$-4 \cdot 10^{-15}$	$+2 \cdot 10^{-10}$
4	1	3	2	35	$-3 \cdot 10^{-14}$	$+0.017$	$+1 \cdot 10^{-14}$	$+2 \cdot 10^{-14}$	$-5 \cdot 10^{-11}$
5	1	4	3	126	$-5 \cdot 10^{-14}$	$+0$	$+7 \cdot 10^{-15}$	$+5 \cdot 10^{-15}$	$-1 \cdot 10^{-11}$
6	2	3	1	34.92	$+0.028$	-0.111	$+0.03$	$+0.011$	$+0.53$
8	2	5	3	326	$+0.033$	-0.061	$+0.025$	-0.01	$+0.298$
11	2	7	4	3,507.34	$+0.086$	-0.228	$+0.062$	$+0.023$	$+0.348$
15	2	10	6	$1.5 \cdot 10^5$	$+0.107$	-0.121	$+0.074$	$+0.034$	$+0.331$
20	2	14	9	$2.8 \cdot 10^7$	$+0.119$	-0.209	$+0.072$	$+0.031$	$+0.29$

Table 12.4: All DGIGA candidates involved in the solution of (12.8), sorted from best to worst.

J	k	p	$\varkappa$	$F(100)$
20	2	14	9	0.461
20	5	7	4	0.477
20	2	12	5	0.486
20	2	15	11	0.487
15	2	10	6	0.493
20	3	11	7	0.496
20	2	16	13	0.506
20	4	10	7	0.507
20	7	7	5	0.515
20	2	17	15	0.517
20	2	11	3	0.52
15	3	8	5	0.52
15	2	11	8	0.522
20	2	18	17	0.522
20	1	19	18	0.523
20	3	13	10	0.524
20	8	5	3	0.526
20	6	9	7	0.528
15	2	9	4	0.529
20	3	17	16	0.534
20	3	15	13	0.536
11	2	7	4	0.538
20	5	11	9	0.542
15	2	12	10	0.542
15	2	8	2	0.544
20	4	16	15	0.544
20	4	13	11	0.544
20	2	13	7	0.545
20	13	7	6	0.546
20	14	6	5	0.548
15	5	6	4	0.55
20	12	8	7	0.551
15	2	13	12	0.551
20	5	15	14	0.552
15	1	14	13	0.553
20	11	9	8	0.558
20	6	14	13	0.56
15	4	8	6	0.561
15	6	4	2	0.561
15	3	10	8	0.561
20	10	10	9	0.564
20	7	13	12	0.565
20	15	5	4	0.565
15	3	12	11	0.566
20	9	11	10	0.567
20	8	12	11	0.568
11	2	8	6	0.573
20	3	7	1	0.576
15	4	11	10	0.577
11	4	4	2	0.582
11	3	6	4	0.582
15	5	10	9	0.586
20	2	10	1	0.587
11	2	9	8	0.588
11	1	10	9	0.589
15	9	6	5	0.59
15	6	9	8	0.592
15	8	7	6	0.592
15	7	8	7	0.594
15	10	5	4	0.594
20	3	9	4	0.597
20	16	4	3	0.598
11	3	8	7	0.606
8	2	5	3	0.611
15	11	4	3	0.617
11	4	7	6	0.618
15	2	7	0	0.62
11	5	6	5	0.626
11	6	5	4	0.631
8	2	6	5	0.631
8	1	7	6	0.631
11	7	4	3	0.643
8	3	3	1	0.647
20	4	7	3	0.648
8	3	5	4	0.651
11	2	5	0	0.656
6	2	3	1	0.656
15	3	6	2	0.659
20	17	3	2	0.664
15	12	3	2	0.666
8	4	4	3	0.668
8	2	4	1	0.674
11	2	6	2	0.675
6	1	5	4	0.675
11	8	3	2	0.678
6	2	4	3	0.68
8	5	3	2	0.694
5	1	4	3	0.706
6	3	3	2	0.708
20	9	3	1	0.716
5	2	3	2	0.716
15	4	5	2	0.731
4	1	3	2	0.744
11	9	2	1	0.752
8	6	2	1	0.756
11	3	4	1	0.756
15	13	2	1	0.758
6	4	2	1	0.759
5	3	2	1	0.764
20	18	2	1	0.767
4	2	2	1	0.769
20	6	4	1	0.778
3	1	2	1	0.796
15	7	2	0	0.825
5	4	1	0	0.839
4	3	1	0	0.84
6	5	1	0	0.848
3	2	1	0	0.85
8	7	1	0	0.853
11	10	1	0	0.864
15	14	1	0	0.875
5	2	2	0	0.884
11	5	2	0	0.886
20	19	1	0	0.887

the exact solution; details are in §A.2.5). These correspond to the optima found in §12.1.2 to improve significantly over their baseline methods—i.e. (some) DGIGA discretizations.

Starting with dispersion, compared to the baseline, the optimal schemes incur in higher error at low wavenumbers; this difference reduces with increasing wavenumber, and it eventually reverses. The DGIGA schemes thus end up maintaining a lower phase shift than DGSEM for a majority of the spectrum. In addition, this crossover occurs (for all methods considered) within the well-resolved wavenumber range; this suggests that the increased error up to that point is negligible.

When it comes to dissipation, the optimal methods appear to allow higher dissipation both in the well-resolved range and in the later portion of the underresolved range. For a small portion at the middle of the spectrum, however, the situation reverses, and dissipation can be clearly observed to be lower for the optimal DGIGA scheme than for the DGSEM baseline.

All in all, it seems that the benefit of the optimal DGIGA schemes stems from a redistribution of both dispersion and dissipation errors away from the early portion of the underresolved range, which is pushed to the lower (where it makes little difference, since both it and the baseline are very accurate there) and higher (where a higher numerical dissipation might be beneficial, as will be argued in §12.3) wavenumber extremes. As a final note, recall that this qualitative analysis is not used to justify these semi-discretizations as being optimal; on the contrary, this is an explanation of why these can be better than the baseline, a fact that is a direct consequence of the choice of objective function. Hence the importance of choosing a sensible one—which I have argued as so due to it being associated with a theoretical error norm, as well as having been used (in a very similar form) in the literature.

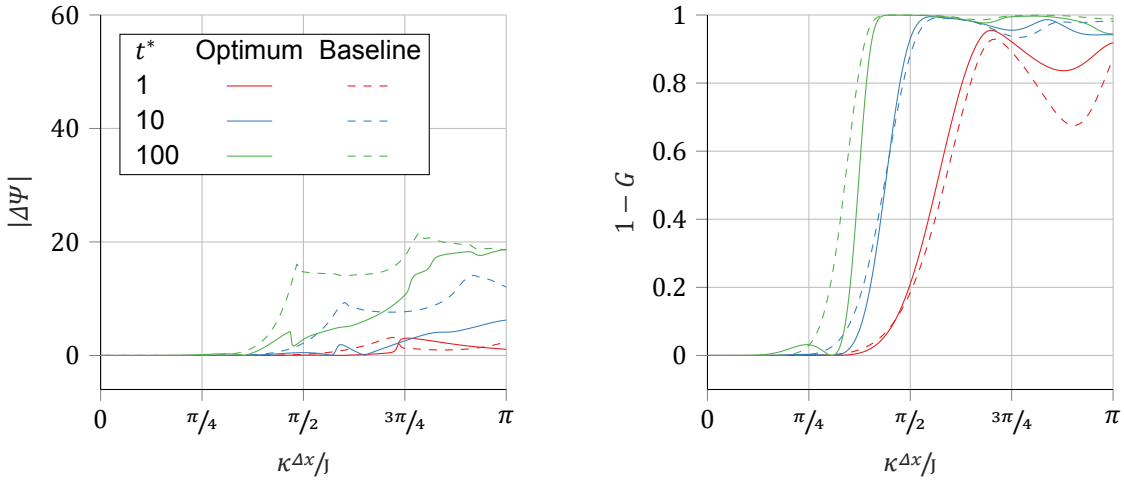


Figure 12.1: Combined-mode errors, as functions of the wavenumber, for $\mathrm{J} = 6$ and several (nondimensional) time instants. Optimum: DGIGA with $\mathrm{k} = 2$, $\mathrm{p} = 3$ and C^1 smoothness. Baseline: DGSEM with $\mathrm{p} = 5$. Lower is better.

12.3. Balance between dispersion and dissipation

Results discussed in §12.2 suggest an additional advantage of the optimal DGIGA: increased dissipation in the underresolved range. Numerical dissipation is typically undesirable, since it causes details of the solution to diffuse away. In some cases, however, one can take advantage of this effect and use it as a numerical filter in an *implicit* LES approach: the numerical dissipation itself ensures that there is no nonphysical back-scatter of the energy that would have been dissipated by molecular viscosity should the discretization have been fine enough to resolve the small flow structures where that process takes place; this is in contrast to *explicit* LES, in which dissipation of unresolved scales is accomplished using a dedicated sub-grid-scale model.

Figure 12.6 shows the amount of dissipation introduced at each wavenumber, in proportion to how badly the same wavenumber is affected by dispersion, for each optimum and its baseline. This ratio between dispersion and dissipation, $\overline{\chi}$, is defined by (A.73) and can be interpreted as a *mean lifetime* (in terms of degrees of freedom crossed) of any spurious wave packet, present in the numerical solution, centered at a given wavenumber (further details are given in §A.5.4). It is shown for the underresolved

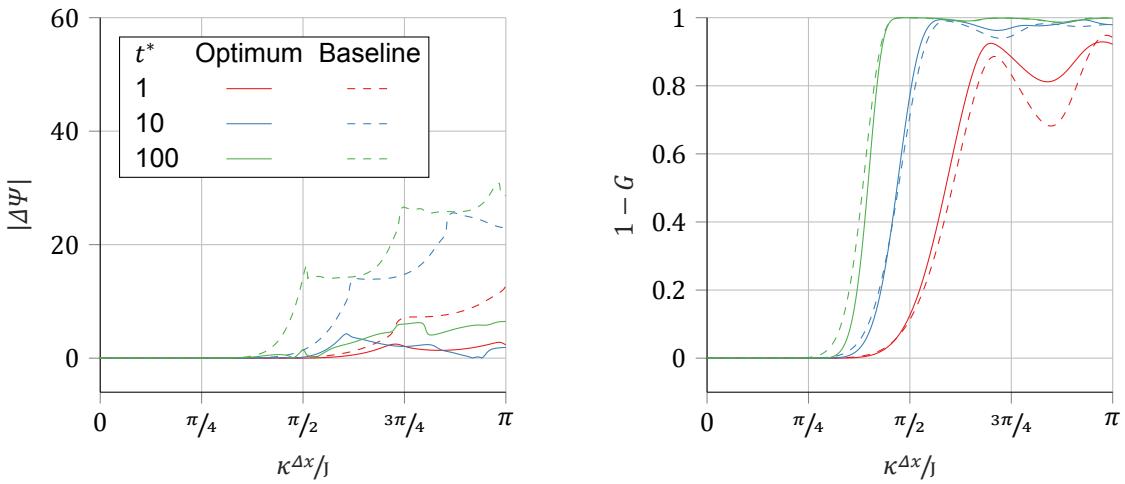


Figure 12.2: Same as figure 12.1, for $\mathrm{J} = 8$. Optimum: DGIGA with $\mathrm{k} = 2$, $\mathrm{p} = 5$, C^3. Baseline: $\mathrm{p} = 7$ DGSEM.

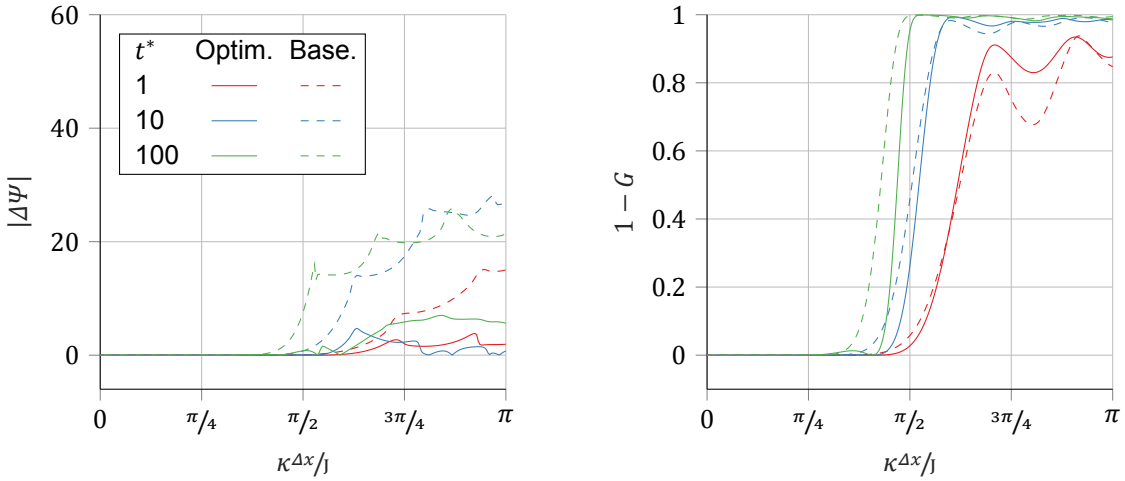


Figure 12.3: Idem for $\mathrm{J} = 11$. DGIGA: $\mathrm{k} = 2$, $\mathrm{p} = 7$, C^4. DGSEM: $\mathrm{p} = 10$.

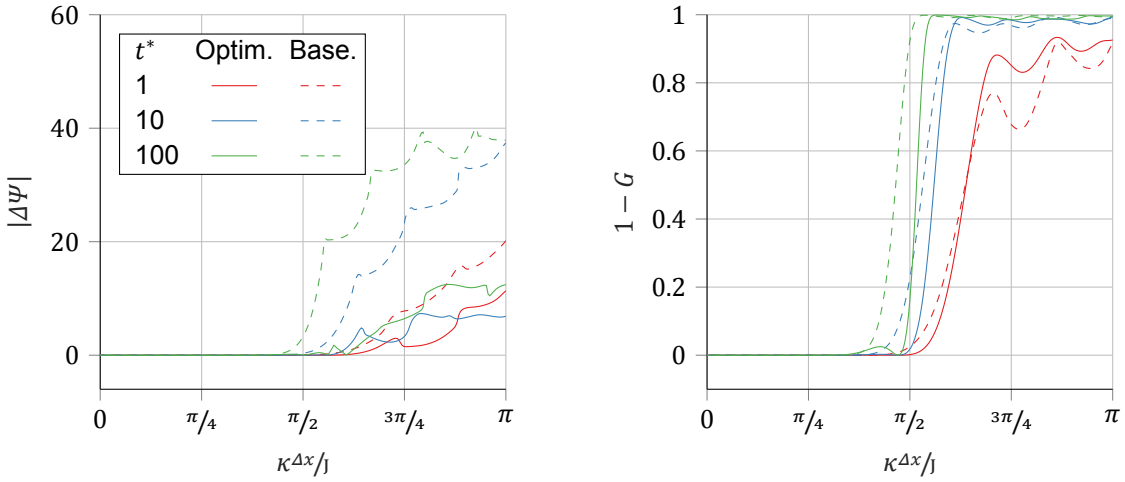


Figure 12.4: Idem for $\mathrm{J} = 15$. DGIGA: $\mathrm{k} = 2$, $\mathrm{p} = 10$, C^6. DGSEM: $\mathrm{p} = 14$.

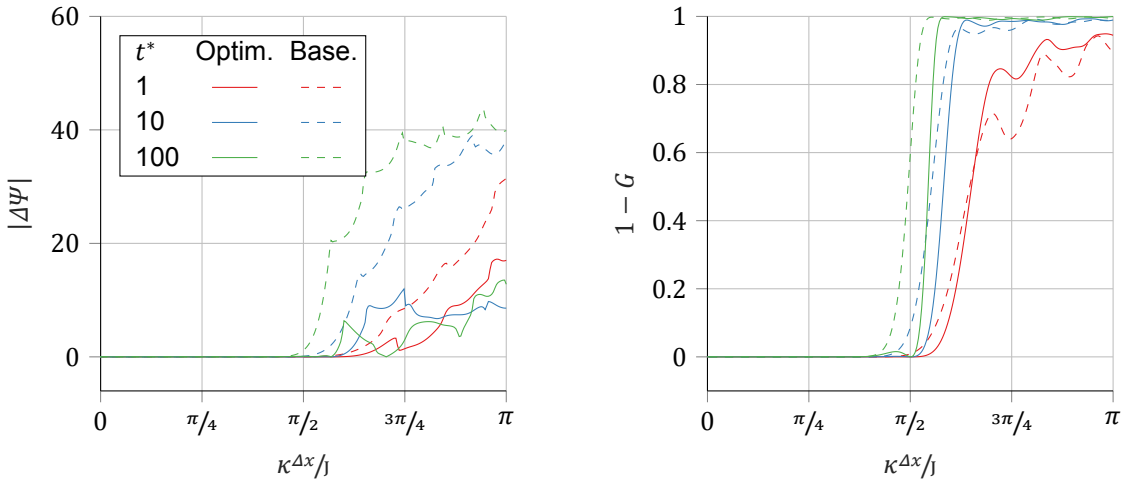


Figure 12.5: Idem for $\mathrm{J} = 20$. DGIGA: $\mathrm{k} = 2$, $\mathrm{p} = 14$, $\mathcal{C}^9$. DGSEM: $\mathrm{p} = 19$.

range (as defined in §A.5.2) of each scheme only; hence, figure 12.6 also compares their resolving efficiencies (recall that the Nyquist wavenumber of a DG-like scheme is $\kappa = {}^{\mathrm{J}\pi}/_{\Delta x}$). In addition, its norm (A.74) for these same cases is listed in table 12.5.

Fixed the number of degrees of freedom, the optimized DGIGA semi-discretization has higher dissipation than its DGSEM counterpart for a given amount of dispersion (at small wavelengths). Moreover, it can also be seen that the divide between well-resolved and underresolved ranges is pushed to higher wavenumbers in the former. It can be argued quite conclusively, therefore, that the optima found are not only still suitable but also even superior than DGSEM for use in implicit LES. Note that the FR/CPR optima do show a reduction (up to 24 %) over the baseline in this metric—see table 12.5; the DGIGA optima, however, do so much more significantly (47 % to 72 % improvement).

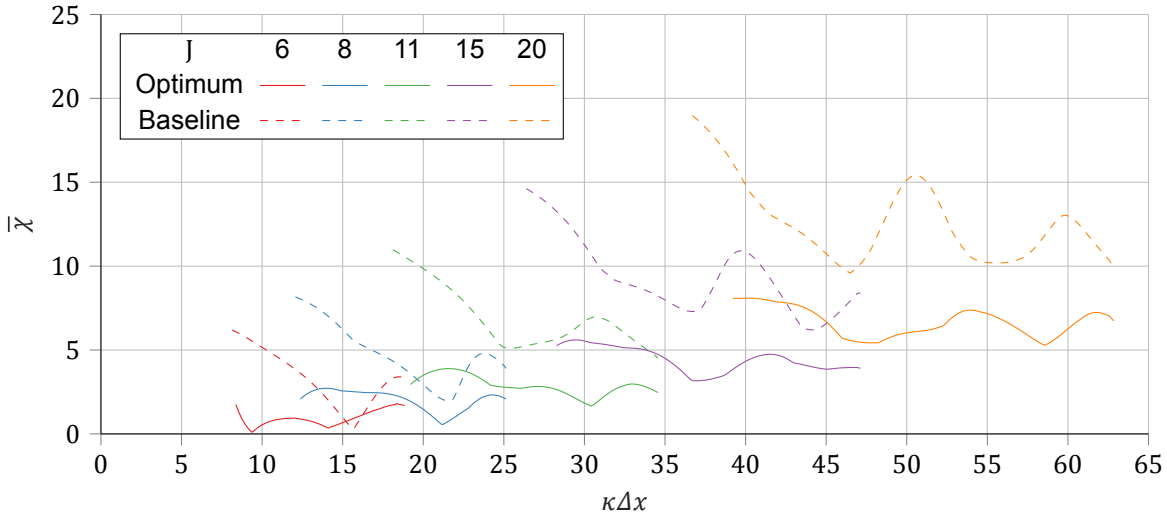


Figure 12.6: Mean lifetime of a spurious wave packet as a function of its wavenumber (A.73), for each optimal DGIGA method with $\mathrm{k} \neq 1$ and its DGSEM baseline. Lower is better; plotted over the underresolved range only (the furthest to the right the graph starts, the more resolution).

12.4. Relative cost at a fixed resolution

As a final criterion through which to judge the results of the optimization, I propose to compare the computational cost in terms of *number of floating point operations* (i.e. time complexity), for a fixed *resolution*, in the following way. Consider the linear advection (2.19) of some range of scales—let the

Table 12.5: Mean lifetime of spurious a wave packet, in degrees of freedom crossed, averaged over the underresolved wavenumber range of each semi-discretization. The lower this lifetime, the sooner spurious energy is removed from the numerical solution; hence, lower is better (i.e. negative increments represent an improvement, in this case). Relative changes are with respect to either the $\mathrm{p} = 0$ case (DGSEM) or the DGSEM scheme of equal J (FR/CPR and DGIGA).

(a) DGSEM

J	$\lVert\overline{\chi}\rVert$	$\frac{\Delta\lVert\overline{\chi}\rVert}{\lVert\overline{\chi}\rVert}$
1	1	0
2	1	0
3	1.47	0.47
4	2.01	1.01
5	2.62	1.62
6	3.3	2.29
8	4.77	3.76
11	6.93	5.92
15	9.36	8.35
20	12.66	11.65

(b) FR/CPR

J	$\lVert\overline{\chi}\rVert$	$\frac{\Delta\lVert\overline{\chi}\rVert}{\lVert\overline{\chi}\rVert}$
3	1.36	−0.07
4	1.82	−0.09
5	2.32	−0.11
6	2.87	−0.13
8	4.01	−0.16
11	5.64	−0.19
15	7.35	−0.21
20	9.62	−0.24

(c) DGIGA

J	$\lVert\overline{\chi}\rVert$	$\frac{\Delta\lVert\overline{\chi}\rVert}{\lVert\overline{\chi}\rVert}$
3	1.45	−0.01
4	1.95	−0.03
5	2.47	−0.06
6	0.93	−0.72
8	2.04	−0.57
11	2.9	−0.58
15	4.44	−0.53
20	6.67	−0.47

smallest of these be denoted λ—over a periodic domain of length L, at a rate a. Suppose that we are interested in the solution after some arbitrary amount of simulated time, $\Delta T > 0$, has passed. In order to obtain this solution numerically, we will employ $\mathrm{K} \in \mathbb{Z}^+$ spatial elements or patches (either DGSEM, FR or DGIGA), each of size Δx and with $\mathrm{J} \in \mathbb{Z}^+$ degrees of freedom, in combination with the SSP-RK3(3) time scheme.

12.4.1. Number of time-steps

The number of time-steps of size Δt necessary to reach ΔT is:

$$\mathrm{N}_{\mathrm{steps}} = \left\lceil \frac{\Delta T}{\Delta t} \right\rceil, \tag{12.9}$$

which accounts for the size of the last time-step possibly being truncated so that $\mathrm{N}_{\mathrm{steps}} \in \mathbb{Z}^+$. Assume that we use the largest possible time-step size that is linearly stable in combination with each given spatial scheme. Since $L = \mathrm{K}\Delta x$, it follows that:

$$\Delta t = \frac{\varsigma_{\max}}{a/\Delta x} \implies \mathrm{N}_{\mathrm{steps}} = \left\lceil \frac{\mathrm{K}}{\varsigma_{\max}} \frac{\Delta T}{L/a} \right\rceil. \tag{12.10}$$

12.4.2. Number of elements or patches

Next, assume[6] that the minimum amount of resolution required to accurately resolve the smallest scale present in the solution is given by the relationship:

$$\frac{2\pi}{\lambda} = \kappa_f\,, \tag{12.11}$$

i.e. the smallest scale present in the solution is precisely at the boundary between well-resolved and underresolved portions of any given spatial scheme's wavenumber spectrum. Consequently, the following relationship holds:

$$\frac{L}{\lambda} = \frac{\kappa_f \Delta x}{\mathrm{J}\pi} \frac{\mathrm{JK}}{2}\,, \tag{12.12}$$

[6] This, of course, is merely a convention: recall that the threshold selected for the definition of κ_f (§A.5.2) is arbitrary, and that the time scheme introduces additional errors not taken into account.

and, therefore, the minimum number of elements or patches (which has to be a positive integer) necessary to ensure sufficient resolution is:

$$\mathrm{K} = \left\lceil \frac{2}{\mathrm{J}} \frac{\mathrm{J}\pi}{\kappa_f \Delta x} \frac{L}{\lambda} \right\rceil , \tag{12.13}$$

which only depends on L/λ, for a given spatial scheme.

A first guess on the typical range for this ratio may be made based on turbulence theory [100], via the relationship $L/\lambda \approx \mathrm{Re}_L^{3/4}$—$\mathrm{Re}_L$ is the Reynolds number referred to the integral length scale, which for estimation purposes can be assumed approximately equal to L. Once a value for this ratio has been selected, both K and $\mathrm{N}_{\mathrm{steps}}$ become fixed—resolution efficiency and maximum Courant numbers of each scheme are listed in tables 12.1, 12.2 and 12.3.

12.4.3. Cost model

The total number of floating-point operations made throughout a simulation is given by the product:

$$\mathrm{N}_{\mathrm{flops}} = \mathrm{N}_{\mathrm{flops/step}} \mathrm{N}_{\mathrm{steps}} , \tag{12.14}$$

where the number of operations per step, $\mathrm{N}_{\mathrm{flops/step}}$, depends on the spatial and temporal discretizations and on how they are implemented. For my particular implementation of these methods, I estimate the following costs per step[7]:

$$\mathrm{N}^{\mathrm{DGSEM}}_{\mathrm{flops/step}} = 6\mathrm{J}^2\mathrm{K} + 45\mathrm{JK} + 24\mathrm{J} + 18\mathrm{K} + 15 , \tag{12.15a}$$

$$\mathrm{N}^{\mathrm{FR/CPR}}_{\mathrm{flops/step}} = 6\mathrm{J}^2\mathrm{K} + 51\mathrm{JK} + 24\mathrm{J} + 24\mathrm{K} + 15 , \tag{12.15b}$$

$$\begin{aligned} \mathrm{N}^{\mathrm{DGIGA}}_{\mathrm{flops/step}} = {} & 12\mathrm{Kkp}^2 - 12\mathrm{Kk}\varkappa^2 + 6\mathrm{J}^2\mathrm{K} - 6\mathrm{Kp}^2 + 12\mathrm{K}\varkappa^2 + 12\mathrm{JKp} + 12\mathrm{Kkp} - 12\mathrm{Kk}\varkappa \\ & + 36\mathrm{JK} - 6\mathrm{Kp} + 12\mathrm{K}\varkappa + 24\mathrm{J} + 36\mathrm{K} + 15 . \end{aligned} \tag{12.15c}$$

Appendix B details how these have been obtained.

Even though I have chosen to use FLOPs as a measure of time complexity by which to compare the various schemes, the reader should keep in mind that this magnitude is regarded as a *bad* estimate of performance in modern computers: it is entirely possible for an algorithm that requires much fewer FLOPs to be slower than an alternative, if the latter has better memory management (recall figure 1.2).

12.4.4. Results and discussion

Figure 12.7 shows the cost predicted under the assumptions of §§12.4.1–12.4.3 for DGSEM. Among those tested, degrees $\mathrm{p} = 4$ and $\mathrm{p} = 5$ achieve (an almost identical) lowest overall cost per unit of simulated time, throughout the range of L/λ ratios sampled. Both higher and lower degrees are sub-optimal according to this metric: $\mathrm{p} = 3$ and $\mathrm{p} = 7$ are about 10 % worse; $\mathrm{p} = 2$ and $\mathrm{p} = 10$ are both 38 % more costly; $\mathrm{p} = 1$ and $\mathrm{p} = 19$ require, respectively, 2.30 and 1.50 times more FLOPs than $\mathrm{p} = 4$. The DGSEM equivalent of first order upwind FVM, despite its cost being minimal, clearly performs much worse (>1000 times more FLOPs) than the rest; this can be attributed to its poor resolution efficiency.

The optimal FR/CPR semi-discretizations shown in figure 12.8 are all about as costly as their baselines. Despite being slightly (up to 10 %) more expensive than the baseline for $\mathrm{J} < 11$, the two highest order schemes tested are actually slightly more cost-effective than their baselines under equal resolution requirements—which suggests that the increase in allowable Courant number gained by using $\eta > 0$ can outweigh the price, in terms of added computational complexity, invested in supporting the correction procedure. That being said, this improvement is very small (<5 %).

Lastly, figure 12.9 shows that the DGIGA optima fail to outperform DGSEM in regards to cost-efficiency, even when possessing *both* larger maximum Courant number and resolving efficiency. This can only be attributed to a significantly increased cost per time-step of DGIGA with respect to DGSEM. The DGIGA optima closest to its baseline (according to this metric) is $\mathrm{J} = 8$, with both higher and lower J schemes being worse in relation to their respective baselines. Note that this results already take into account the fact that DGIGA's operators can be sparser than their DGSEM counterparts for an equal number of degrees of freedom per patch (see B.2).

[7] Note that $\mathrm{N}^{\mathrm{DGSEM}}_{\mathrm{flops/step}} < \mathrm{N}^{\mathrm{FR/CPR}}_{\mathrm{flops/step}}$, the latter requiring $2(\mathrm{J}+1)$ additional FLOPs per stage and element (in my implementation).

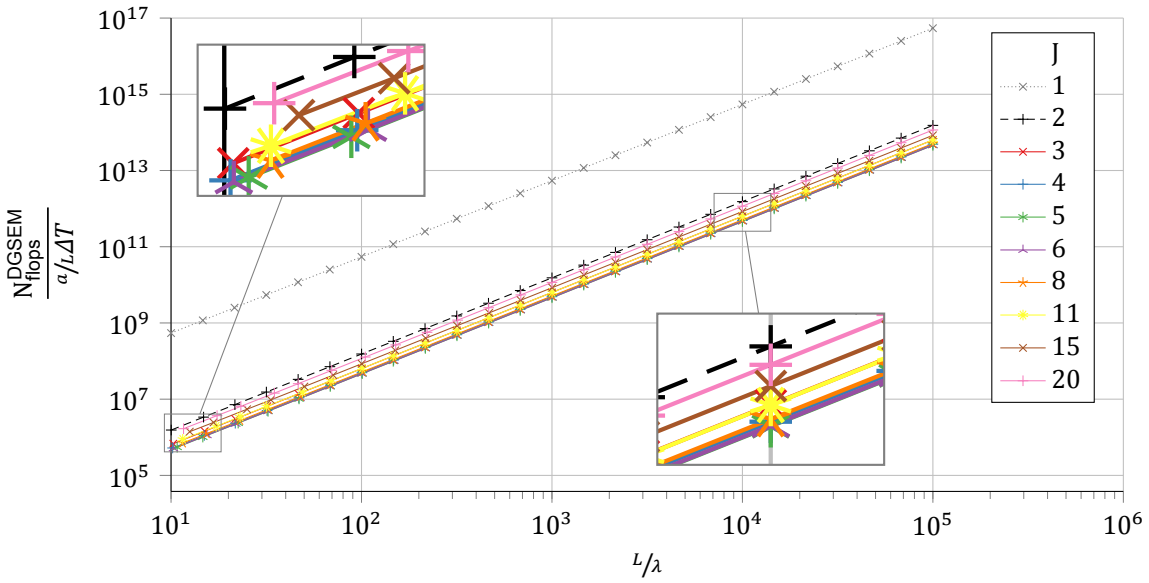


Figure 12.7: Estimated cost of solving the advection equation with SSP-RK3(3) and DGSEM, in terms of number of FLOPs per dimensionless unit of simulated time. Even though this cost increases strongly with the resolution requirement (note the logarithmic axis scales), the relative differences between orders remain approximately constant.

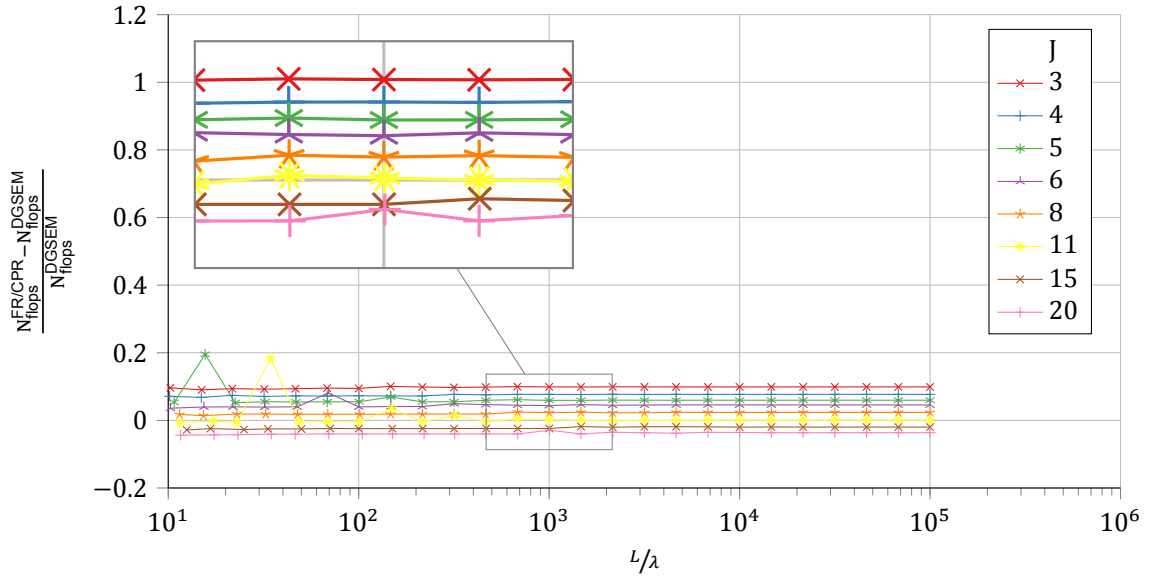


Figure 12.8: Cost of the FR/CPR optima of indicated order, relative to their DGSEM counterparts (figure 12.7).

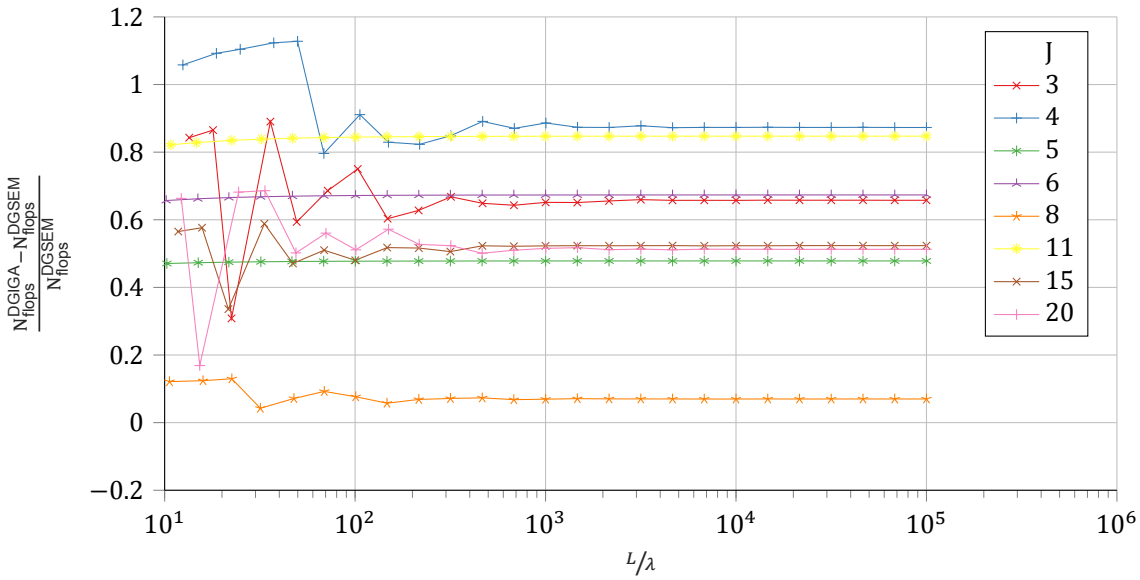


Figure 12.9: Relative cost of the DGIGA optima with respect to their baselines (DGSEM of equal number of degrees of freedom per element/patch).

13

Nonlinear Physics

All results shown and discussed in chapter 12 were based on *a priori* analytical considerations valid only for the linear advection equation. The extent to which any conclusions reached in such a setting can be generalized to more realistic flows is unclear. In this chapter, I verify numerically whether the optimal configurations found previously are actually so in practice, starting with the linear advection case, and compare their behavior when switching to the nonlinear advection that the inviscid Burgers equation represents (§13.1). Then, as a preliminary to chapter 14, I consider the Euler equations without using limiters, studying them via *a posteriori* modified wavenumber analysis (§13.2).

13.1. Burgers equation

The combined-mode analysis employed in §12.2 does not generalize easily to the nonlinear case. A simple alternative is to compare the growth of the numerical error as the simulation progresses, for each triplet of schemes of given J—these being: DGSEM baseline, FR/CPR optimum and DGIGA optimum (which, in addition, I consider in both its modal and nodal variants), as established in chapter 12. This strategy, purely experimental and *a posteriori* (as opposed to the analytically obtained *a priori* combined-mode analysis) has the downside of only detecting the combined effect of both dispersion and dissipation, and for all wavenumbers present in the numerical solution at once. I consider test problems (9.2), (9.8) and (9.10), and compare the L^2 norm of their error, as a function of time, with that of their linear advection counterparts: (9.1), (9.7) and (9.9).

13.1.1. Verification of combined-mode analysis results

In this subsection, I focus on comparing baseline and optimized spatial schemes in the latter's "design conditions"—i.e. for a marginally resolved monochromatic wave, advected in a linear fashion. The following test matrix details the setup of the experiments from which the results I discuss below have been obtained.

Table 13.1: Confirming the advantage of optimized schemes (most favorable conditions).

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N_{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor
1	13.4a	(9.1), $n = 7$	3(3)	10^{-3}	20	DGSEM	60	20	2	-	-	∞	-	-
2						FR/CPR				0.023				
3						DGIGA				-	1	1		
4						(nodal)								
5	13.6a	$n = 12$				DGSEM		10	5		-	∞		
6						FR/CPR				0.058				
7						DGIGA			3	-	2	1		
8						(nodal)								
9	13.8a	$n = 16$				DGSEM		3	19		-	∞		
10						FR/CPR				0.152				
11						DGIGA			14	-	2	9		

(continues in the next page)

Table 13.1: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
12						(nodal)								

All runs of table 13.1 indicate that the theoretical combined-mode analysis results also occur in practice. The sinusoidal initial condition (see figure 13.1a) in these three sets of runs has been selected such that $n \in \mathbb{Z}$ (i.e. an integer number of wavelengths fit in the domain) at the same time that it results in a wavenumber $\approx$90 % of the well-resolved threshold for DGSEM of degree 2, 5 and 19 (respectively); in this way, I target the range where the combined-mode response of each DGIGA optimum differs the most from its respective baseline (see §12.2). For $J = 3$ (figure 13.4a), the optimal DGIGA basis (runs 3 and 4) spans the same function space as the DGSEM one (run 1)—see §6.2.2 for details—hence we observe no difference between the two (in this linear context). The same applies to $J = 4$ and 5 (not shown). Whenever the optimized spatial scheme differs from the baseline, however, the former's error grows at a reduced rate as the simulation progresses, in relation to the latter; this suggests a reduced numerical dispersion and/or dissipation, as predicted (see figures 12.1 and 12.5). The advantage of the optimized basis over the baseline in this regard is more pronounced the higher the J gets, as expected (this also holds for the rest of J, which are not shown for conciseness). Likewise, the FR/CPR optimum is less so than the DGIGA one in every case. All in all, these results confirm that the optimization procedure, including the choice of objective function, is sensible (in the linear case)—at least, for a solution containing a wide range of scales over a long time. Note that in all linear cases, nodal and modal versions of DGIGA give identical results.

13.1.2. Influence of the initial condition

Considering still the linear advection equation, let us now investigate whether or not the previous observations can be extended to less favorable initial conditions.

Table 13.2: Testing the advantage of optimized schemes (unfavorable conditions).

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	13.4b	(9.7)	3(3)	10^{-3}	20	DGSEM	60	20	2	-	-	∞	-	-
2						FR/CPR				0.023				
3						DGIGA				-	1	1		
4						(nodal)								
5	13.6b					DGSEM		10	5		-	∞		
6						FR/CPR				0.058				
7						DGIGA			3	-	2	1		
8						(nodal)								
9	13.8b					DGSEM		3	19		-	∞		
10						FR/CPR				0.152				
11						DGIGA			14	-	2	9		
12						(nodal)								
13	13.4c	(9.9)				DGSEM		20	2		-	∞		
14						FR/CPR				0.023				
15						DGIGA				-	1	1		
16						(nodal)								
17	13.6c					DGSEM		10	5		-	∞		
18						FR/CPR				0.058				
19						DGIGA			3	-	2	1		
20						(nodal)								
21	13.8c					DGSEM		3	19		-	∞		
22						FR/CPR				0.152				
23						DGIGA			14	-	2	9		
24						(nodal)								

For a Gaussian initial condition, figure 13.6b suggests that the optimized bases (both FR and DGIGA) may lose their advantage over DGSEM at large scales—note that a Gaussian signal contains

most of its energy in the smallest wavenumbers (see figure 13.1b). Nevertheless, this is no longer the case in figure 13.8b (I have observed the same to occur for $\mathrm{J} = 8, 11, 15, 20$ as well): the DGIGA optimum returns to being the most accurate. The FR/CPR optima differ only very slightly from their baselines for all J. Be it as it may, these results show that for a smooth and well-resolved solution of the advection equation, any of the three types of bases produce very small errors, even for long simulation times (10 domain lengths in this case).

With problem (9.9), a similar situation as for the Gaussian initial condition occurs (note the similarity between figures 13.1b and 13.1c). In this case, however, the error norm associated with DGIGA is always equal or lower than that of DGSEM (figures 13.4c, 13.6c and 13.8c; also for the rest of values, not shown). Note that the initial condition is now only C^0 smooth; the dominant contributions to the error in this test case are the greatly diffused regions of the approximate solution around the sharp extrema of its exact counterpart.

13.1.3. Linear vs. nonlinear advection

To conclude this section, I repeat all the previous simulations for the Burgers equation and consider the question: do the optimized methods retain their advantage in this (scalar) nonlinear context?

Table 13.3: Exploring the behavior of optimized schemes applied to the Burgers equation.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N_{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor
1	13.5a	(9.2), $n = 7$	3(3)	10^{-3}	0.409	DGSEM	60	20	2	-	-	∞	-	-
2						FR/CPR				0.023				
3						DGIGA				-	1			
4						(nodal)								
5	13.7a	$n = 12$			0.239	DGSEM		10	5		-			
6						FR/CPR				0.058				
7						DGIGA			3	-	2	1		
8						(nodal)								
9	13.9a	$n = 16$			0.179	DGSEM		3	19		-	∞		
10						FR/CPR				0.152				
11					0.06 (unstable)	DGIGA			14	-	2	9		
12					0.034 (unstable)	(nodal)								
13	13.5b	(9.8)			0.4	DGSEM		20	2		-	∞		
14						FR/CPR				0.023				
15						DGIGA				-	1	1		
16						(nodal)								
17	13.7b					DGSEM		10	5		-	∞		
18						FR/CPR				0.058				
19						DGIGA			3	-	2	1		
20						(nodal)								
21	13.9b					DGSEM		3	19		-	∞		
22						FR/CPR				0.152				
23					0.208 (unstable)	DGIGA			14	-	2	9		
24					0.284 (unstable)	(nodal)								
25	13.5c	(9.10)			7	DGSEM		20	2		-	∞		
26						FR/CPR				0.023				
27						DGIGA				-	1	1		
28						(nodal)								
29	13.7c					DGSEM		10	5		-	∞		
30						FR/CPR				0.058				
31						DGIGA			3	-	2	1		
32					1.229 (unstable)	(nodal)								
33	13.9c					DGSEM		3	19		-	∞		
34						FR/CPR				0.152				
35					0.171 (unstable)	DGIGA			14	-	2	9		
36					0.111 (unstable)	(nodal)								

Let us first return to the sinusoidal initial condition. The significant advantage that DGIGA had over DGSEM in the linear situation is lost (compare figures 13.6a and 13.7a). Moreover, nodal and modal

versions of it are no longer identical, and the former experiences significantly higher errors than its modal counterpart for all J (which is consistent with the increased numerical dissipation that I attribute to it, see §6.3.1). Similarly, DGIGA and DGSEM no longer produce identical results for $J = 3, 4, 5$ (only the first is shown), with DGIGA being always worse (figure 13.5a, lower J not shown).

For problems (9.8) and (9.10), modal DGIGA (when stable) is on par with the baseline, perhaps even marginally better for some J (runs 17 to 20 and 29 to 31; see also figure 13.3). I attribute this to the fact that the DGIGA optima have lower order than their DGSEM baselines; this, in turn, results in less pronounced spurious oscillations near steep gradients. This same effect is even more pronounced in nodal DGIGA, yet it appears that the increased numerical dissipation ends up being excessive: its L^2 errors are much larger than for its modal counterpart in nonlinear cases. The advantage over DGSEM that the FR/CPR optima possess appears to be minimal.

Runs 11, 12, 23, 24, 32, 35 and 36 experience numerical instability. I do not fully understand what is causing this phenomenon; all I can offer are the following observations:

- All schemes that experience instability in the Burgers equation are linearly stable (confirmed both theoretically and numerically).
- The only difference between stable and unstable spatial schemes, not only conceptually but also at the implementation level (see appendix B), is the computation of the spatial residuals; this instability, therefore, must have to do with the particularities of the B-spline basis itself.
- The higher the number of degrees of freedom per patch, the more prone to instability a basis is. This coincides with more ill-posed mass, gradient and residual matrices, but that alone does not explain the crashes (while the condition number of the mass matrix of optimal $J = 20$ DGIGA is $\approx 4 \times 10^7$, it is only ≈ 53 for $J = 6$).
- Increasing the number of patches, even if keeping each one's basis fixed, can counteract this instability.
- Nonlinear stabilization via limiters seems to correct the problem (see table 13.4 and associated figures).

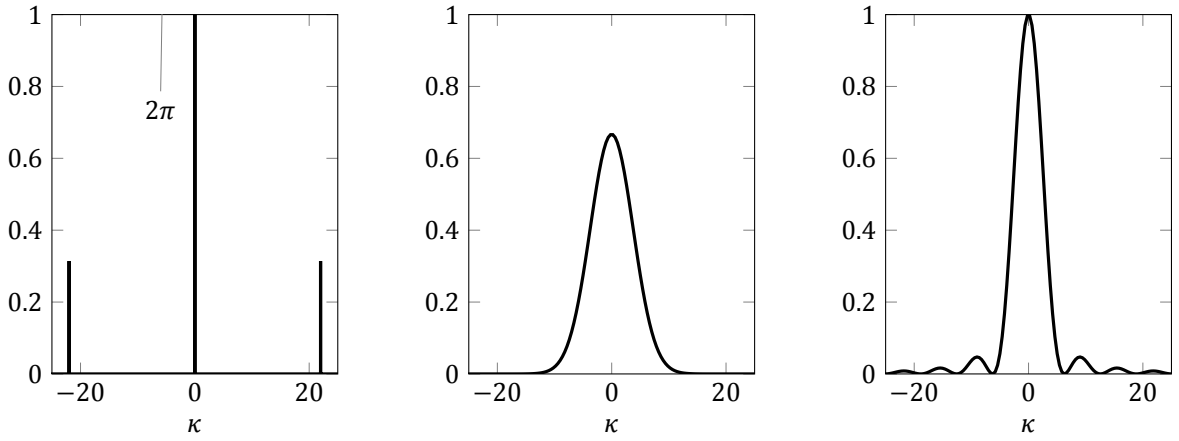
Table 13.4: Investigating the unexplained nonlinear instability of DGIGA.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	13.2a	(9.8)	3(3)	10^{-3}	0, 0.1, 0.2, 0.3, 0.43	DGSEM	60	20	2	-	-	∞	-	-
2						DGIGA			2		1			
3						(nodal)								
4						DGIGA-AFC							AFC	
5						(nodal)								
6	13.2b				0, 0.1, 0.16, 0.22, 0.33, 0.43	DGSEM		3	19		-	∞	-	
7					0, 0.1, 0.16 (unstable)	DGIGA			14		2	9		
8					0, 0.1, 0.16, 0.22 (unstable)	(nodal)								
9					0, 0.1, 0.16, 0.22, 0.33, 0.43	DGIGA-AFC							AFC	
10						(nodal)								
11	13.3	(9.10)			1, 2, 3	DGSEM		10	5		-	∞	-	
12						DGIGA			3		2	1		
13					1 (unstable)	(nodal)								

The main conclusion to be extracted from the comparison between linear and nonlinear scalar conservation laws is that the schemes optimized for high resolution in the linear advection and assuming a smooth solution, lose most of their advantage over plain DGSEM if applied to the Burgers equation. And, not only that, they can even become unstable.

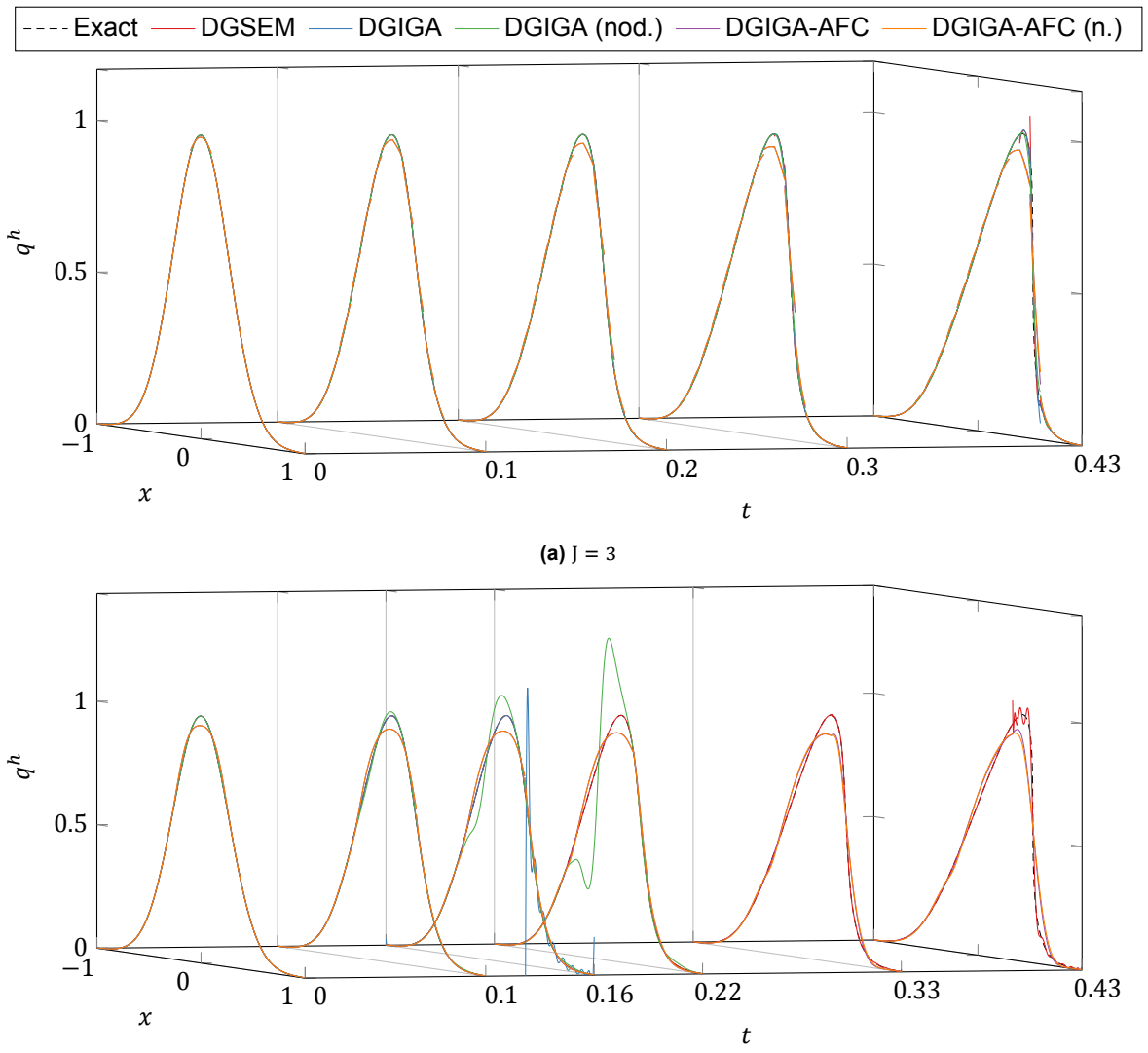
13.2. Euler equations

For the Euler equations, spurious oscillations of the solution—unlike in the Burgers equation—can easily cause the numerical solution to attain invalid states (e.g. negative density and/or imaginary speed



(a) Monochromatic, $n = 7$ **(b)** Gaussian **(c)** Triangular

Figure 13.1: Fourier transform of the initial conditions of problems (9.7), (9.8), (9.9) and (9.10). Defined as: $\mathcal{F}(q(x)) \coloneqq \int_{-\infty}^{+\infty} q(x) \mathrm{e}^{-\mathrm{i}\kappa x} \, \mathrm{d}x$. Vertical axes show its magnitude (that of figure 13.1a is clipped).



(a) J = 3

(b) J = 20

Figure 13.2: Solution at multiple time instants of batches 1 to 5 (top) and 6 to 10 (bottom), of table 13.4.

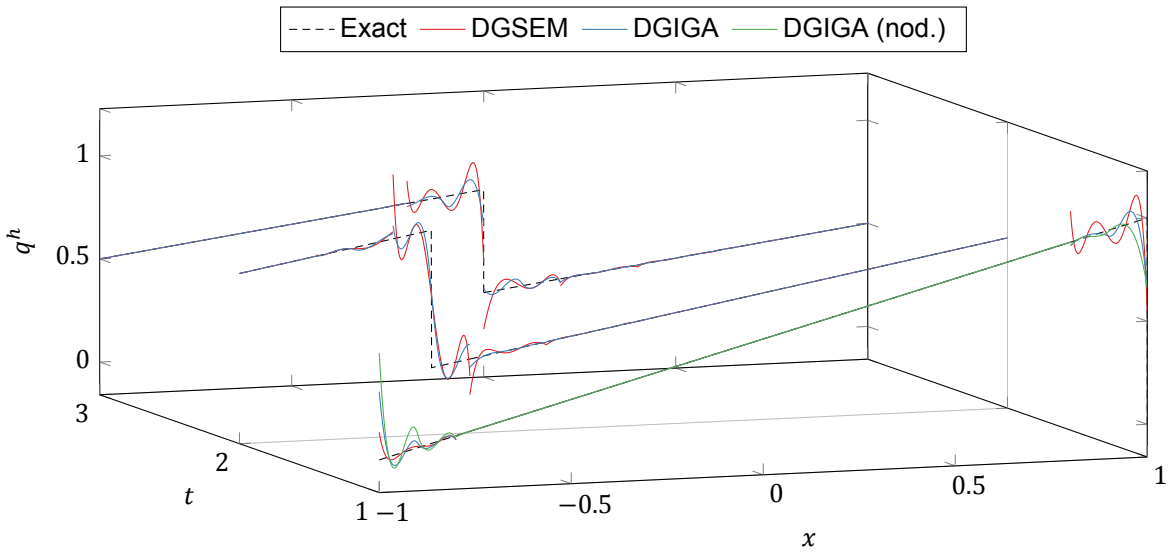


Figure 13.3: Solution of (9.10) at multiple instants; batches 11 to 13 of table 13.4. Notice that the shock position coincides with, either, the midpoint of one element/patch ($x = -0.5$, $t = 2$) or the interface between two ($x = \pm 1$, $t = 1$; $x = 0$, $t = 3$).

of sound), typically causing the solver algorithm to fail. In order to compare the accuracy of the different spatial schemes in the same manner as in §13.1, an exact solution that remains smooth (and, hence, free of said oscillations) for some period of time is required[1].

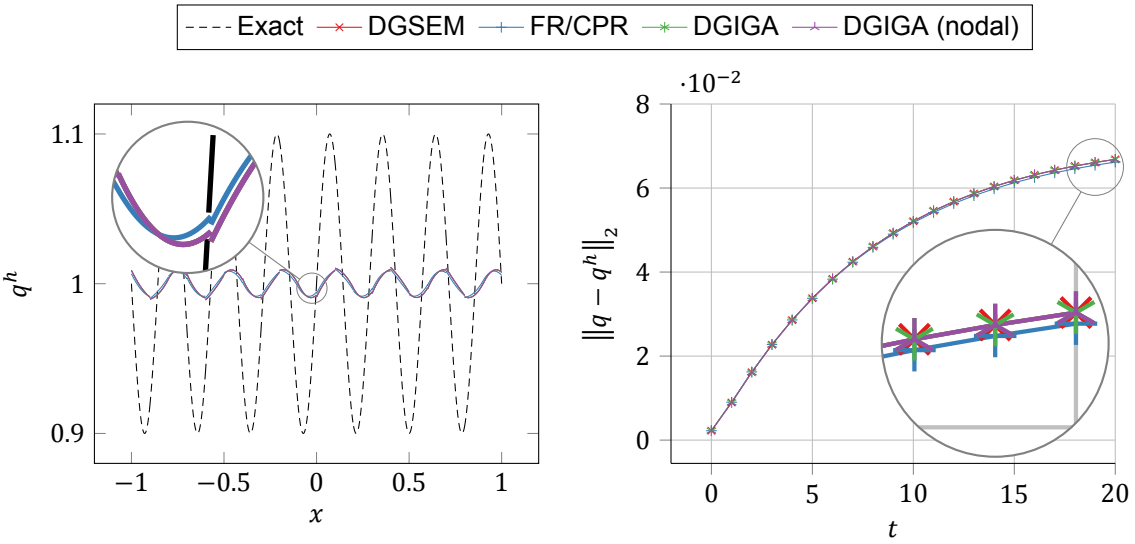
Unfortunately, exact solutions of this type are very scarce, even in one dimension. Yet, there is one case (albeit rather trivial) which does fulfill these requirements: problem (9.3). The solution of this problem consists on the propagation of the initial density profile, unmodified in amplitude nor frequency, at a phase speed of $u^0 = 1$; it is identical to the solution of the linear advection equation. Because there is essentially no nonlinear phenomena in this solution, I do not repeat here the comparison between spatial bases of §13.1—expecting similar results as for the linear case. Instead, I shall use the methodology proposed in [48, appendix C], which provides separate measures of dispersion and dissipation[2].

13.2.1. Modified wavenumber analysis a la Hickel et al.

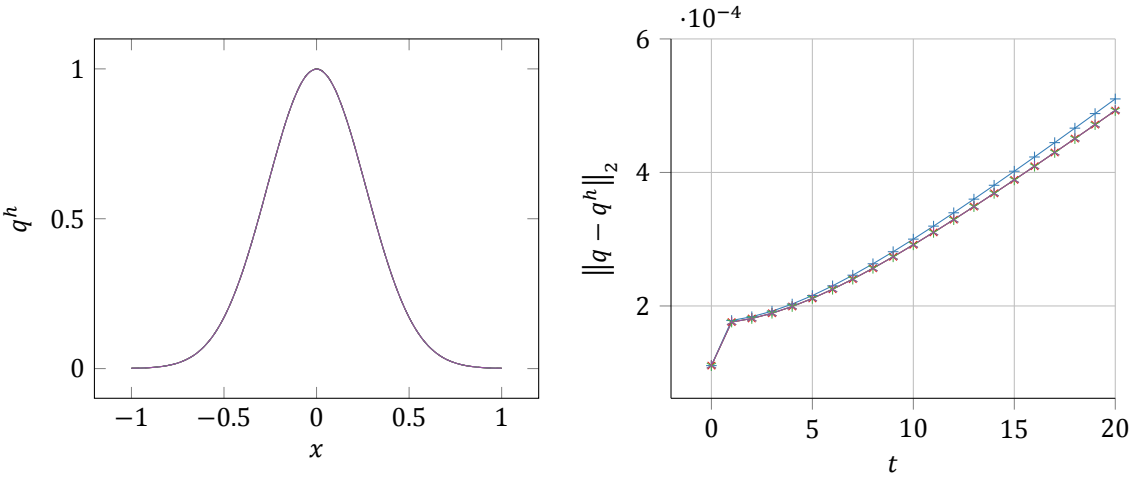
Hickel et al. [48] refer to this approach as an *a posteriori* modified wavenumber analysis. This stems from the fact that, in it, one evaluates a modified wavenumber associated with each given baseline wavenumber (see §A.2.2) using the spatial residual functions, once these have been computed by the numerical scheme just as if in the middle of any time-stage update. In all a posteriori results, I use 100 elements or patches and 100 samples in each (i.e. 10000 mesh-wide), to compute every modified wavenumber from the first component (i.e. density) of residual and state vectors (transformed to Fourier domain using Matlab's FFT routine[3]).

A feature of this approach is that it does not require the semi-discretization it is applied to to be linear (be it because the PDE is nonlinear or because e.g. limiters have been used). An immediately apparent shortcoming that I have observed in its high-order generalization, is that the modified wavenumber is misspredicted—there is a sudden spike in the dispersion relation—whenever two eigenmodes coincide on the same wavenumber. This, of course, cannot happen if there is no multiplicity of eigenmodes, i.e. if $\mathrm{J} = 1$. As a matter of fact, I argue in the following subsection (§13.2.2) that the presence of multiple modified wavenumbers for each baseline wavenumber means that this approach is not suitable for studying high-order schemes.

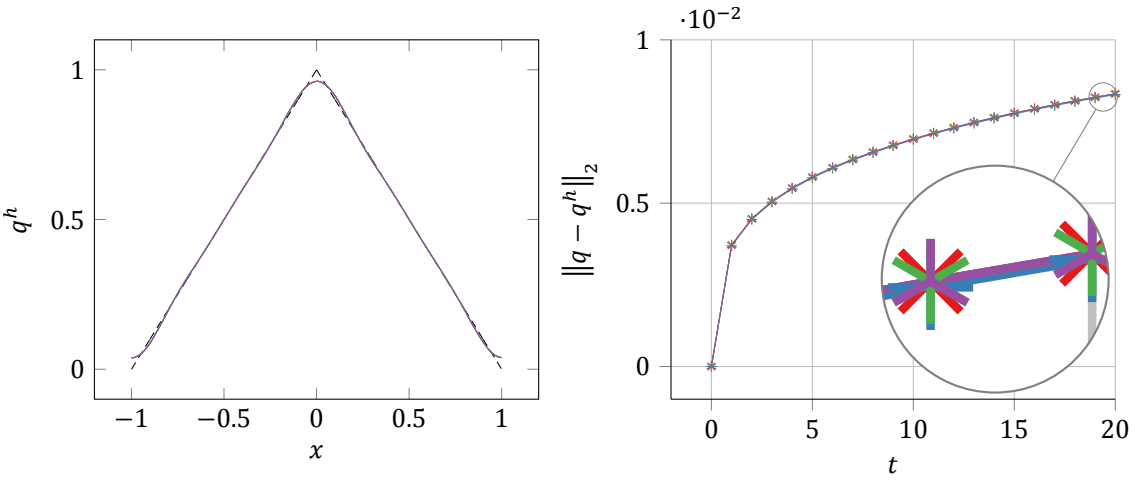
[1] I postpone all numerical experiments involving limiters for the Euler equations until chapter 14.
[2] A posteriori modified wavenumber analysis is presented in [48] for finite volume schemes. Nevertheless, its generalization to DG-like schemes is straight-forward.
[3] https://nl.mathworks.com/help/matlab/ref/fft.html



(a) Monochromatic wave, $n = 7$

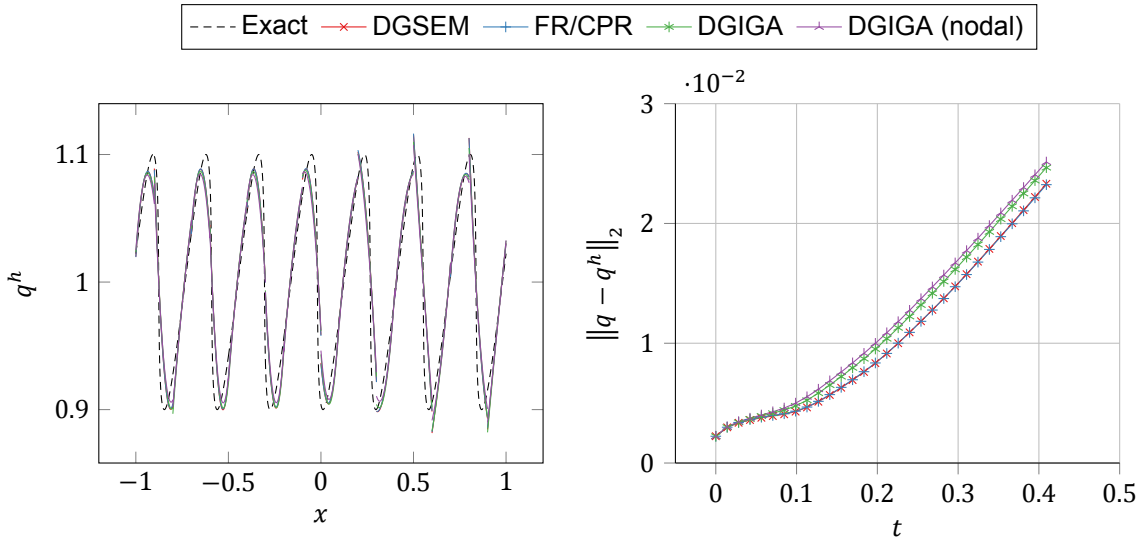


(b) Gaussian hump

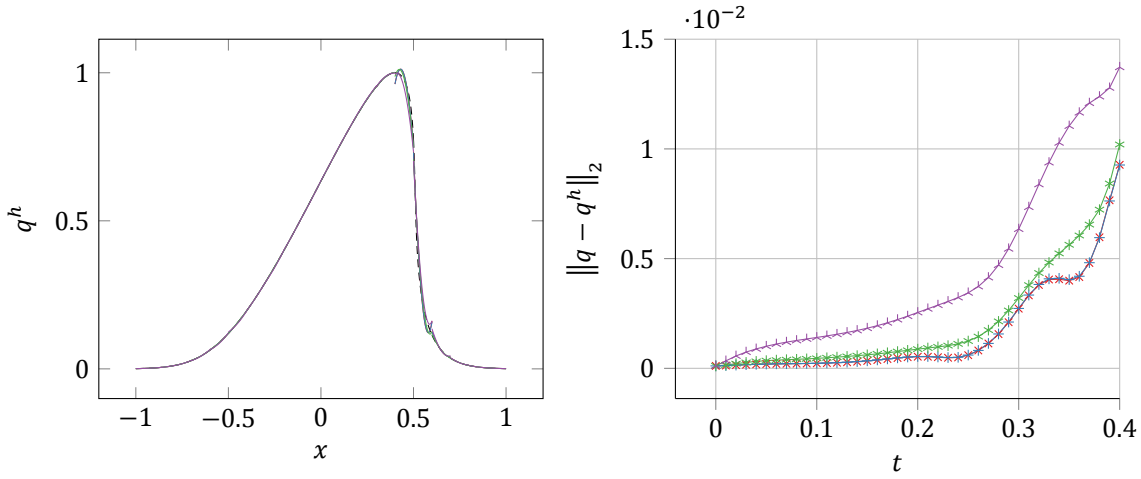


(c) Triangular pulse

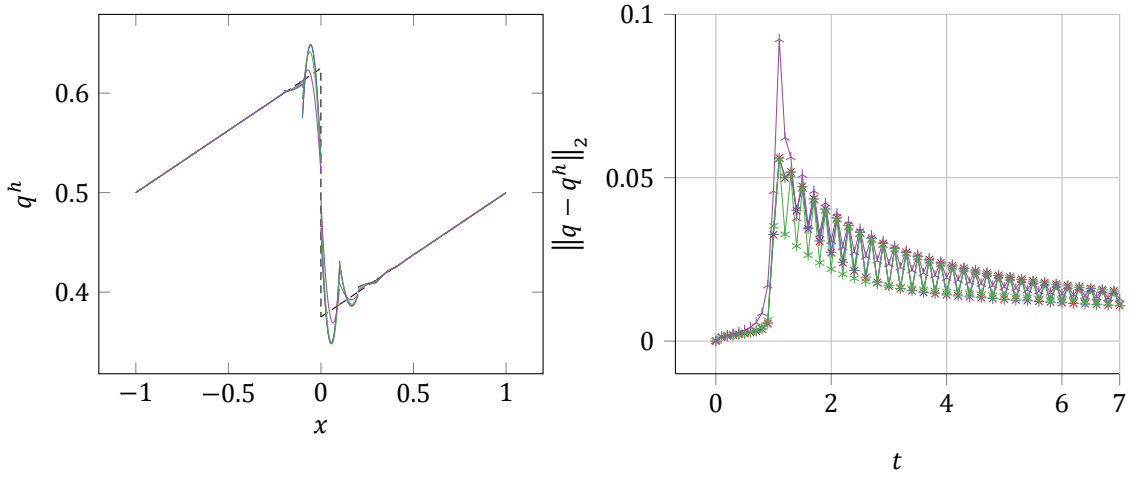
Figure 13.4: Solution at $t = \Delta T$ (left) and L^2 norm of the error as a function of time (right) of all $\mathrm{K} = 20 \Leftrightarrow \mathrm{J} = 3$ runs in test matrices 13.1 and 13.2.



(a) Monochromatic wave, $n = 7$

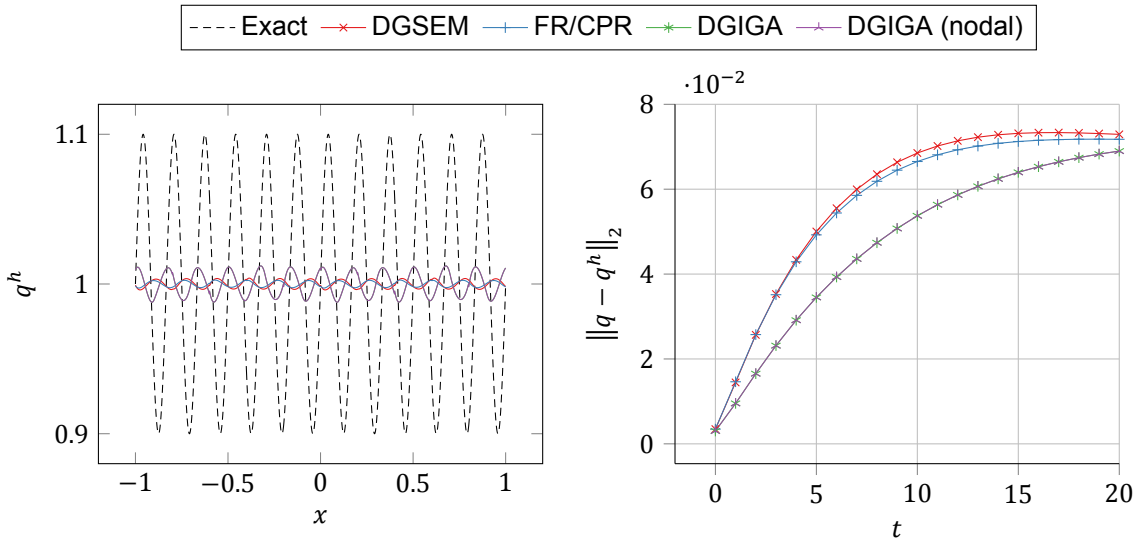


(b) Gaussian hump

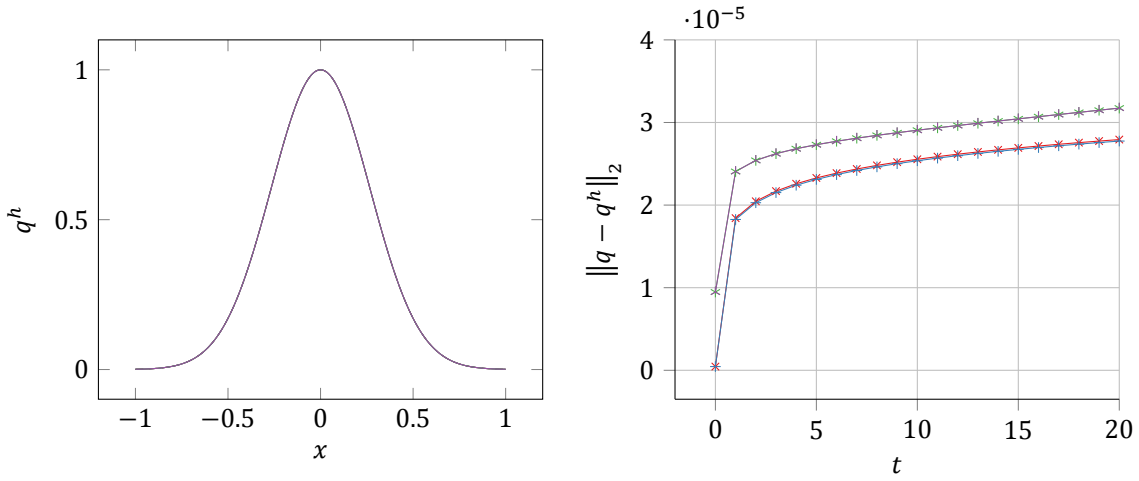


(c) Triangular pulse

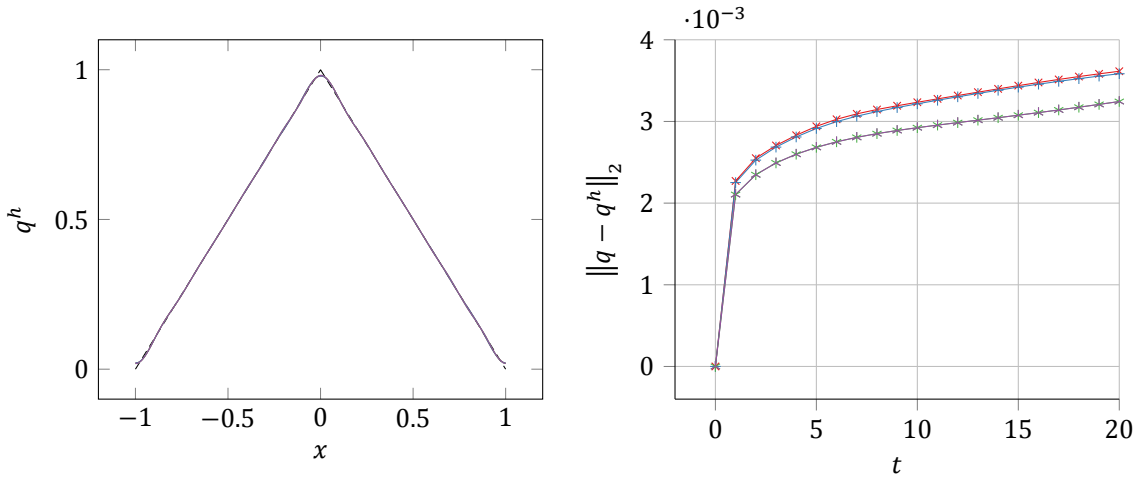
Figure 13.5: Idem for test matrix 13.3.



(a) Monochromatic wave, $n = 12$

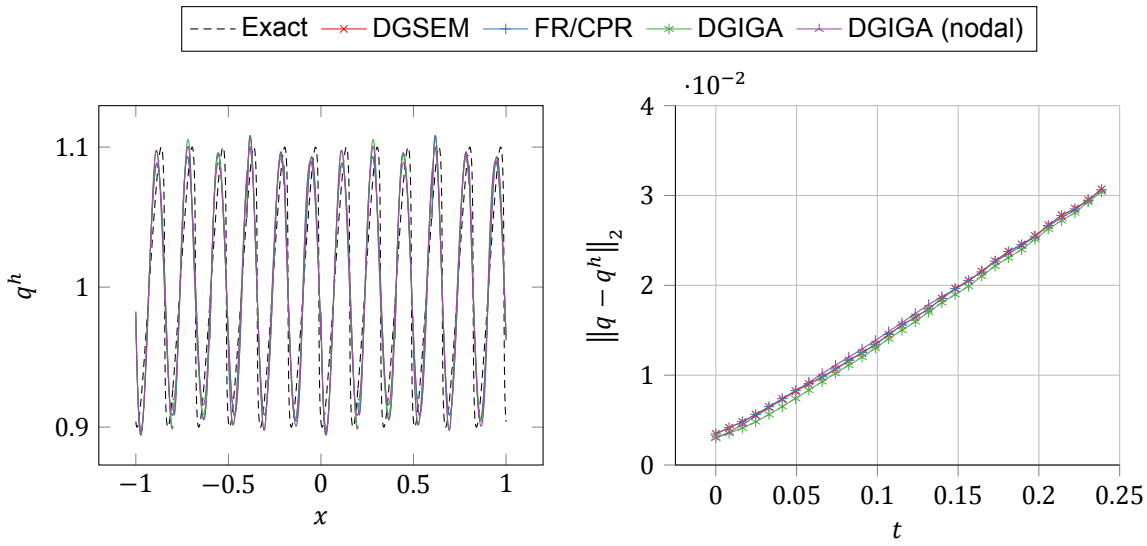


(b) Gaussian hump

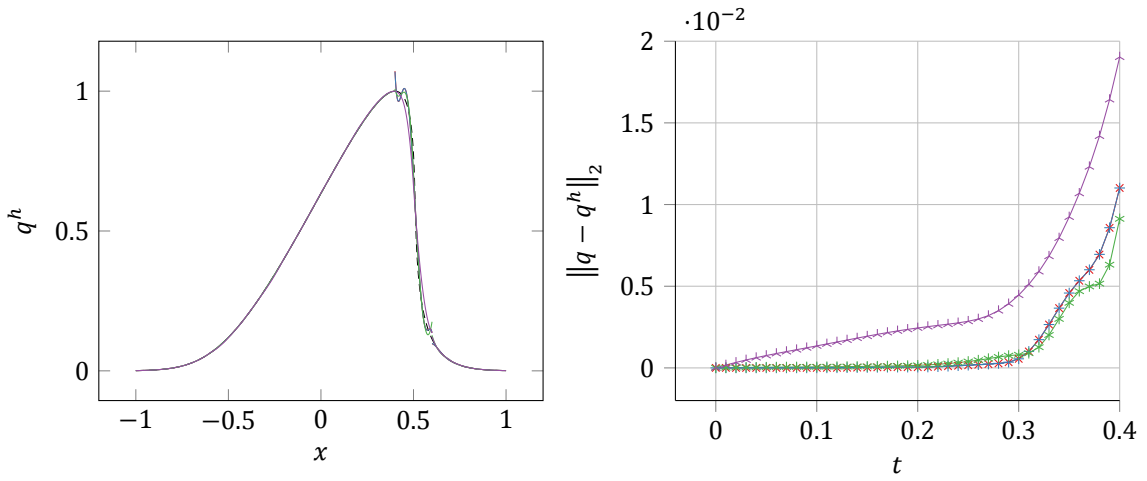


(c) Triangular pulse

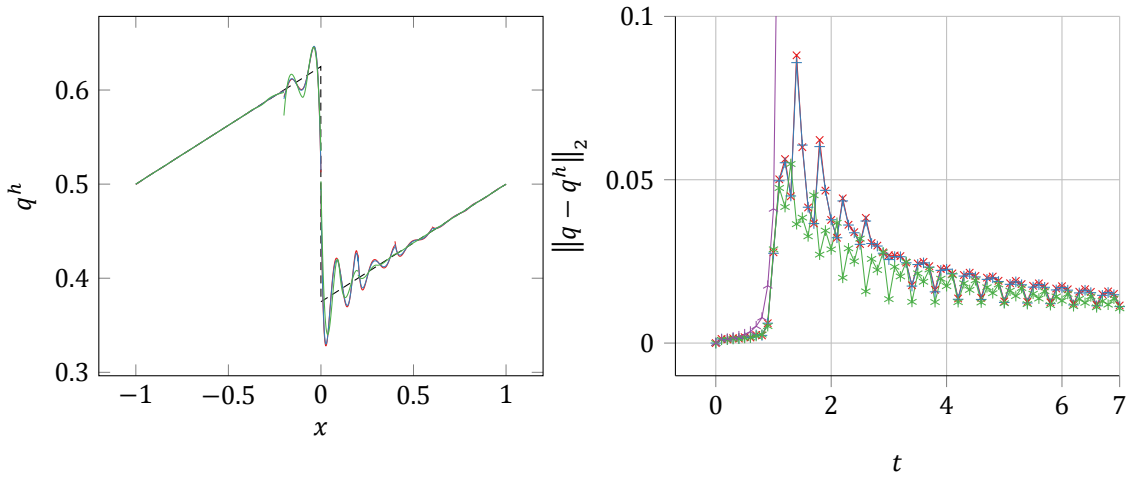
Figure 13.6: Idem, for all $\mathrm{K} = 10 \iff \mathrm{J} = 6$ runs of test matrices 13.1 and 13.2.



(a) Monochromatic wave, $n = 12$

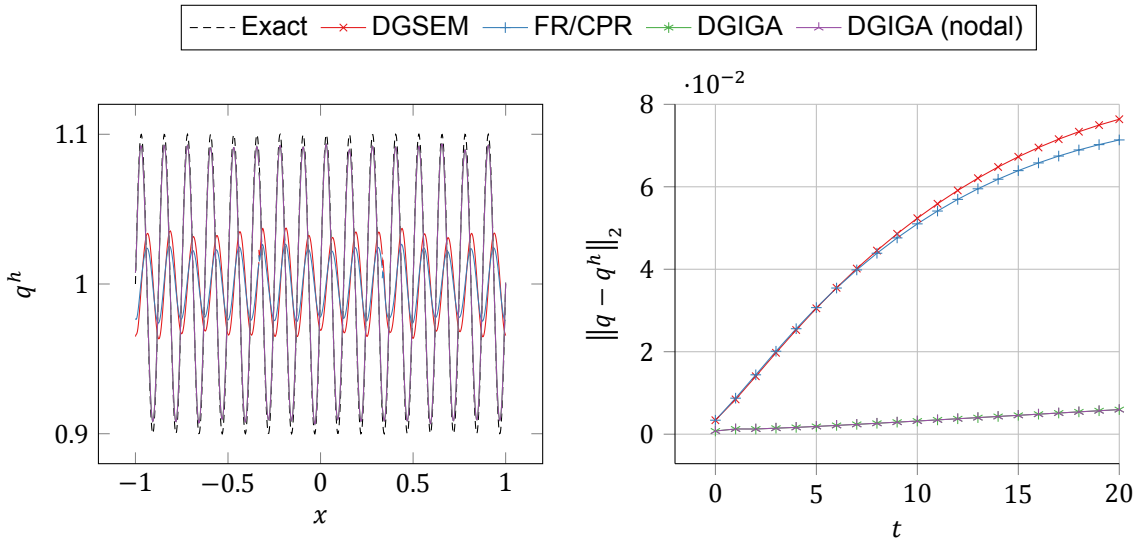


(b) Gaussian hump

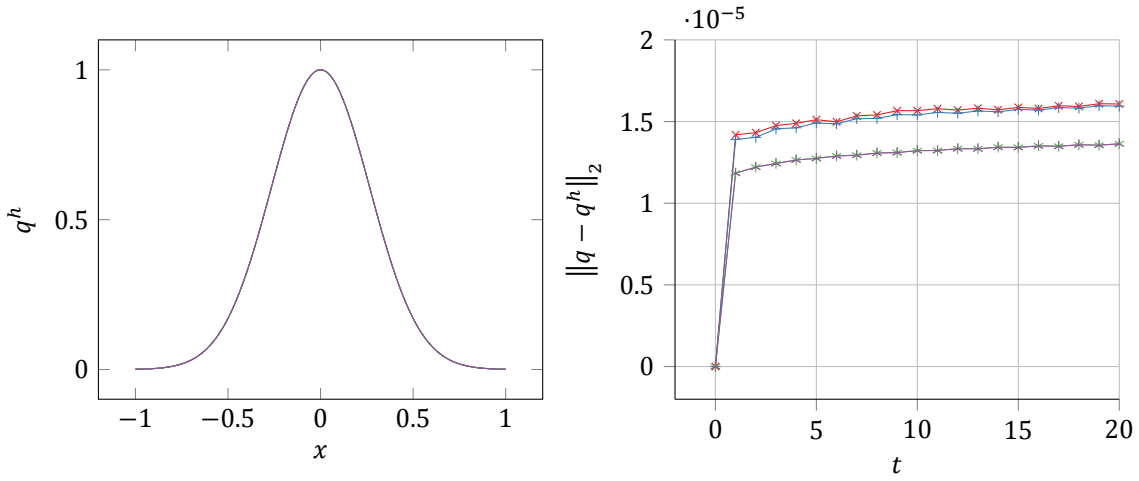


(c) Triangular pulse

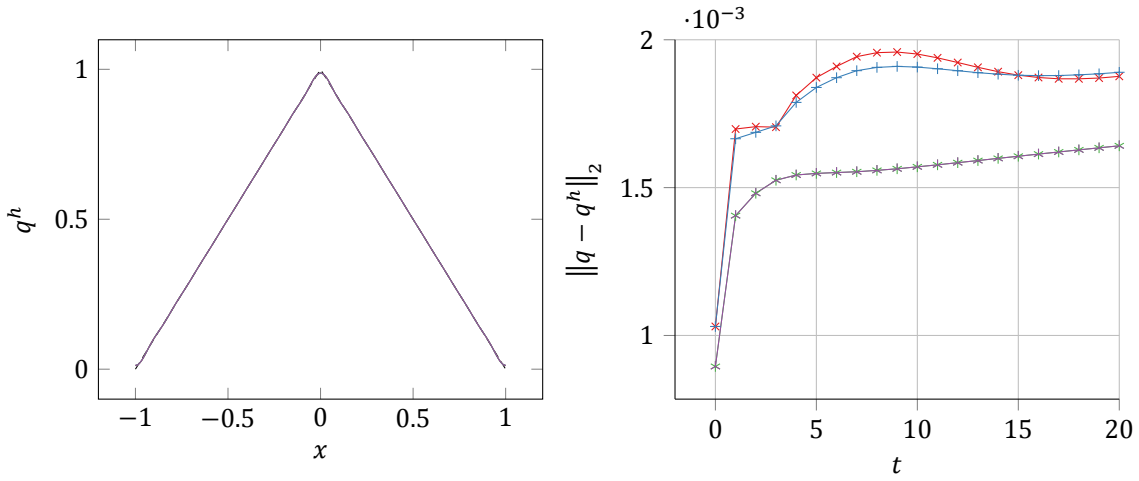
Figure 13.7: Idem for test matrix 13.3.



(a) Monochromatic wave, $n = 16$

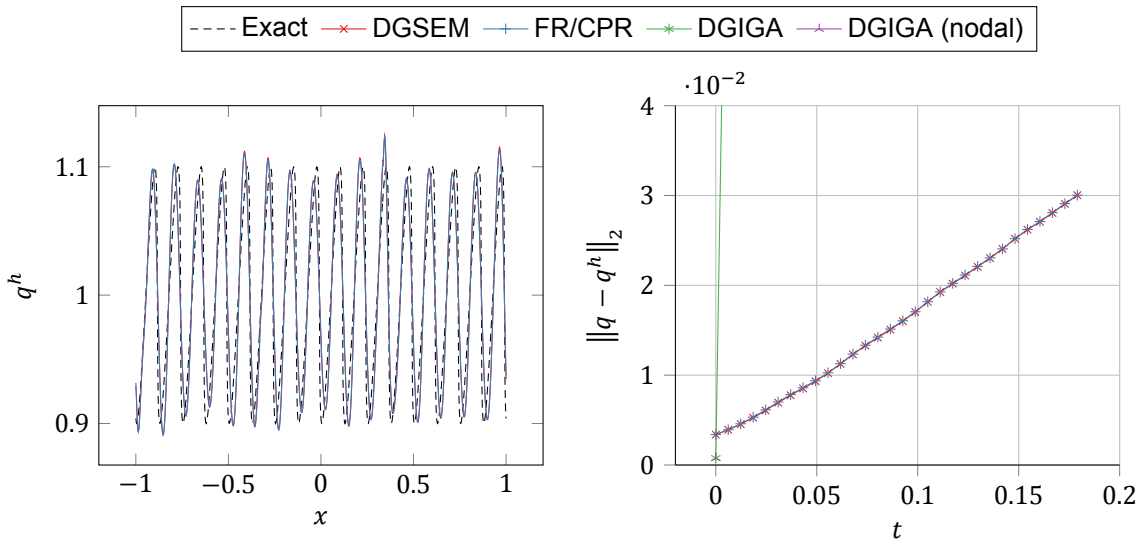


(b) Gaussian hump

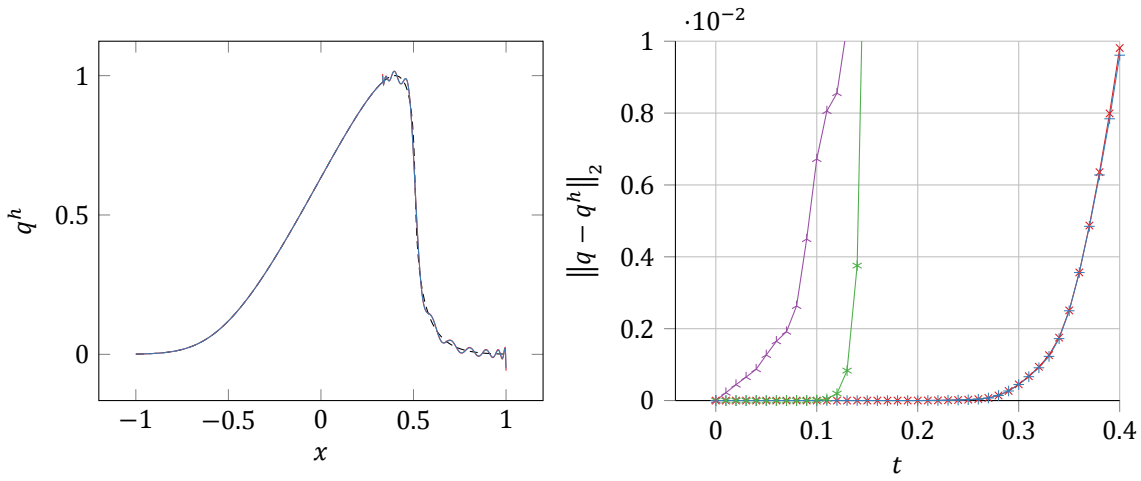


(c) Triangular pulse

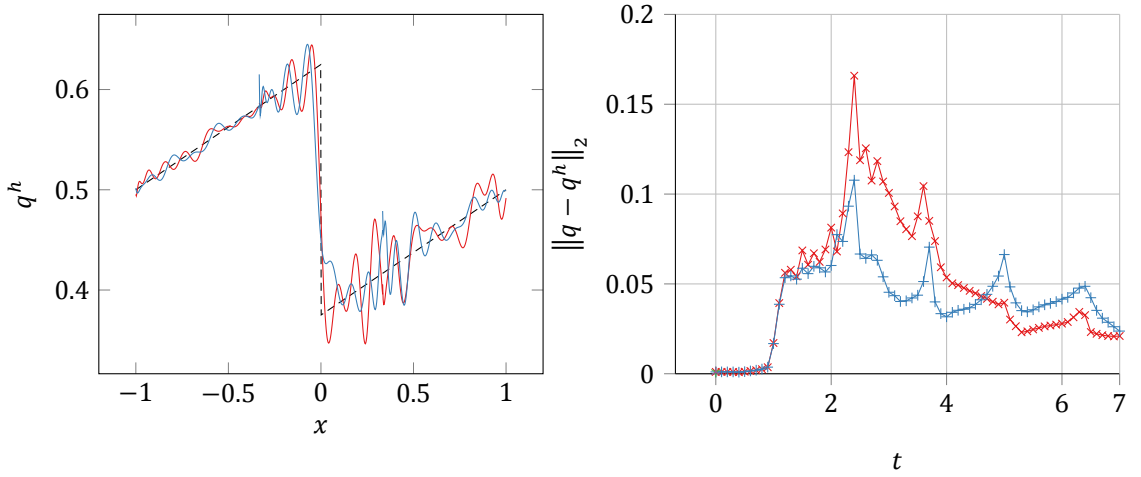
Figure 13.8: Idem, for all $\mathrm{K} = 3 \iff \mathrm{J} = 20$ runs of test matrices 13.1 and 13.2.



(a) Monochromatic wave, $n = 16$



(b) Gaussian hump



(c) Triangular pulse

Figure 13.9: Idem for test matrix 13.3.

13.2.2. A posteriori vs. combined-mode analyses

The comparisons in figure 13.10 strongly suggest that a posteriori modified wavenumber analysis is not a good predictor of the long-term spectral characteristics of high-order DG-like schemes. After all, this procedure attempts to characterize the dispersion and dissipation of the spatial scheme via a single (complex-valued) function, analogously to the analytical *a priori* approach used extensively in chapter 11. I would argue, therefore, that is has the same fundamental problem as the latter: for high-order schemes, a single modified wavenumber can not be expected to encode these properties accurately [5]. Of course, this conclusion is only supported by the aforementioned numerical evidence as long as combined-mode analysis *is* representative of the true behaviour of the scheme (this is supported by the literature [5, 117]).

There are, nevertheless, three details worth pointing out. First, that a priori (combined-mode) and a posteriori approaches coincide for the lowest-order case (figure 13.10a); this is so because, in the $\mathrm{J} = 1$ case, there is no multiplicity of eigenmodes. Second: they differ more strongly the higher p becomes (the more basis components, the larger the discrepancy). And, third, that the two predictions become closer to each other as $t^* \to 0$, especially for $\kappa \approx \kappa_f$, for $\mathrm{p} > 0$.

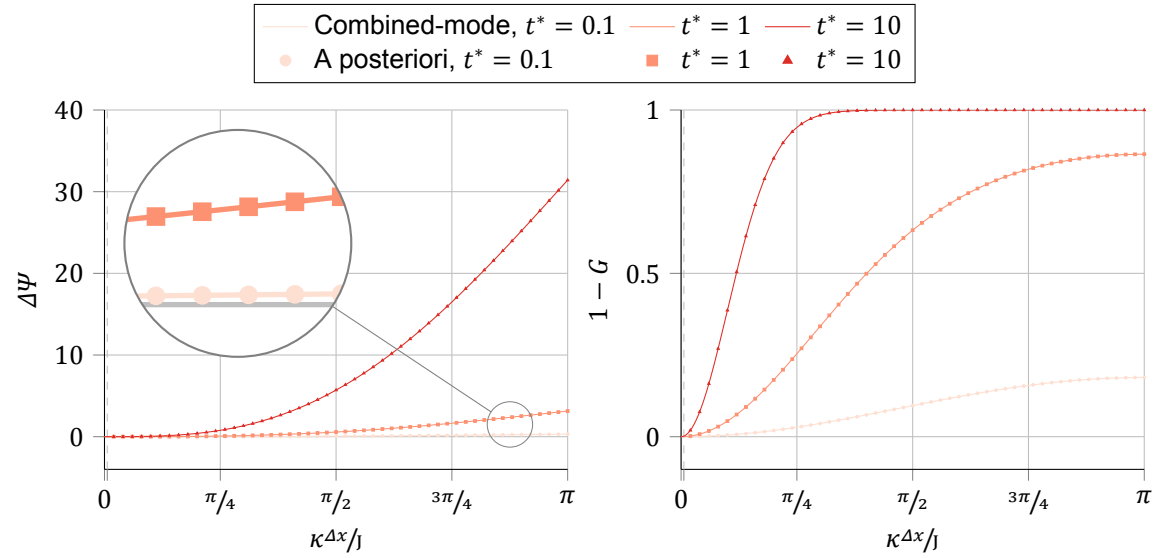
13.2.3. Spatial schemes

Figure 13.11 shows a posteriori dispersion and dissipation relations of DGSEM, FR/CPR and DGIGA of a given J. According to the conclusions of the previous subsection, I do not consider these results representative of the long-time behaviour of the schemes considered. That being said, they are actually consistent with those of the linear case.

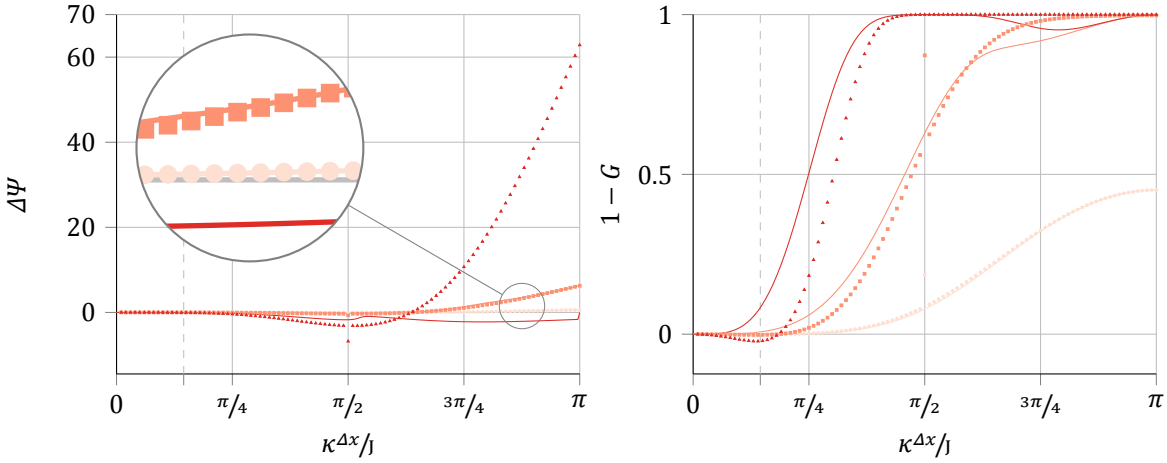
13.2.4. Riemann solvers

Figure 13.12 shows a new set of results of a posteriori modified wavenumber analysis, this time for a number of different Riemann solvers of the Euler equations all used in conjuction with $\mathrm{p} = 5$ DGSEM. This comparison can be interpreted as a direct measure of the amount of numerical dispersion/dissipation introduced by each Riemann solver in one time-stage; therefore, the concerns about the reliability of the a posteriori approach are mitigated—it is not the long-time behaviour of the discretization that is being assessed, but the relative effect per time-stage of each solver.

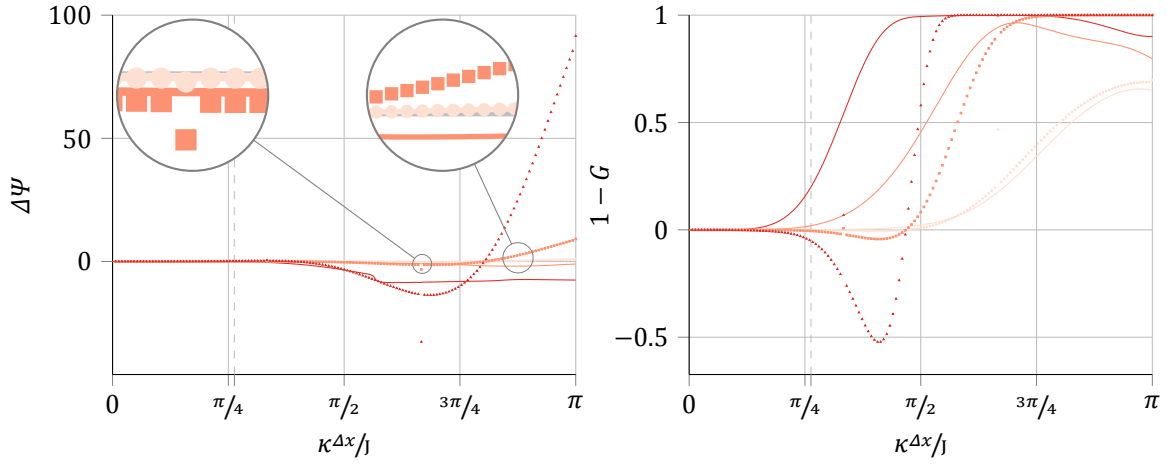
Taking these results at face value, I would place HLLC as the optimal Riemann solver, due to it possessing the second-lowest dispersion and lowest (nonzero) dissipation of the set, in addition to being simple and efficient (it is as accurate as Roe's and exact solvers, yet these two require an entropy fix or/and are not as cost-effective). Figure 13.12 can also be used to justify the following fallback strategy: should HLLC fail to produce a stable solution, the next best option would be HLL, then HLLE, and, as a last resort, LLF. The central numerical flux (KEP), which renders the discretization completely nondissipative, is not of interest in this work because numerical dissipation (at high wavenumbers) is a *desirable feature* in scale-resolving flow simulations (see §12.3).



(a) p = 0

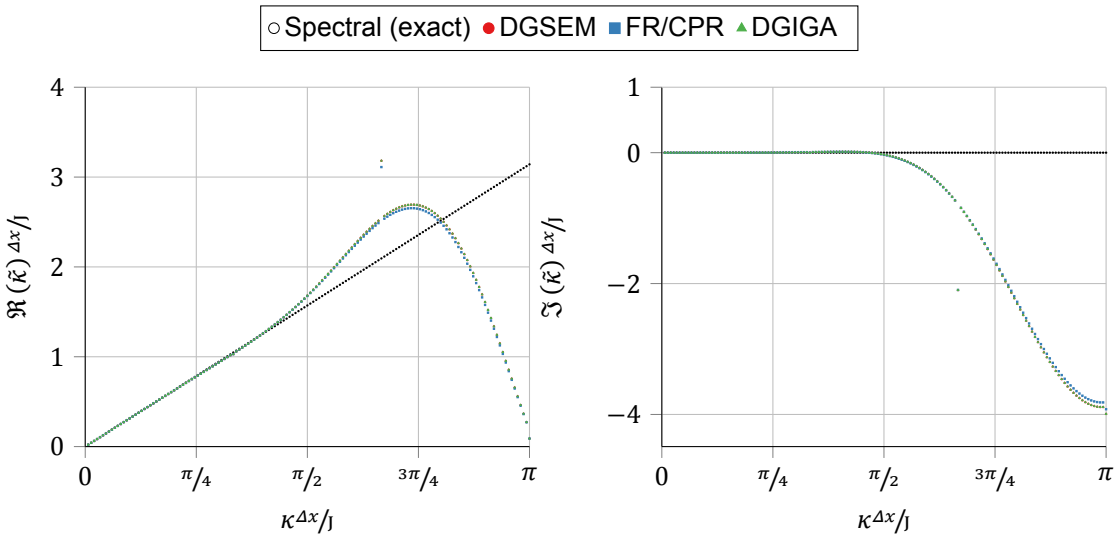


(b) p = 1

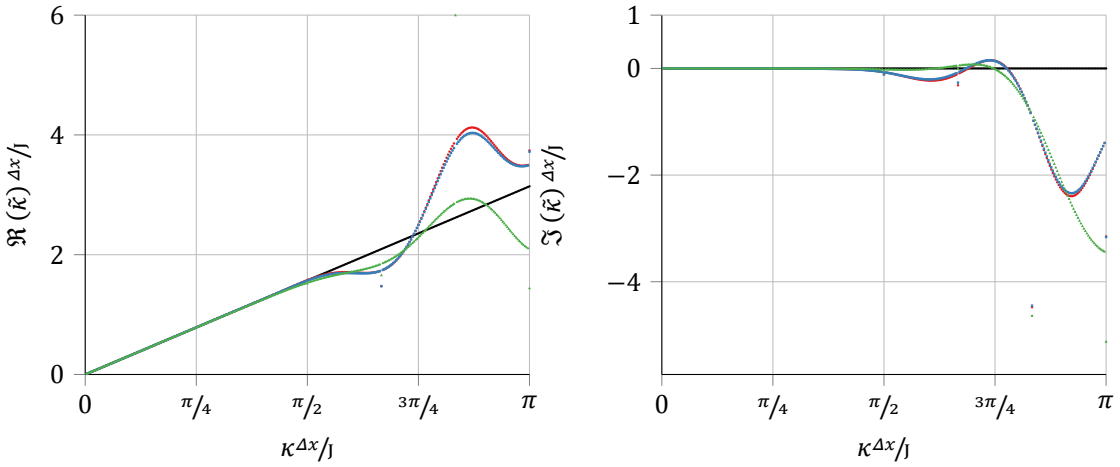


(c) p = 2

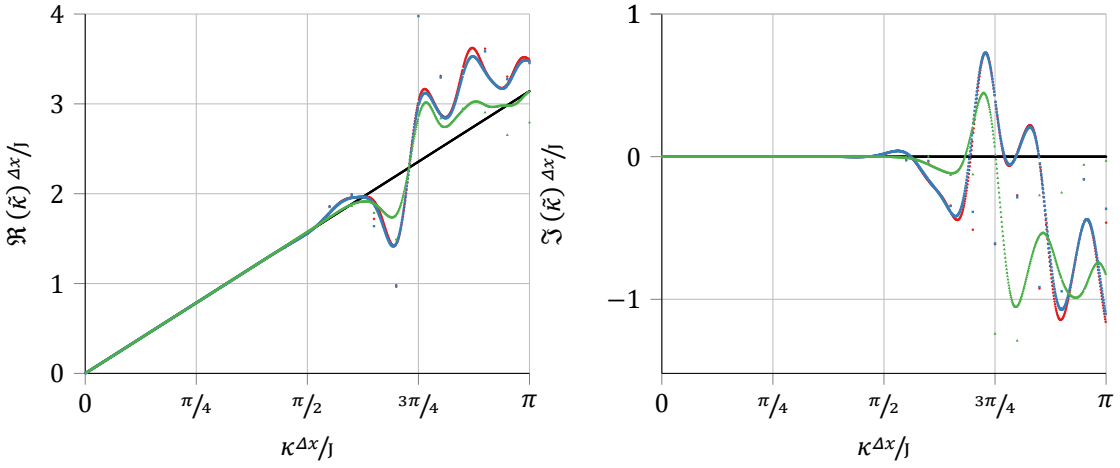
Figure 13.10: Predicted errors in dispersion (phase angle, left) and dissipation (damping factor, right) for DGSEM of degree p, according to both *a posteriori* modified wavenumber analysis (markers) and combined-mode semidiscrete analysis (lines, see §A.2.5). Dashed grid lines indicate $\kappa = \kappa_f$.



(a) J = 3



(b) J = 6



(c) J = 20

Figure 13.11: Dispersion (left) and dissipation (right) relations obtained from *a posteriori* modified wavenumber analysis, for DGSEM and the optimized schemes from §12 in combination with the exact Riemann solver.

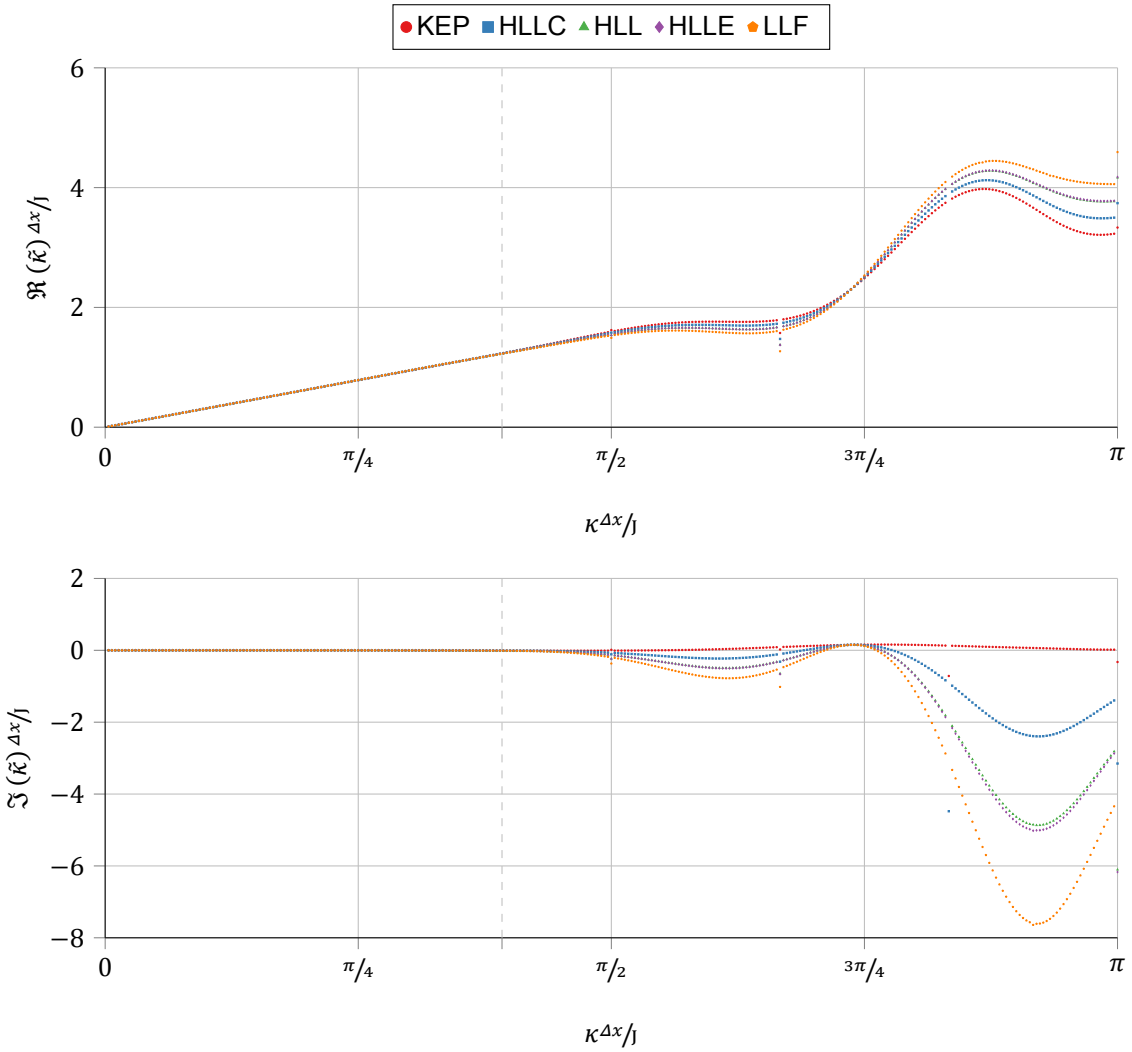


Figure 13.12: Idem to figure 13.11b, for DGSEM (of degree 5) in combination with various Riemann solvers (see §2.8.2 for a brief description of each). I do not show neither exact nor Roe's (with and without entropy fix) because they are indistinguishable from HLLC. Consistent results can be obtained for other J, as well as FR/CPR and DGIGA schemes (not shown). A dashed grid line separates well-resolved and underresolved regimes, in each case.

14

Non-smooth Solutions, Sensors and Limiters

The aim of this chapter is to judge the extend to which each of the types of schemes under consideration are able to deal with problems the exact solution of which contains discontinuities. My approach of doing so has been to test several promising combinations of limiters, sensors and discretization bases in a battery of test cases which I myself have selected from the literature. The hope is that, when analyzed together, any conclusions extracted from their results will be reasonably representative of each studied configuration.

This last set of numerical experiments is split into a total of 7 batches, each corresponding to a separate section. In §14.1, I compare (for DGSEM only) the performance of the Legendre-based limiters introduced in §§8.4.3 and 8.5. The most successful of those is then coupled, in §14.2, with each of the two sensors described in §8.3, and each combination is compared with the limiter used alone (without any sensor). Section 14.3 focuses on the AFC/FCT limiter (applicable to DGIGA-AFC only), with and without sensors, and compares its results to those of the best hierarchical limiter applied to DGIGA. These DGIGA-AFC results are complemented with those of §14.4, which instead considers IGA-AFC. A final comparison (of sorts) between DGSEM, FR/CPR, DGIGA and DGIGA-AFC bases and their best limiter/sensor combinations is done in §14.5. Section 14.6 adds to the results of the said final comparison by explicitly testing the influence of the balance between patches and breakpoint spans (i.e. between DG and CG elements) in FCT-limited DGIGA-AFC. Finally, §14.7 closes the chapter by exploring the relationship between high order discretizations and limiters. In it, first, I explicitly test which of the limiters considered preserve the order of the discretization at smooth extrema (often considered the most basic requirement of any successful limiting strategy in the literature); then, I use an example to discuss whether that really matters in practice.

In all figures of this chapter, the left vertical axis represents either the scalar unknown of the advection equation, or the density in the Euler case. Element and patch interfaces are indicated with minor grid lines (when applicable). Its approximate solution is represented with solid lines, each color associated to a particular run of the current batch (see each figure's legend). The right vertical axis, when shown, corresponds to the *instantaneous* limiter activation ratio—denoted $P_{\text{limiter}}(q)$—which measures how many, out of all degrees of freedom of quantity q in an element/patch, have been limited[1]. This quantity is meant to indicate *where* limiting is being applied at any given instant, and at what strength. It is represented in the figures by a single point located at the centroid of each element/patch; each run has a different marker type to make visualization possible.

A more representative measure of *how much* limiting has been applied overall is the *mean* limiter activation ratio. I define it as:

$$\overline{P}_{\text{limiter}} \coloneqq \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{\mathrm{I}} P_{\text{limiter}}^{(n)}(q_i) \ , \tag{14.1}$$

[1] Note that with hierarchical and slope limiters the maximum number of limitable degrees of freedom per element/patch is $\mathrm{J}-1$ (the 1st Legendre coefficient, associated with the element-wise mean, is never modified); for these, therefore, $P_{\text{limiter}} < 1$. For AFC/FCT, since all J control values may be modified, $P_{\text{limiter}} \leq 1$.

where N denotes the total number of times the limiter has been applied up to the measured instant (usually once per time-scheme stage). This quantity is displayed in all tables of this chapter, together with the L^1 norm of the error (when applicable), the total variation of the approximate solution component (or its ratio with that of the exact solution, if available) and the wall-clock time required for the run to complete[2]. I define the L^1 norm as:

$$\left\| q(t,x) - q^h(t,x) \right\|_1 \coloneqq \frac{\int_\Omega \left| q(t,x) - q^h(t,x) \right| \, \mathrm{d}x}{\int_\Omega \, \mathrm{d}x} \,. \tag{14.2}$$

The integral in the numerator is approximated in the same manner as in (10.1).

14.1. Hierarchical limiters

Test matrix 14.1 summarizes the runs intended to reveal whether any one of the limiters of §8.5 is clearly superior to the rest. These results include two baseline configurations: that of no limiter (which results in oscillations due to the Gibbs phenomena) and that associated with the TVB slope limiter (§8.4.3), used in its TVDM mode. On top of that, all runs are first fail-safe limited as in §8.8.1, and then as in §8.8.2. Note that the time-step size indicated in 14.1 is half of $\Delta t_{\max}$; this indicates that the Courant number is fixed (and not the time-step size itself), at a value equal to $\varsigma = \varsigma_{\max}/2$—i.e. half of the maximum linearly stable Courant number for the basis in question, as predicted from linear stability analysis (see chapter 11 and §A.4).

Results of runs 1 to 12, both qualitative and quantitative, show that Krivodonova's limiter is superior to the rest for the Jiang-Shu problem (9.12), displaying a clear advantage at smooth extrema (I focus on this particular aspect in §14.7). Also distinguished, but in the opposite sense, is the behavior of the HWENO limiter (§8.6): it starts with sub-par performance for the $\mathrm{p} = 2$ discretization, and becomes completely outclassed, even by the TVB limiter, when the degree is increased to $\mathrm{p} = 5$. The other two hierarchical limiters, BDF (§8.5.1) and BSB (§8.5.2), are tied in second place with the former seemingly being superior near discontinuous features yet slightly inferior in smooth ones—which is interesting, as that the latter is considered an improvement over the former in the literature.

The advantage of krivodonova's limiter is not as clear in the rest of test cases. I attribute this to these other problems having relatively short simulated time spans and not as rich solutions. Interestingly, the unlimited case is able to resolve contact discontinuities much more sharply than any of the limiters, in all test cases, and even more so the higher the degree. In fact, despite its spurious oscillations, the unlimited approximate solution is *more* accurate in the L^1 norm than any of the limited ones for the Jiang-Shu and Toro's shock tube problems, for both $\mathrm{J} = 3$ and $\mathrm{J} = 6$.

Table 14.1: Comparing inter-cell limiters used in combination with DGSEM.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N_{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.2a	(9.12)	4(10)	$\Delta t_{\max}/2$	8	DGSEM	300	100	2	-	-	∞	-	-
2													TVB, M = 0 (§8.4)	-
3													BDF (§8.5.1)	-
4													BSB (§8.5.2)	-
5													Krivodonova (§8.5.3)	-
6													HWENO (§8.6)	-
7	14.3a						600		5				-	-
8													TVB, M = 0	-
9													BDF	-
10													BSB	-
11													Krivodonova	-
12													HWENO	-
13	14.2b	(9.13)			0.2		300		2				-	-
14													TVB, M = 0	-

(continues in the next page)

[2] I need to insist that these timings may not be representative at all of the actual computational cost associated with the schemes, as my implementation has not been designed for performance. A slightly more rigorous (albeit still imperfect) cost estimation is provided in §12.4 (see also appendix B).

Table 14.1: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
15													BDF	-
16													BSB	-
17													Krivodonova	-
18													HWENO	-
19	14.3b						600		5				-	-
20													TVB, M = 0	-
21													BDF	-
22													BSB	-
23													Krivodonova	-
24													HWENO	-
25	14.2c	(9.14)			0.15		300		2				-	-
26													TVB, M = 0	-
27													BDF	-
28													BSB	-
29													Krivodonova	-
30													HWENO	-
31	14.3c						600		5				-	-
32													TVB, M = 0	-
33													BDF	-
34													BSB	-
35													Krivodonova	-
36													HWENO	-
37	14.2d	(9.15)			0.038		600	200	2				-	-
38													TVB, M = 0	-
39													BDF	-
40													BSB	-
41													Krivodonova	-
42													HWENO	-
43	14.3d						1200		5				-	-
44													TVB, M = 0	-
45													BDF	-
46													BSB	-
47													Krivodonova	-
48													HWENO	-

14.2. Sensors

Having established Krivodonova's limiter to be the best among those tested, this next batch is aimed at doing the same for the two sensors under consideration. To do so, I try each of them in combination with the aforementioned limiter, for the same set of test cases. The unlimited case is repeated from §14.1.

This time, however, the comparison remains inconclusive: no sensor is consistently better than the other, nor than using none at all. Therefore, I include both sensors in the comparisons of §§14.3 and 14.5. Purely from an activation perspective (see table 14.9), both sensors are able to reduce the limiter usage (with KXRCF being very effective for $p = 2$, but seemingly conceding its advantage to AP-TVD for $p = 5$). Nevertheless, the actual improvement that this has on accuracy is barely noticeable.

That being said, results in §14.5 do seem to reveal KXRCF as the superior choice, in terms of precise location of discontinuities (although any improvement on accuracy remains minimal), for the Shu-Osher test case at least (which has not been part of the current batch).

Table 14.2: Comparing sensors, against each other and none at all, for DGSEM.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.4a	(9.12)	4(10)	$\Delta t_{max}/2$	8	DGSEM	300	100	2	-	-	∞	Krivodonova	-
2														KXRCF (§8.3.1)
3														AP-TVD (§8.3.2)

(continues in the next page)

Table 14.2: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
4	14.5a						600		5					-
5														KXRCF
6														AP-TVD
7	14.4b	(9.13)			0.2		300		2					-
8														KXRCF
9														AP-TVD
10	14.5b						600		5					-
11														KXRCF
12														AP-TVD
13	14.4c	(9.14)			0.15		300		2					-
14														KXRCF
15														AP-TVD
16	14.5c						600		5					-
17														KXRCF
18														AP-TVD
19	14.4d	(9.15)			0.038		600	200	2					-
20														KXRCF
21														AP-TVD
22	14.5d						1200		5					-
23														KXRCF
24														AP-TVD

14.3. DGIGA-AFC

Next is the turn of the AFC/FCT limiting approach. I consider, in this section, the optimal DGIGA bases of 3 and 6 basis functions per patch (see §12.1) and compare the results of using no limiter, Krivodonova's limiter, and the FCT limiter (with each sensor as well as with none). All runs employ a modal treatment of DGIGA (6.14). Also, all runs use the hierarchical fail-safe limiters (as in previous batches) but FCT ones additionally employ the FCT fail-safe (§8.8.4); moreover, the latter apply the hierarchical fail-safe on control values and both after every step (i.e. on the predictor) and stage (i.e. after the FCT correction).

There are two observations to be made about these batch's results. First, DGIGA-AFC with FCT limiting is very clearly outclassed by DGIGA coupled with Krivodonova's sensor in terms of how diffused its numerical solution ends up. This is most striking in the Jiang-Shu problem (the initial condition is no longer recognizable after crossing the domain 4 times), but is also present in the rest of test cases in the form of a less sharp capture of discontinuities and kinks. A possible explanation for this excessive diffusion might be found in the relatively large jumps that can be seen to exist across patch interfaces in all AFC runs (recall that the only mechanism capable of introducing numerical diffusion in DG is the Riemann solver used at every element or patch interface [120]).

And second: unlimited DGIGA experiences what appears to be a very strong entropy shock in the expansion region at the left of Toro's shock tube. This was not present in any of the unlimited DGSEM cases; its cause must associated with some particularity of the B-spline basis itself. Note that this occurs even in the $J = 3$ run, for which DGIGA reduces to 3rd order, Bernstein polynomial-based, DG. I am inclined to believe that this is another symptom of the same underlying issue that caused the unexplained stability issues in the Burgers equation, in §10.4. Furthermore, notice how the position of the right-most shock in the numerical solution of runs 36 and 37 is clearly biased to the left. I am not able to provide an explanation for this either; perhaps it is yet another manifestation of the previously mentioned issue.

Table 14.3: Comparing AFC against the best inter-cell limiter, for DGIGA.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.6a	(9.12)	4(10)	$\Delta t_{max}/2$	8	DGIGA	300	100	2	-	1	∞	-	-
2													Krivodonova	
3						DGIGA-AFC							FCT (§8.7.2)	

(continues in the next page)

Table 14.3: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
4														KXRCF
5														AP-TVD
6	14.7a					DGIGA	600		3		2	1	-	-
7													Krivodonova	
8						DGIGA-AFC							FCT	
9														KXRCF
10														AP-TVD
11	14.6b	(9.13)			0.2	DGIGA	300		2		1	∞	-	-
12													Krivodonova	
13						DGIGA-AFC							FCT	
14														KXRCF
15														AP-TVD
16	14.7b					DGIGA	600		3		2	1	-	-
17													Krivodonova	
18						DGIGA-AFC							FCT	
19														KXRCF
20														AP-TVD
21	14.6c	(9.14)			0.15	DGIGA	300		2		1	∞	-	-
22													Krivodonova	
23						DGIGA-AFC (6.36)							FCT	
24														KXRCF
25														AP-TVD
26	14.7c					DGIGA	600		3		2	1	-	-
27													Krivodonova	
28						DGIGA-AFC (6.36)							FCT	
29														KXRCF
30														AP-TVD
31	14.6d	(9.15)			0.038	DGIGA	600	200	2		1	∞	-	-
32													Krivodonova	
33						DGIGA-AFC							FCT	
34														KXRCF
35														AP-TVD
36	14.7d					DGIGA	1200		3		2	1	-	-
37													Krivodonova	
38						DGIGA-AFC							FCT	
39														KXRCF
40														AP-TVD

14.4. IGA-AFC

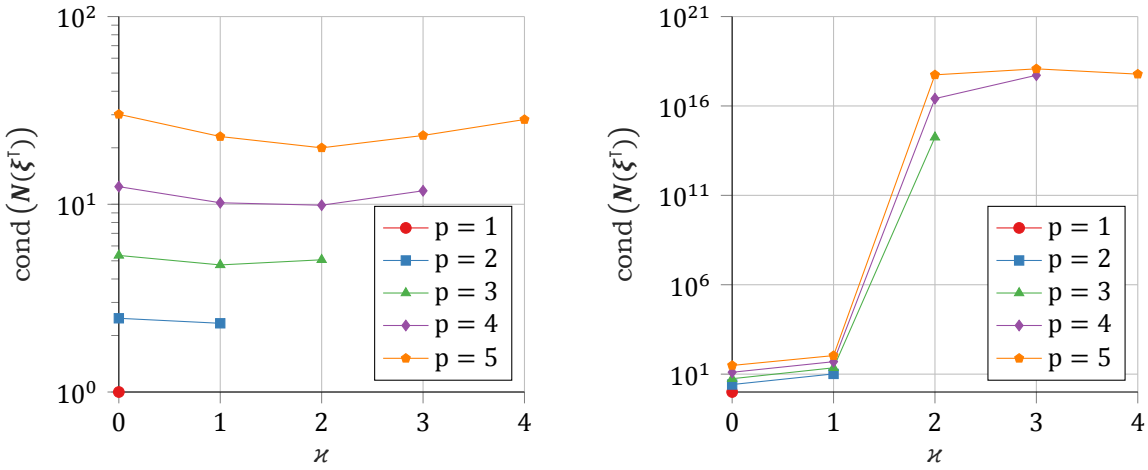
In §14.3, AFC limiting turned out to be sub-par due to its very high numerical diffusion. I attribute this excess of diffusion to the relatively large differences between Riemann problem left and right states. If that is indeed the case, one would expect a reduction of said diffusion the lower the number of patches employed. To confirm this, I now test IGA-AFC at all degrees and smoothnesses up to $\mathrm{p} = 3$. Note in test matrix 14.4 that some runs employ nodal DGIGA; this is because the modal treatment in them is unfeasible, due to either conditioning issues (see figure 14.1) and/or the presence of invalid control point values (in the sense of figure 8.5a). This is the only option to help stabilize them, as IGA is incompatible with inter-cell fail-safe limiting (the entire mesh is one cell), so none of these runs can use it. FCT fail-safe is still employed.

Comparison between figures 14.8a and 14.6a (and associated tables) confirms that IGA-AFC is not as diffusive as DGIGA-AFC for a given total number of degrees of freedom, at least in the case of the latter using a single breakpoint span (i.e. opposite extremes) and $\mathrm{p} = 2$. For this test case, the most accurate basis is that of run 3 (piece-wise polynomials of degree 2, joined at breakpoints in C^1 fashion; modal treatment). In the rest of test cases the difference is less clear, but this can be attributed to their exact solutions being simpler. An interesting deviation is that of run 11 (degree 3, smoothness C^1, modal treatment); in it, the contact discontinuity in Toro's shock tube is markedly more sharply resolved than all other bases. This suggests that the optimal ratio between degree and smoothness in terms of discontinuous features might not be at either extreme (neither C^0 nor $C^{\mathrm{p}-1}$). I do not pursue this line of

research further in the present work.

Table 14.4: Attempting IGA-AFC, the extreme case of DGIGA with a single patch.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.8a	(9.12)	4(10)	10^{-2}	8	DGIGA-AFC	300	1	1	-	299	0	FCT	-
2							301		2		150			
3							300				298	1		
4							301		3		100	0		
5							300				149	1		
6						(nodal)					297	2		
7	14.8b	(9.13)		10^{-3}	0.2	(modal)			1		299	0		
8							301		2		150			
9							300				298	1		
10							301		3		100	0		
11							300				149	1		
12						(nodal)					297	2		
13	14.8c	(9.14)			0.15	(modal; 6.36)			1		299	0		
14							301		2		150			
15							300				298	1		
16							301		3		100	0		
17							300				149	1		
18						(nodal; 6.36)					297	2		
19	14.8d	(9.15)		10^{-4}	0.038	(nodal; 6.34)	600		1		599	0		
20							601		2		300			
21							600				598	1		
22							601		3		200	0		
23							600				299	1		
24											597	2		



(a) 2 breakpoint spans

(b) 200 breakpoint spans

Figure 14.1: Condition number of the control Vandermonde matrix (6.13) of an IGA patch of smoothness $C^\varkappa$, for increasing degrees. When the number of breakpoint spans is large, the modal treatment (6.14) becomes unfeasible for $\varkappa > 1$. The nodal formulation (6.15), which does not use this matrix, is unaffected.

14.5. Final comparison

The Shu-Osher problem is a very interesting test case from a practical point of view, as it combines the resolution requirements of e.g. the Jiang-Shu problem with the presence of discontinuities and nonlinear physics. In it, an ideal method would resolve the smooth features accurately, while capturing the shock sharply yet without spurious oscillations. This section continues with the direct comparison between

the two optimal bases (FR and DGIGA) and DGSEM (see chapters 12 and 13), now incorporating the usage of limiters and sensors. As indicated in test matrix 14.5, all runs use the same total number of degrees of freedom.

Except for run 35, results of each of the three bases limited via Krivodonova's scheme essentially coincide for a given J and sensor. The deviation of run 35 is due to the action of the inter-cell fail-safe. This suggests that limiting destroys all of the inherent advantage in terms of lowered dispersion and dissipation errors studied in chapter 12. I insist on this issue in section 14.7.

The FCT limiter, applied to DGIGA-AFC, remains clearly inferior to the hierarchical one for $\mathrm{J} = 3, 6$. For $\mathrm{J} = 20$ the situation reverses, but only because all four discretizations turn out to be exceedingly inaccurate. Note how the higher J, the more diffused the approximate solution becomes at the left of the shock. This is because, having fixed $\mathrm{N}_{\mathrm{dofs}}$, a higher J implies fewer yet larger patches. Once the shock reaches a given element/patch, it forces the limiter to reduce the order of the approximation locally. Inevitably, then, any information associated with the higher modes is lost; the element size becomes the dominant factor influencing accuracy.

Interestingly, the KXRCF sensor happens to be very effective with respect to both AP-TVD and the control case for all $\mathrm{J} = 3$ and $\mathrm{J} = 6$ runs, targeting only those elements actually near to the highest gradients in the approximate solution and reducing $P_{\mathrm{limiter}}(q_1)$ and $\overline{P}_{\mathrm{limiter}}$ very significantly. This breaks the ambiguity of the results of section 14.2, bringing me to lean in favor of KXRCF. In fact, it turns out to be cheaper to use this sensor than to use no sensor at all (this can be seen in table 14.12). This is evidence that the overhead of sensing can be compensated by the consequently reduced number of times that the limiter needs to be applied. Notice as well that both sensors are able to reduce the large jumps across patches in the DGIGA-AFC discretization. In the region left of the shock, however, neither of the sensors is able to impact accuracy significantly. Finally, it might be worth highlighting that AFC, despite being very diffusive regardless, does manage to explicitly target only the region around the shock, even for $\mathrm{J} = 20$, suggesting the possibility of using it as a (sub-cell) sensor instead.

Table 14.5: Comparing optimal vs. baseline DG bases, for $\mathrm{J} = 3, 6, 20$.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N}_{\mathrm{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.9a	(9.16)	4(10)	$\Delta t_{\max}/2$	1.8	DGSEM	1200	400	2	-	-	∞	Krivodonova	-
2						FR/CPR				0.023				
3						DGIGA				-				
4						DGIGA-AFC							FCT	
5	14.9b					DGSEM							Krivodonova	KXRCF
6						FR/CPR				0.023				
7						DGIGA				-				
8						DGIGA-AFC							FCT	
9	14.9c					DGSEM							Krivodonova	AP-TVD
10						FR/CPR				0.023				
11						DGIGA				-				
12						DGIGA-AFC							FCT	
13	14.10a					DGSEM		200	5	-		∞	Krivodonova	-
14						FR/CPR				0.058				
15						DGIGA			3	-	2	1		
16						DGIGA-AFC							FCT	
17	14.10b					DGSEM			5		-	∞	Krivodonova	KXRCF
18						FR/CPR				0.058				
19						DGIGA			3	-	2	1		
20						DGIGA-AFC							FCT	
21	14.10c					DGSEM			5		-	∞	Krivodonova	AP-TVD
22						FR/CPR				0.058				
23						DGIGA			3	-	2	1		
24						DGIGA-AFC							FCT	
25	14.11a					DGSEM		60	19			∞	Krivodonova	-
26						FR/CPR				0.152				
27						DGIGA			14	-	2	9		
28						DGIGA-AFC							FCT	
29	14.11b					DGSEM			19		-	∞	Krivodonova	KXRCF

(continues in the next page)

Table 14.5: (continued)

	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
30						FR/CPR				0.152				
31						DGIGA			14	-	2	9		
32						DGIGA-AFC							FCT	
33	14.11c					DGSEM			19		-	∞	Krivodonova	AP-TVD
34						FR/CPR				0.152				
35						DGIGA			14	-	2	9		
36						DGIGA-AFC							FCT	

14.6. DGIGA-AFC revisited

In light of the observation made in §§14.3 and 14.5, consider this: could there be an optimum number of patches for which DGIGA-AFC with FCT limiting becomes competitive with the hierarchical limiter? I aim to test this hypothesis for the Shu-Osher test case, as I consider it the most representative of a flow of engineering interest, via test matrix 14.6. None of these runs apply either of the inter-cell fail-safe limiters; as a consequence, run 6 (which turns out to incur in invalid control values) fails.

Figure 14.12 reveals that for a fixed degree, smoothness and total number of degrees of freedom, the more DGIGA-AFC patches, the more diffusion (regardless of any sensor being used or not). Moreover, comparison between it and figure 14.9 shows that even IGA-AFC remains inferior to DGIGA coupled with krivodonova's limiter. This was the last opportunity for AFC/FCT to prove itself competitive when applied in a DG context; in the form I have proposed to use it at least (§8.7.2), AFC is *not* a good alternative to Krivodonova's limiter for B-spline based DG.

Table 14.6: Isolating the effect of the number of patches in FCT-limited DGIGA-AFC.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.12a	(9.16)	4(10)	10^{-3}	1.8	DGIGA-AFC	1200	1	2	-	1198	1	FCT	-
2								10			118			
3								100			10			
4								200			4			
5								400			1	∞		
6	14.12b				(crashes)			10			118	1		KXRCF
7					1.8			100			10			
8								200			4			
9								400			1	∞		
10	14.12c							10			118	1		AP-TVD
11								100			10			
12								200			4			
13								400			1	∞		

14.7. Do limiters preserve high order?

In §14.5, I touched upon the fact that higher order may not lead to higher accuracy when limiters are involved. With this final batch of experiments, I aim to verify experimentally whether said order is preserved by Krivodonova's and AFC/FCT limiters. I shall do so first in the linear case and for a smooth solution (which is the situation most often assumed when addressing this issue in the literature), and then in the Shu-Osher problem—this time for a fixed number of patches, i.e. the most favorable condition. Runs that use the TVD limiter, known to *not* preserve it, are also included for comparison purposes.

Data from table 14.14 confirms that Krivodonova's limiter preserves accuracy at smooth extrema, and is consistent with [69]. This happens both for DGSEM and DGIGA bases, which give practically indistinguishable results in this category. The FCT limiter, however, clearly does not. With IGA-AFC, the discretization seems to maintain second order accuracy regardless of the B-spline degree; this is similar to the TVB limiter (when used in TVDM fashion), both according to these results and the literature [26]. With DGIGA-AFC, tested here in the most diffusive case (single breakpoint span per patch), this effect is even more pronounced: accuracy degrades to, at best, first order. This explains

why most DGIGA-AFC results in this chapter were much less accurate than their respective inter-cell limited counterparts.

The question remains of whether a high-order preserving limiter facilitates that a DG discretization will approximate a solution combining underresolved and discontinuous features more accurately the higher its number of basis functions per element/patch. Results shown in figure 14.13 indicate that this is actually not the case: Krivodonova's limiter fails to facilitate an accuracy improvement in the Shu-Osher test problem when refining in p only, even if the number of elements/patches is kept constant (the most generous situation possible—compare it with that of batch 14.12, in which the total number of degrees of freedom was maintained instead). As a matter of fact, they suggest that the TVB limiter can be about as accurate if coupled with an effective sensor. And yet, DGIGA-AFC with FCT limiting is still very much worse than either—while IGA-AFC should perform better, recall that higher that second degree discretizations are generally unfeasible due to conditioning issues (see §14.4).

All in all, these results are consistent with the idea that high order preservation at smooth extrema is *necessary* but *not sufficient* to maintain high order in practice. I will go even further; as I see it, the missing ingredient is h-refinement: via hp-adaptation (to solution gradients), it should in principle be possible to reduce the discretization all the way to first order locally (this ensures a monotone capture of the shock) at the same time that accuracy is maintained by concentrating more elements around the discontinuity. This view is already taking shape in the literature, an example being the limiter by Dumbser and Loubère [30]. Unfortunately, the much increased complexity of such an approach has forced me to leave it out of the scope of the present work.

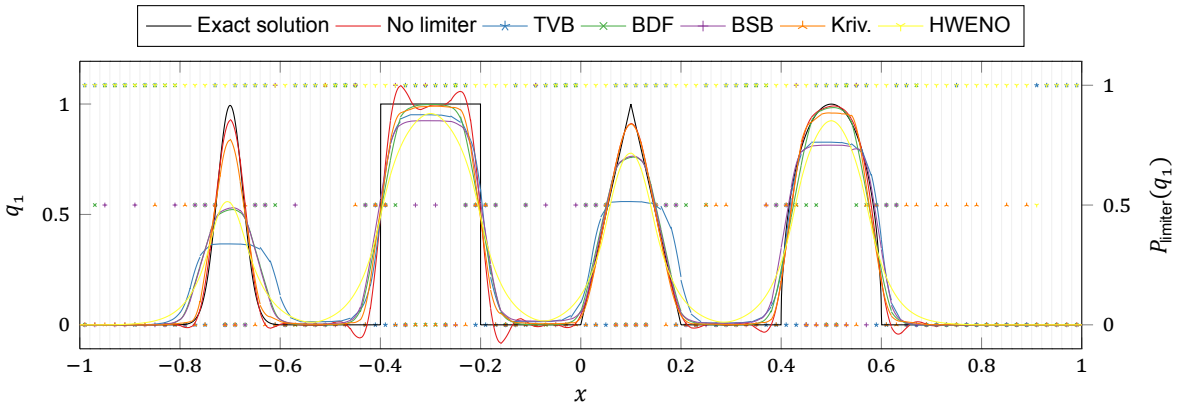
Table 14.7: Checking whether the three main limiters considered preserve accuracy.

			Time discr.			Space discretization							Limiting	
	Fig.	Prob.	RK	Δt	ΔT	Method	$\mathrm{N_{dofs}}$	K	p	η	k	$\varkappa$	Limiter	Sensor
1	14.14a (table)	(9.7)	4(10)	$\Delta t_{\max}/2$	8	DGIGA	48	24	1	-	1	∞	TVB, $M = 0$	-
2							72	36						
3							96	48						
4	14.14b (table)						48	16	2					
5							72	24						
6							96	32						
7	14.14c (table)						48	12	3					
8							72	18						
9							96	24						
10	14.14d (table)						48	24	1				Krivodonova	
11							72	36						
12							96	48						
13	14.14e (table)						48	16	2					
14							72	24						
15							96	32						
16	14.14f (table)						48	12	3					
17							72	18						
18							96	24						
19	14.14g (table)					DGIGA-AFC	48	24	1				FCT	
20							72	36						
21							96	48						
22	14.14h (table)						48	16	2					
23							72	24						
24							96	32						
25	14.14i (table)						48	12	3					
26							72	18						
27							96	24						
28	14.14j (table)					DGSEM	48	24	1				Krivodonova	
29							72	36						
30							96	48						
31	14.14k (table)						48	16	2					
32							72	24						
33							96	32						
34	14.14l (table)						48	12	3					
35							72	18						

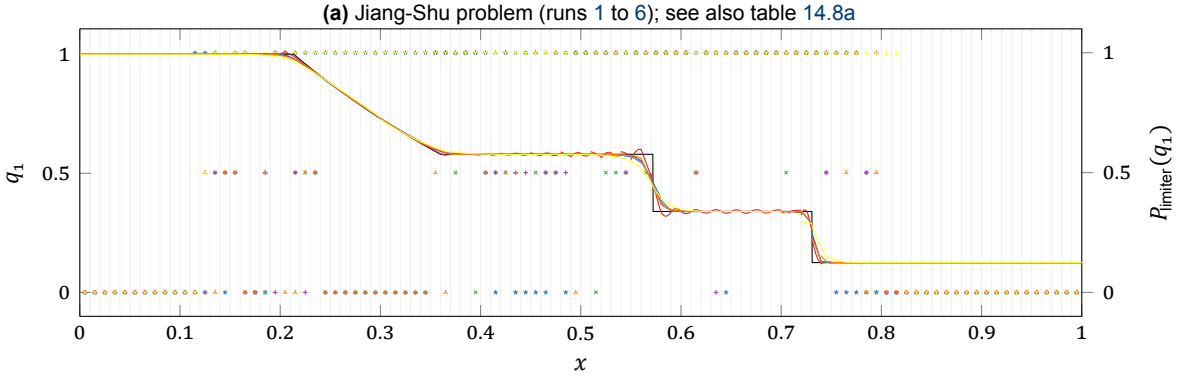
(continues in the next page)

Table 14.7: (continued)

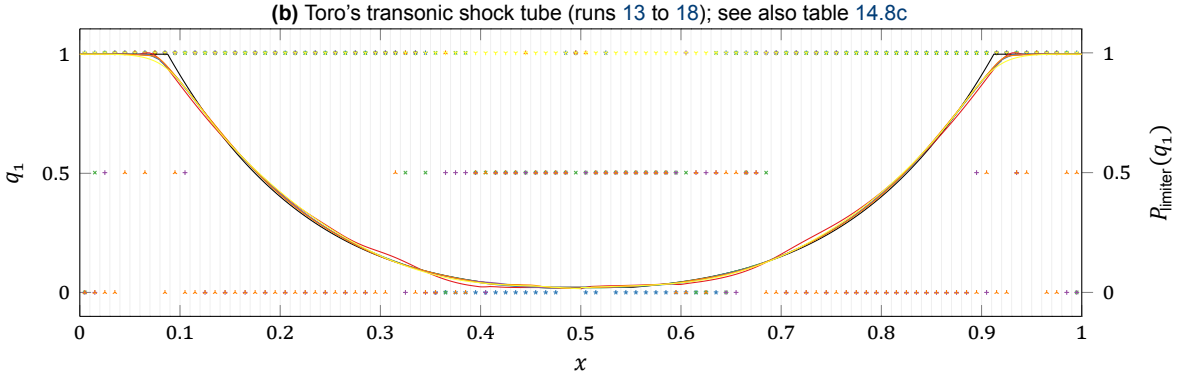
	Fig.	Prob.	RK	Δt	ΔT	Method	N_{dofs}	K	p	η	k	$\varkappa$	Limiter	Sensor
36							96	24						
37	14.14m (table)					DGIGA-AFC	48	1	1		47	0	FCT	
38							72				71			
39							96				95			
40	14.14n (table)						48		2		46	1		
41							72				70			
42							96				94			
43	14.14o (table)						48		3		45	2		
44							72				69			
45							96				93			
46	14.13a	(9.16)			1.8	DGIGA	600	300	1		1	∞	TVB, $M = 0$	KXRCF
47							900		2					
48							1200		3					
49	14.13b						600		1				Krivodonova	
50							900		2					
51							1200		3					
52	14.13c					DGIGA-AFC	600		1				FCT	
53							900		2					
54							1200		3					



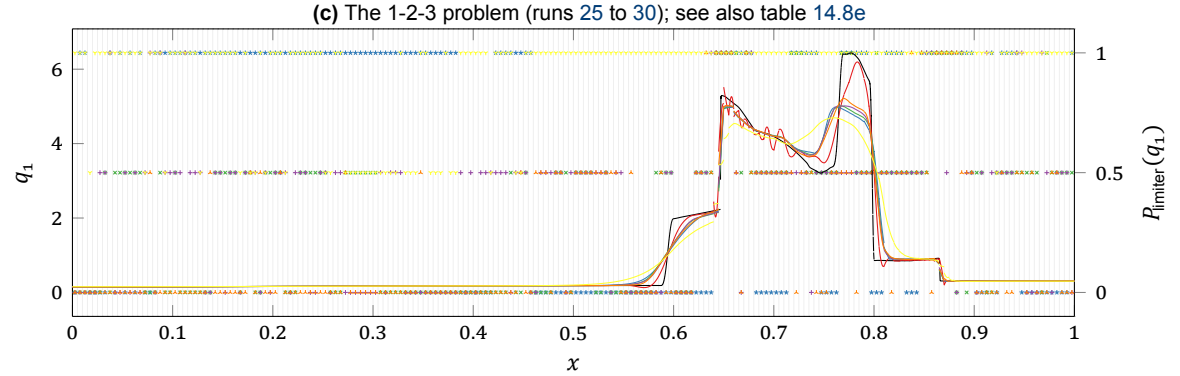
(a) Jiang-Shu problem (runs 1 to 6); see also table 14.8a



(b) Toro's transonic shock tube (runs 13 to 18); see also table 14.8c

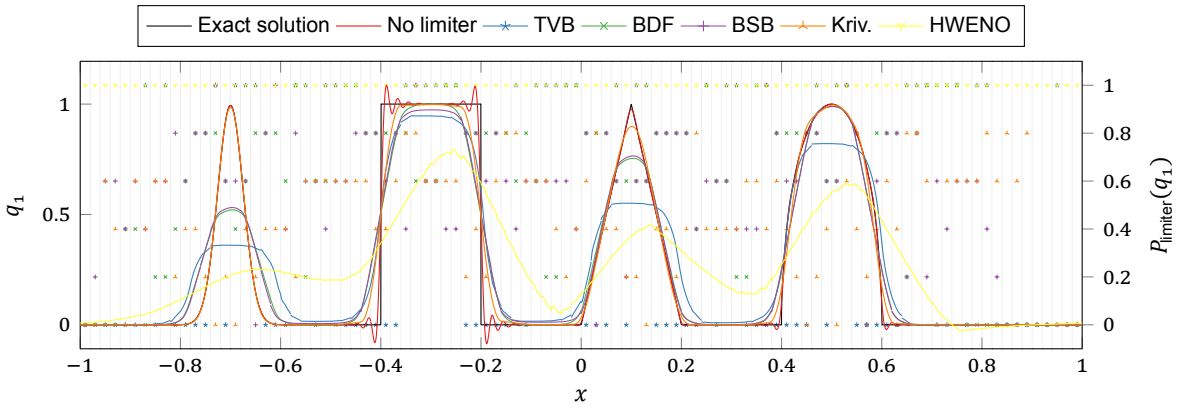


(c) The 1-2-3 problem (runs 25 to 30); see also table 14.8e

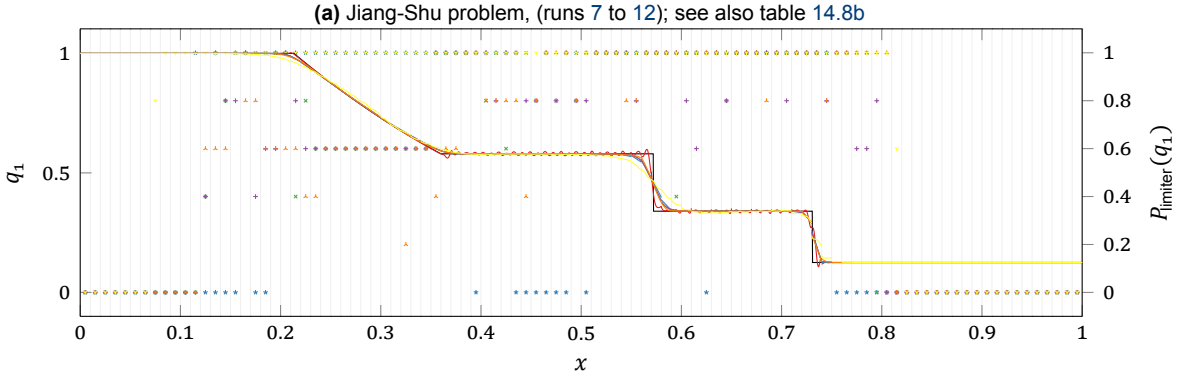


(d) Blast wave interaction (runs 37 to 42); see also table 14.8g

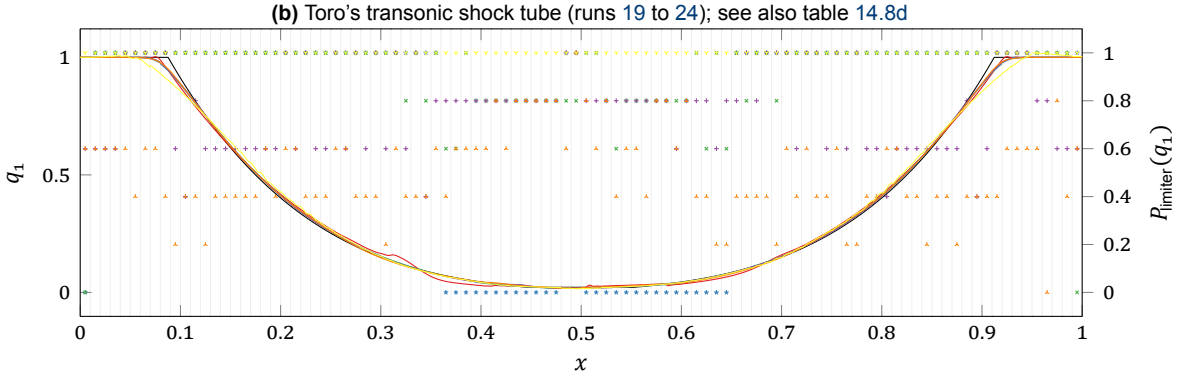
Figure 14.2: Solution and instantaneous limiter activation for all $\mathrm{p} = 2$ runs in table 14.1.



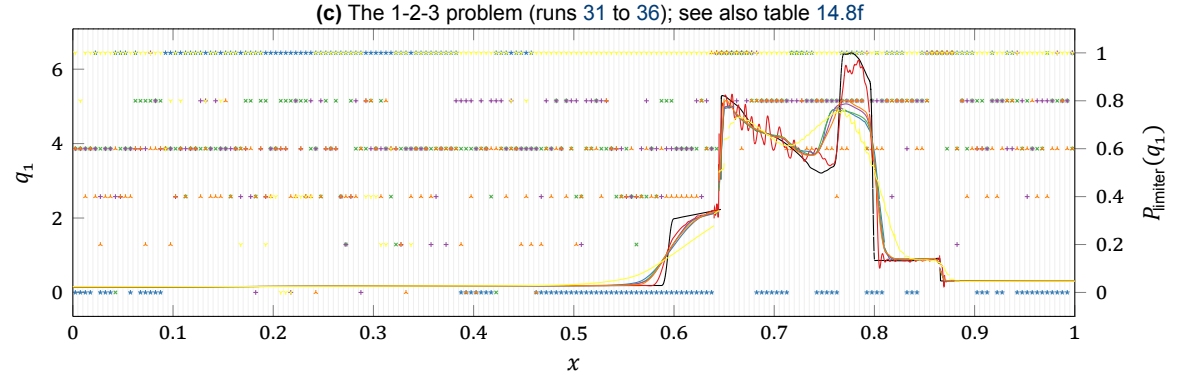
(a) Jiang-Shu problem, (runs 7 to 12); see also table 14.8b



(b) Toro's transonic shock tube (runs 19 to 24); see also table 14.8d

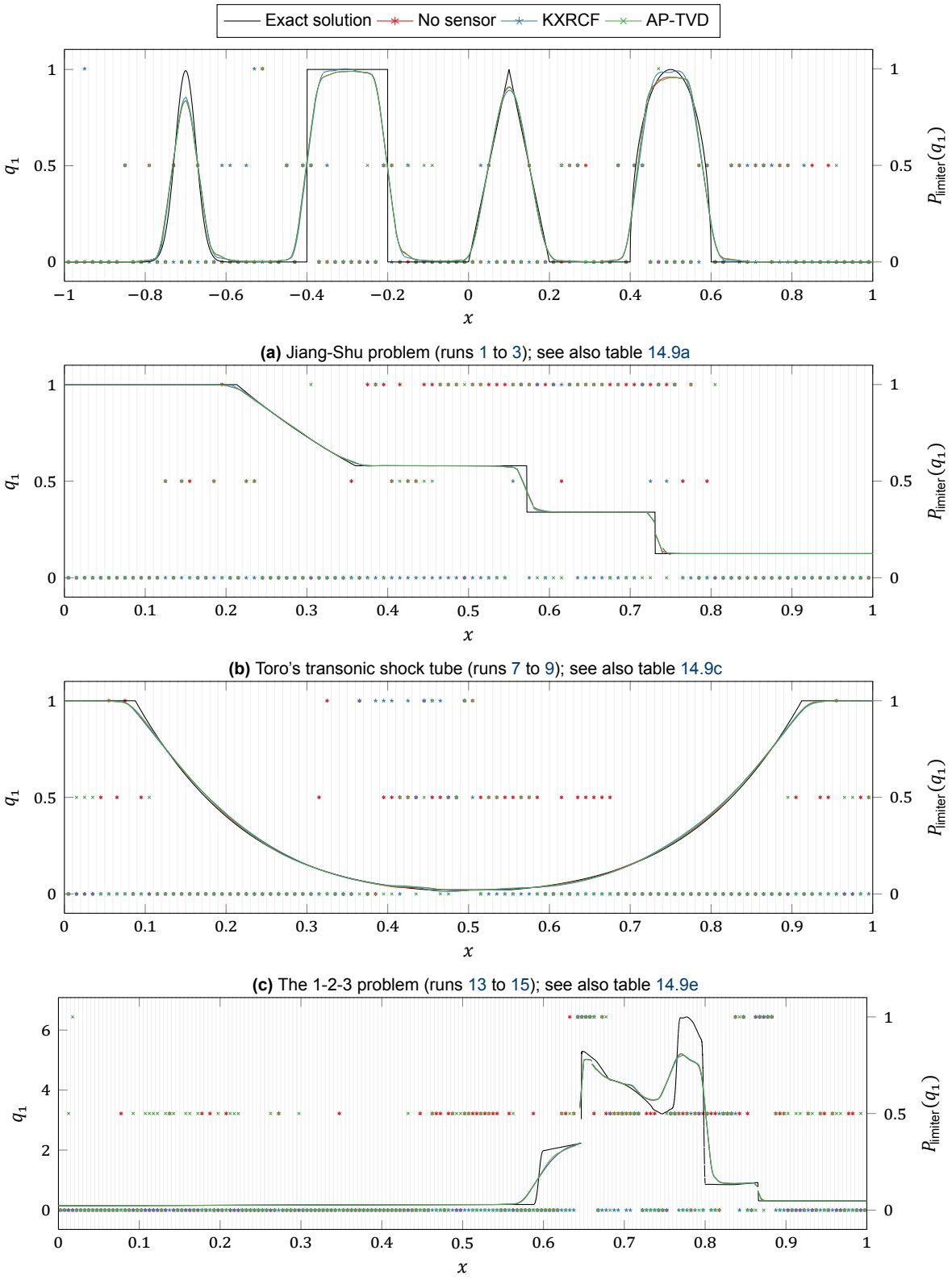


(c) The 1-2-3 problem (runs 31 to 36); see also table 14.8f



(d) Blast wave interaction (runs 43 to 48); see also table 14.8h

Figure 14.3: Idem, for all $\mathrm{p} = 5$ runs in table 14.1.



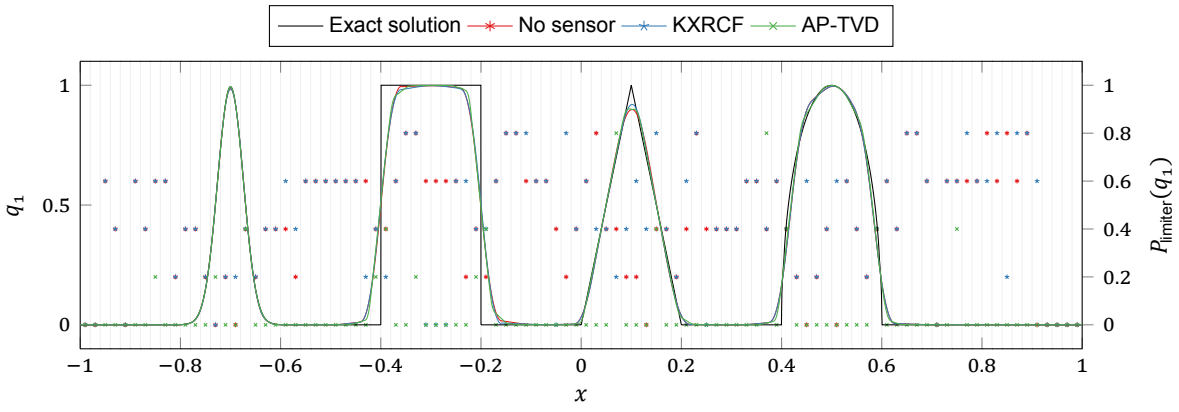
(a) Jiang-Shu problem (runs 1 to 3); see also table 14.9a

(b) Toro's transonic shock tube (runs 7 to 9); see also table 14.9c

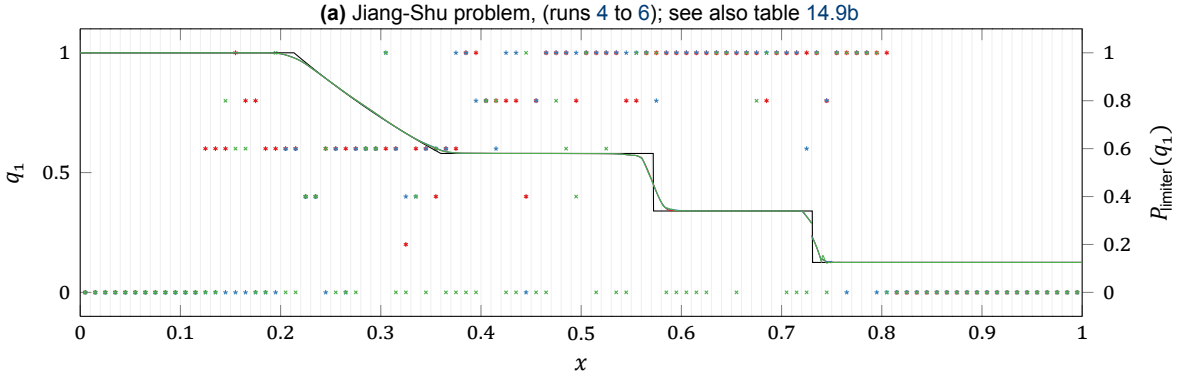
(c) The 1-2-3 problem (runs 13 to 15); see also table 14.9e

(d) Blast wave interaction (runs 19 to 21); see also table 14.9g

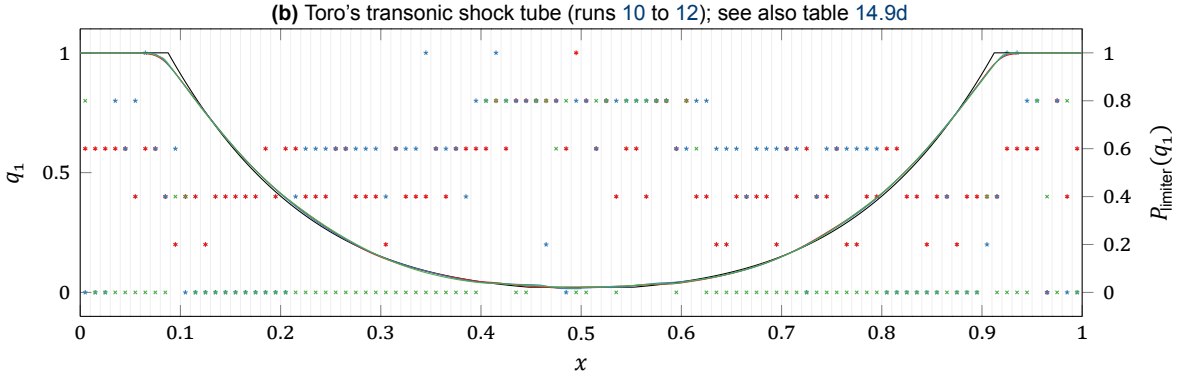
Figure 14.4: Solution and instantaneous limiter activation for all $\mathrm{p} = 2$ runs in table 14.2.



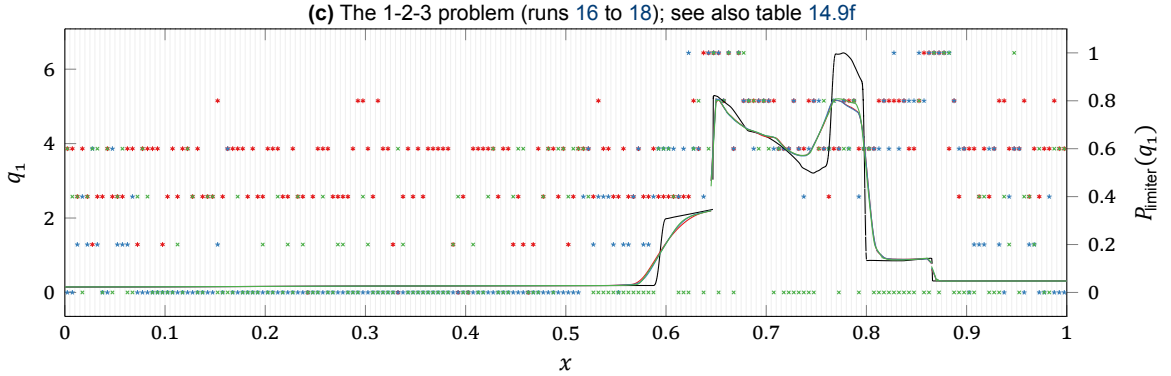
(a) Jiang-Shu problem, (runs 4 to 6); see also table 14.9b



(b) Toro's transonic shock tube (runs 10 to 12); see also table 14.9d

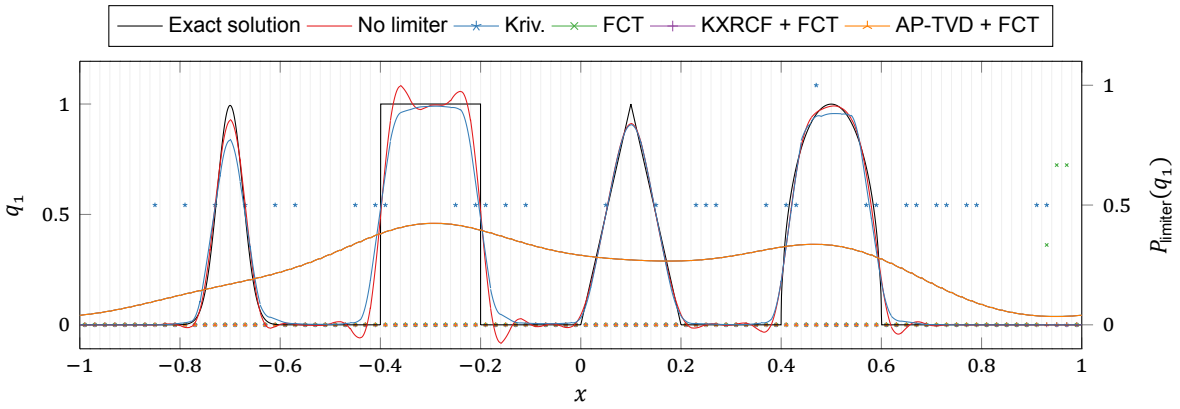


(c) The 1-2-3 problem (runs 16 to 18); see also table 14.9f

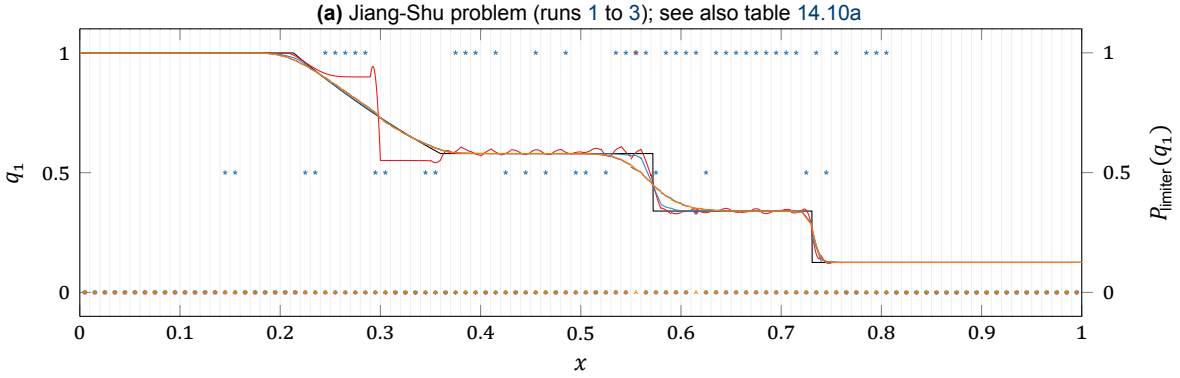


(d) Blast wave interaction (runs 22 to 24); see also table 14.9h

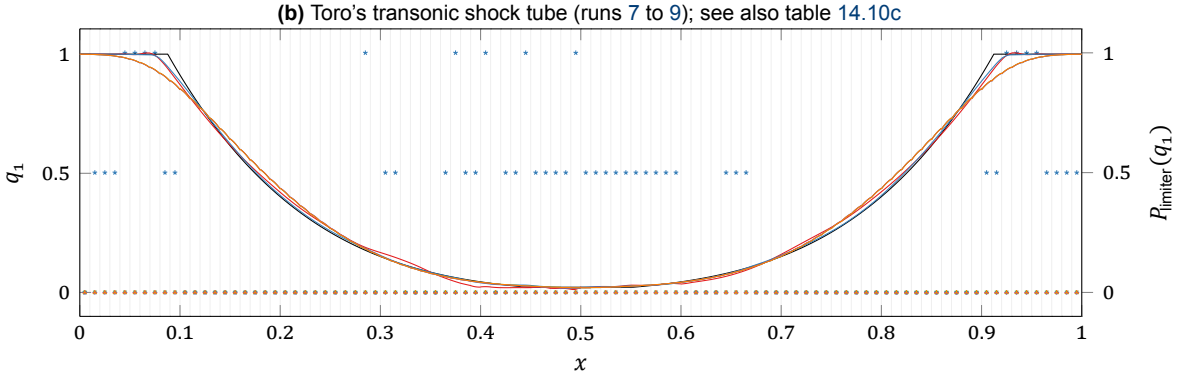
Figure 14.5: Idem, for all $\mathrm{p} = 5$ runs in table 14.2.



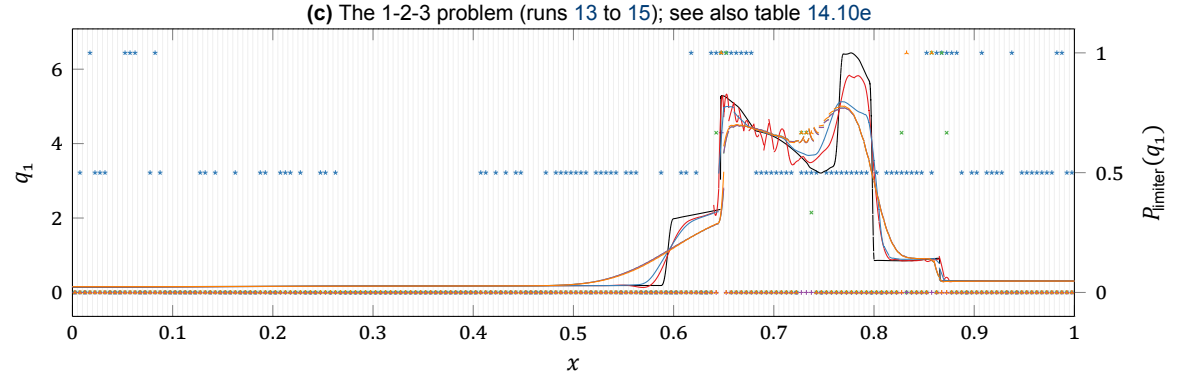
(a) Jiang-Shu problem (runs 1 to 3); see also table 14.10a



(b) Toro's transonic shock tube (runs 7 to 9); see also table 14.10c

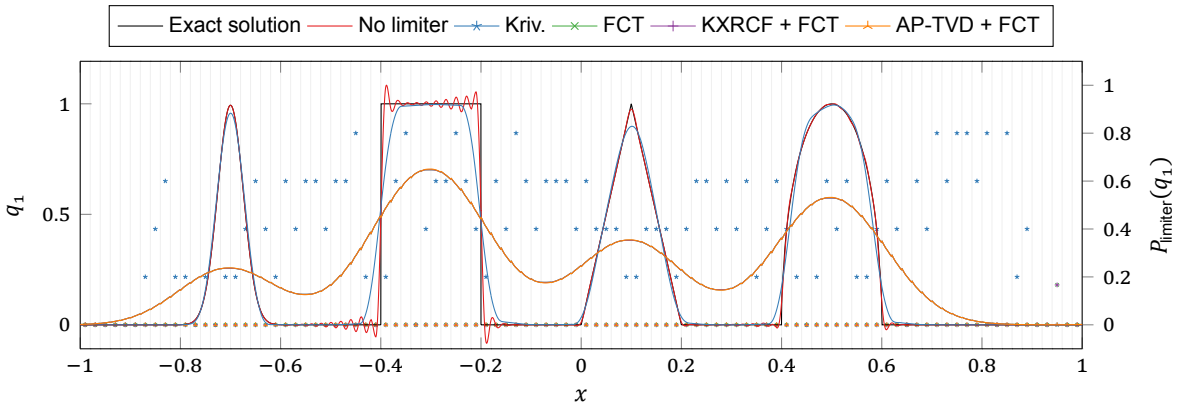


(c) The 1-2-3 problem (runs 13 to 15); see also table 14.10e

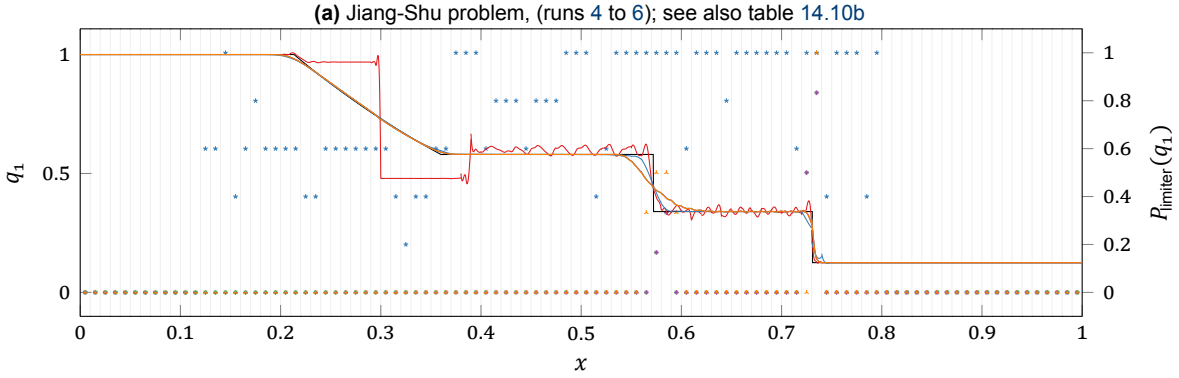


(d) Blast wave interaction (runs 19 to 21); see also table 14.10g

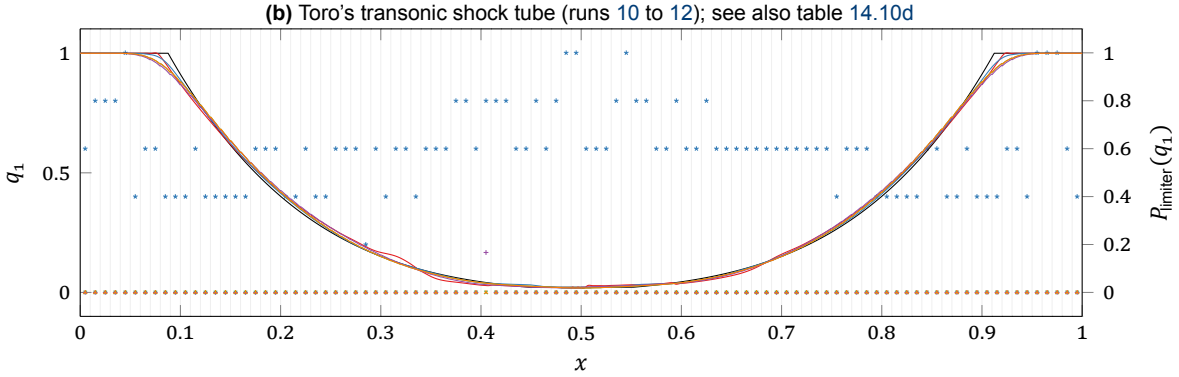
Figure 14.6: Solution and instantaneous limiter activation for all $\mathrm{J} = 3$ runs in table 14.3.



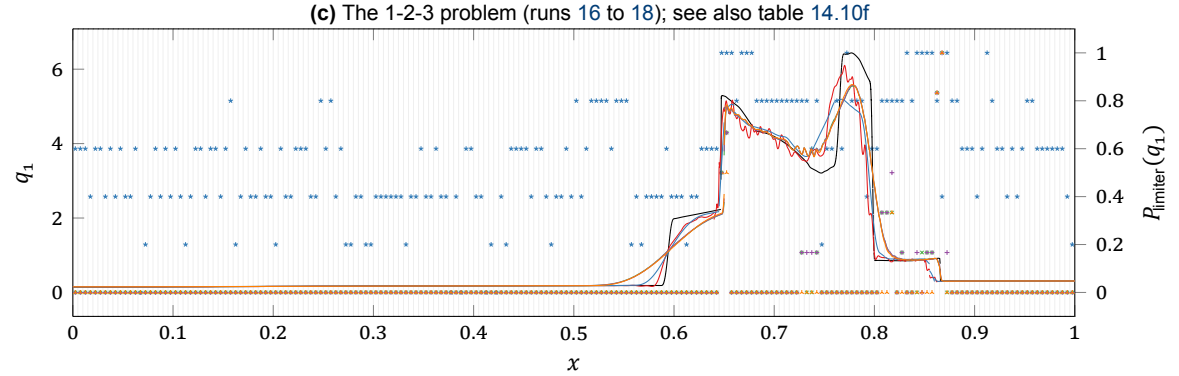
(a) Jiang-Shu problem, (runs 4 to 6); see also table 14.10b



(b) Toro's transonic shock tube (runs 10 to 12); see also table 14.10d

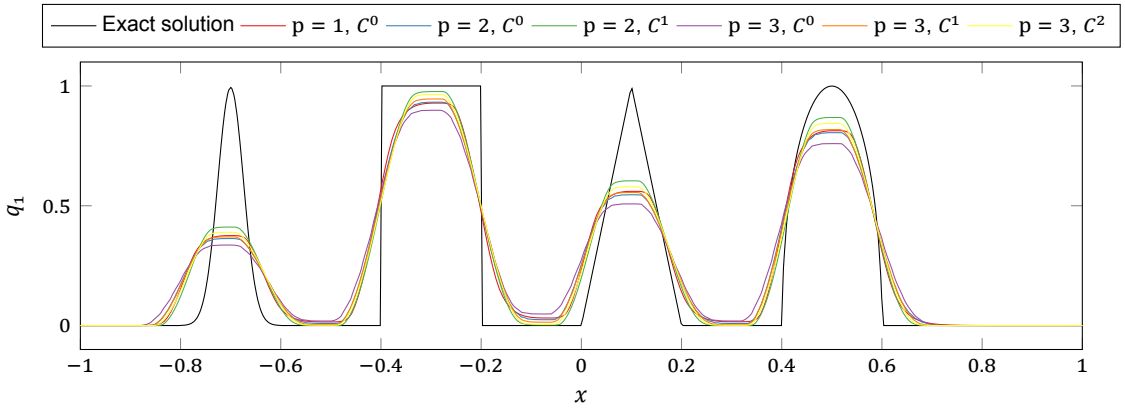


(c) The 1-2-3 problem (runs 16 to 18); see also table 14.10f

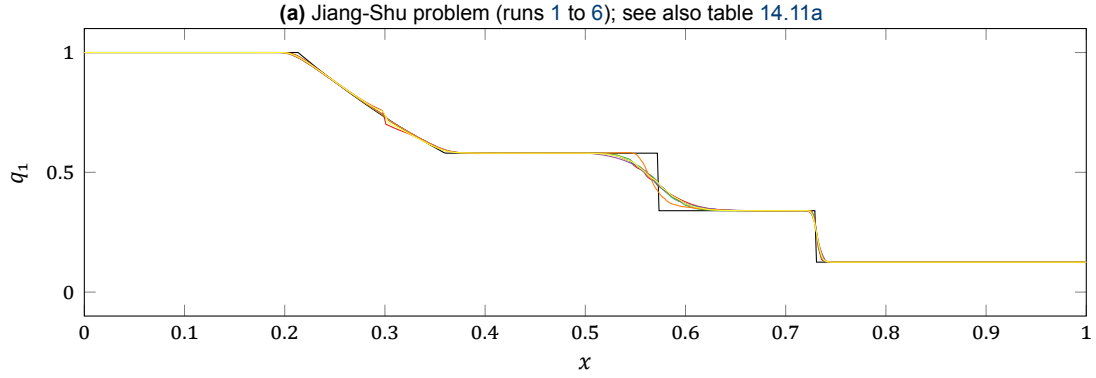


(d) Blast wave interaction (runs 22 to 24); see also table 14.10h

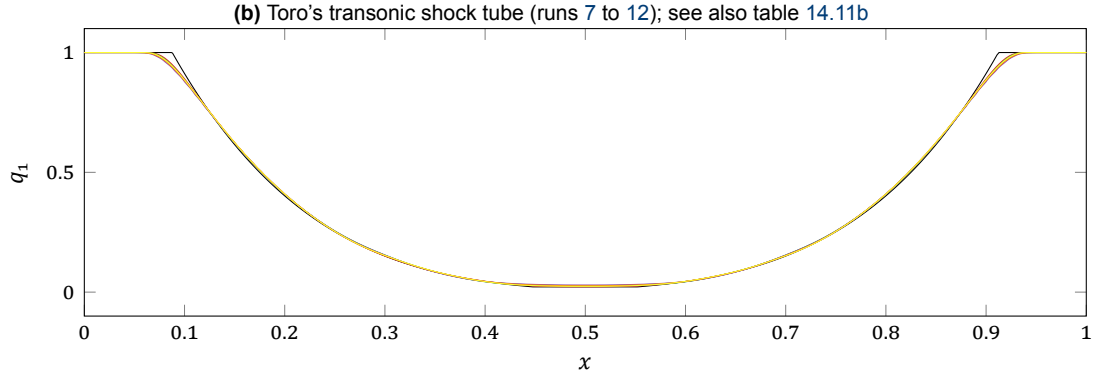
Figure 14.7: Idem, for all $\mathrm{J} = 6$ runs in table 14.3.



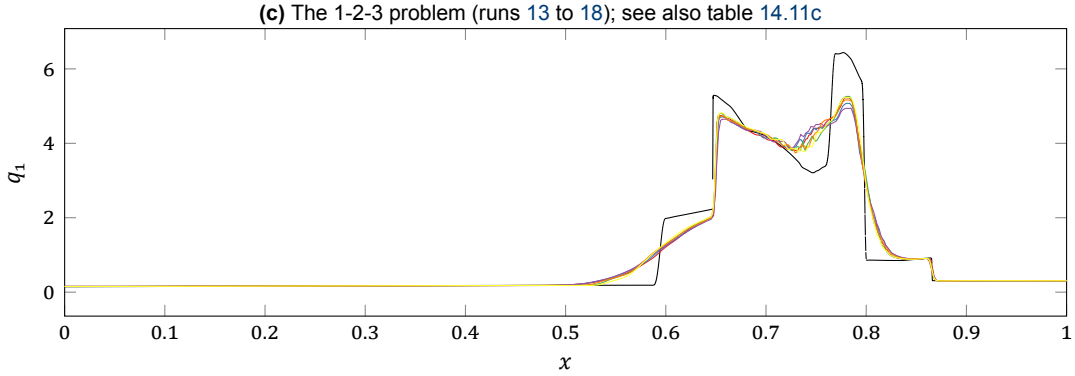
(a) Jiang-Shu problem (runs 1 to 6); see also table 14.11a



(b) Toro's transonic shock tube (runs 7 to 12); see also table 14.11b

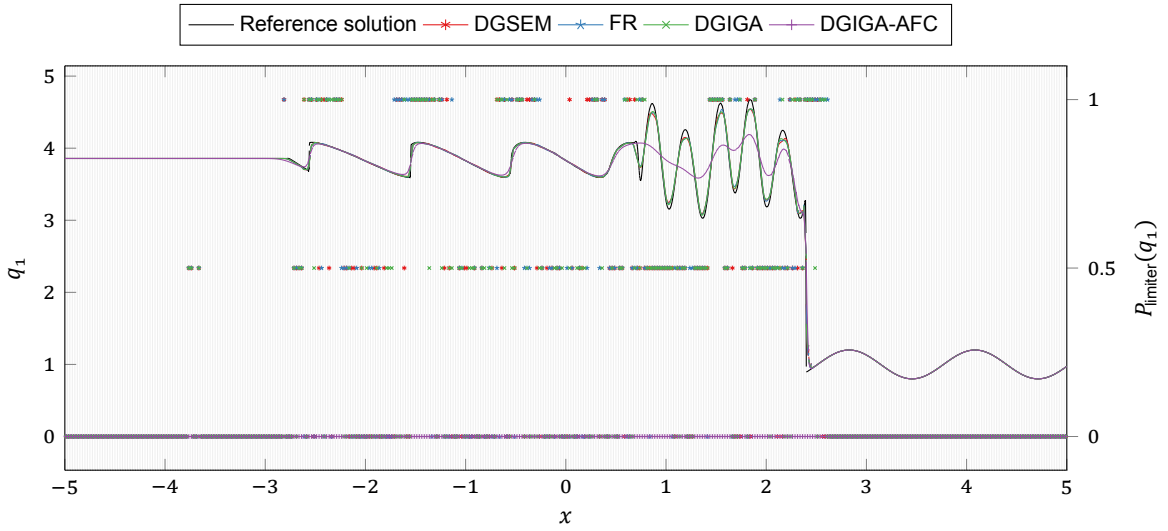


(c) The 1-2-3 problem (runs 13 to 18); see also table 14.11c

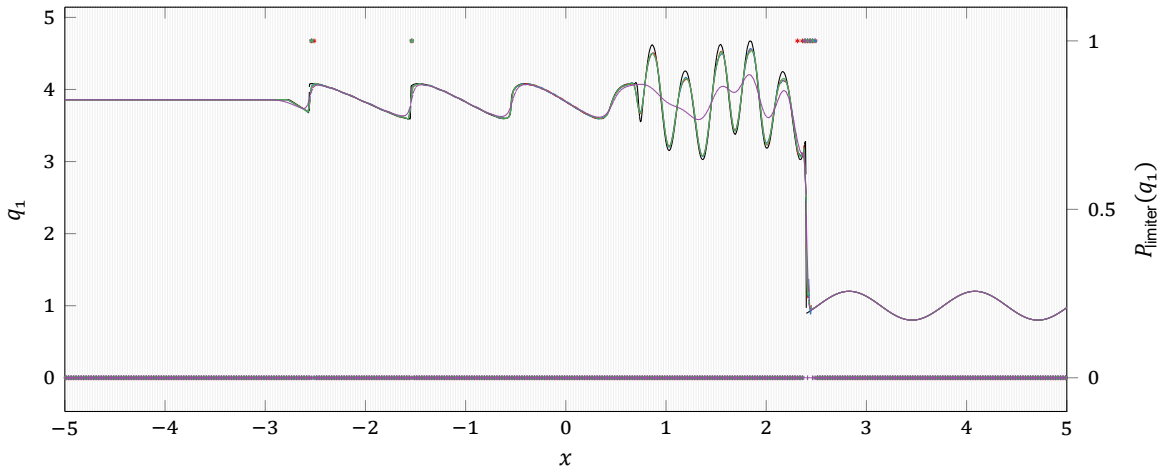


(d) Blast wave interaction (runs 19 to 24); see also table 14.11d

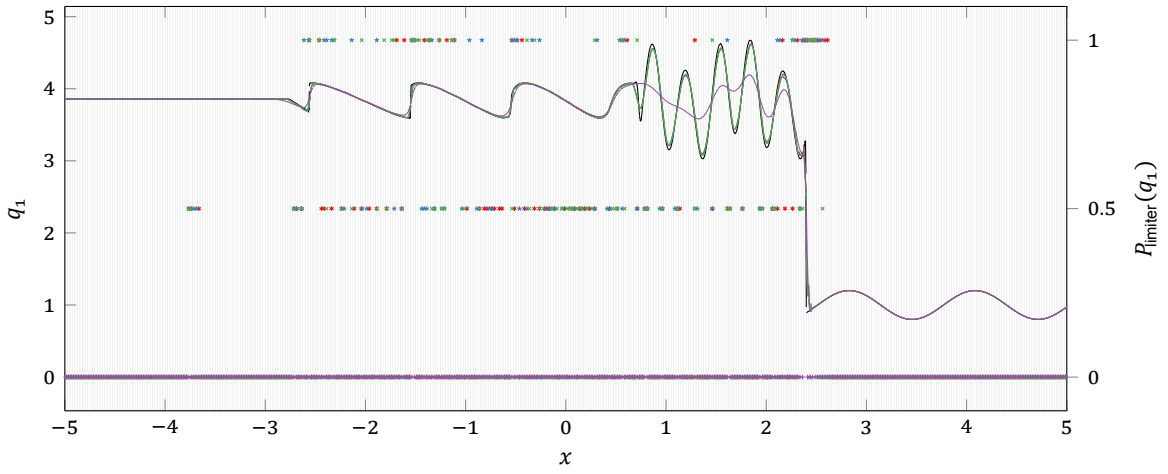
Figure 14.8: Solution of all the runs in table 14.4; these correspond to IGA-AFC, i.e. single-patch DGIGA limited via AFC. The total (mesh-wide) number of degrees of freedom is the same as in figures 14.2, 14.4 and 14.6.



(a) No sensor (runs 1 to 4); see also table 14.12a

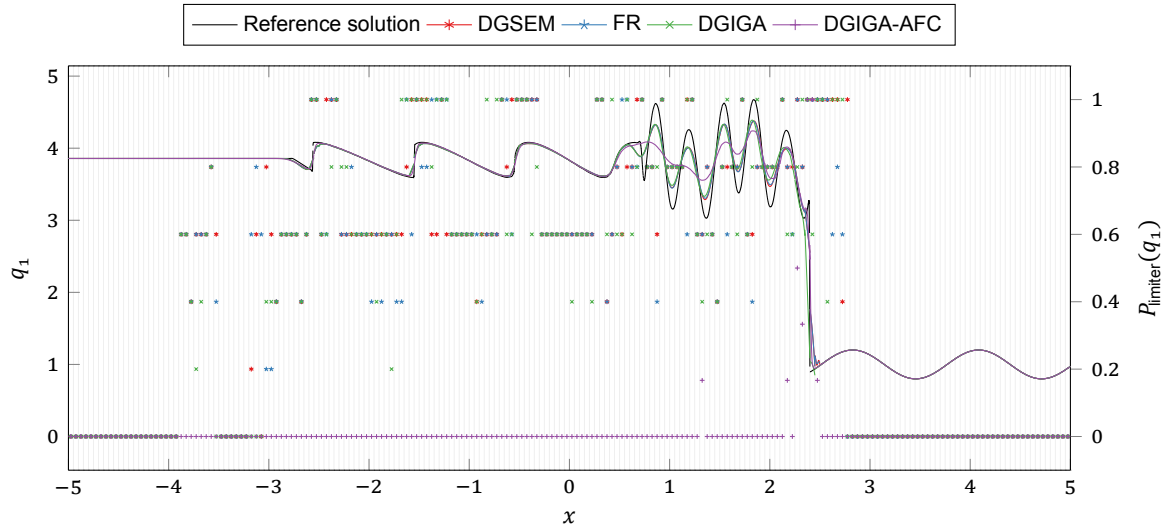


(b) KXRCF sensor (runs 5 to 8); see also table 14.12b

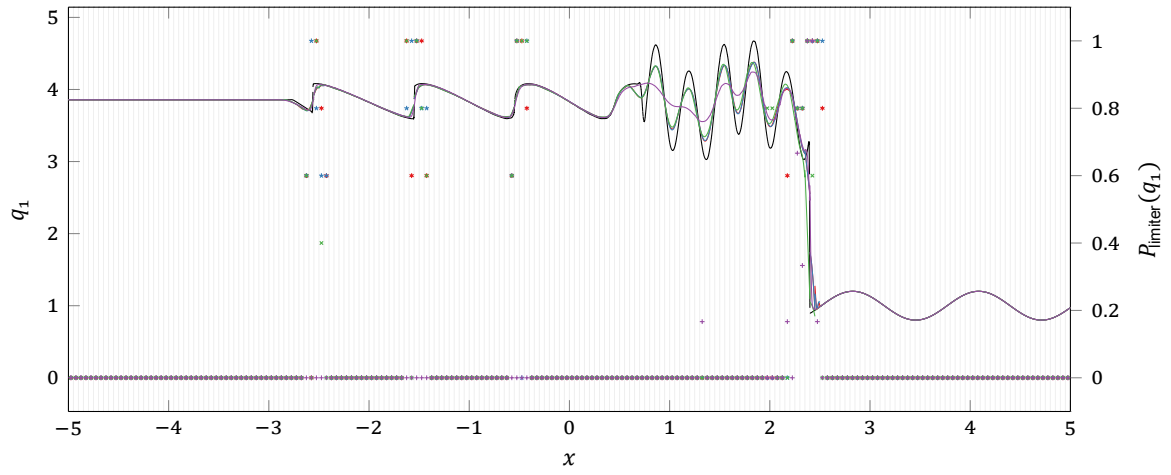


(c) AP-TVD sensor (runs 9 to 12); see also table 14.12c

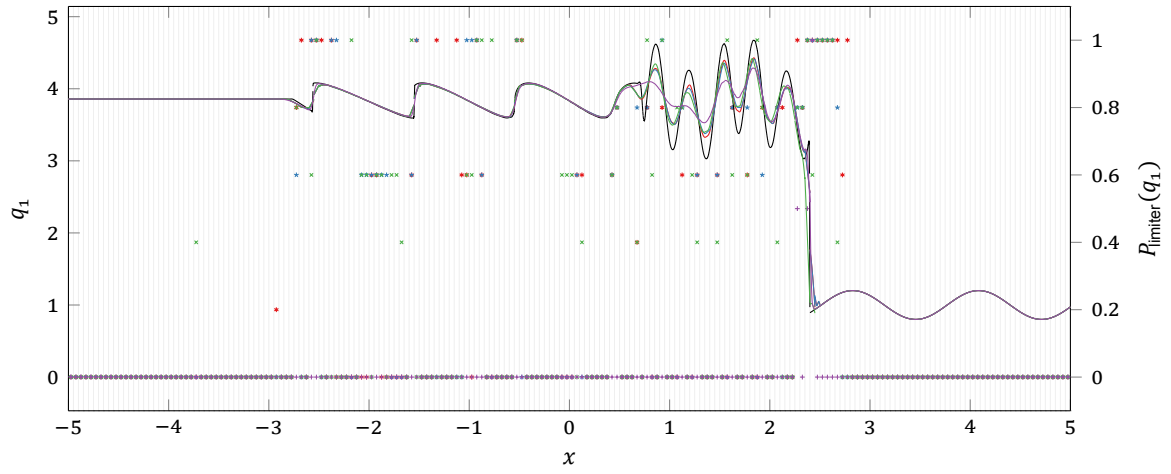
Figure 14.9: Solution and instantaneous limiter activation for all $\mathrm{J} = 3$ runs in table 14.5.



(a) No sensor (runs 13 to 16); see also table 14.12d

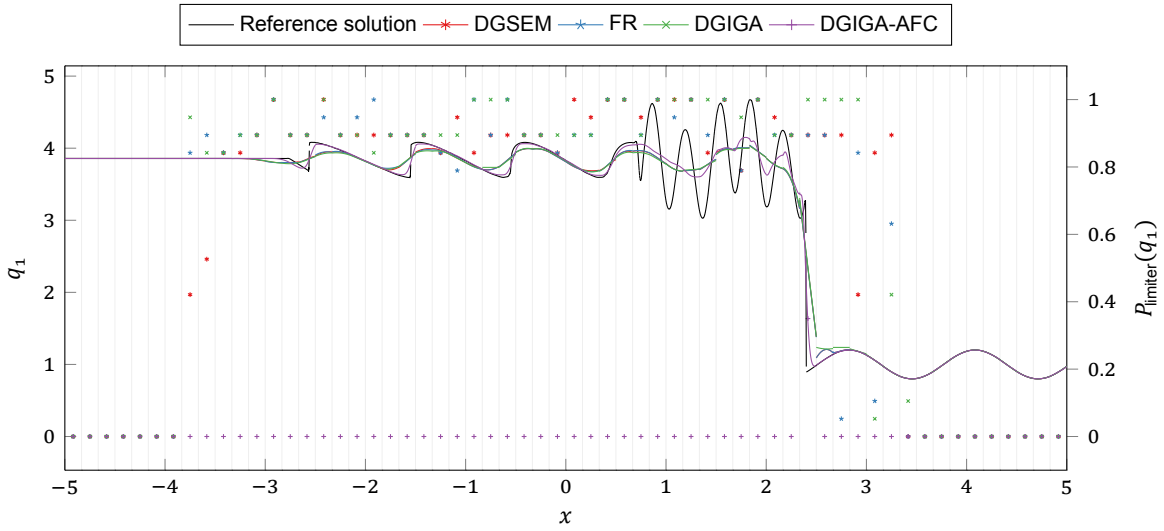


(b) KXRCF sensor (runs 17 to 20); see also table 14.12e

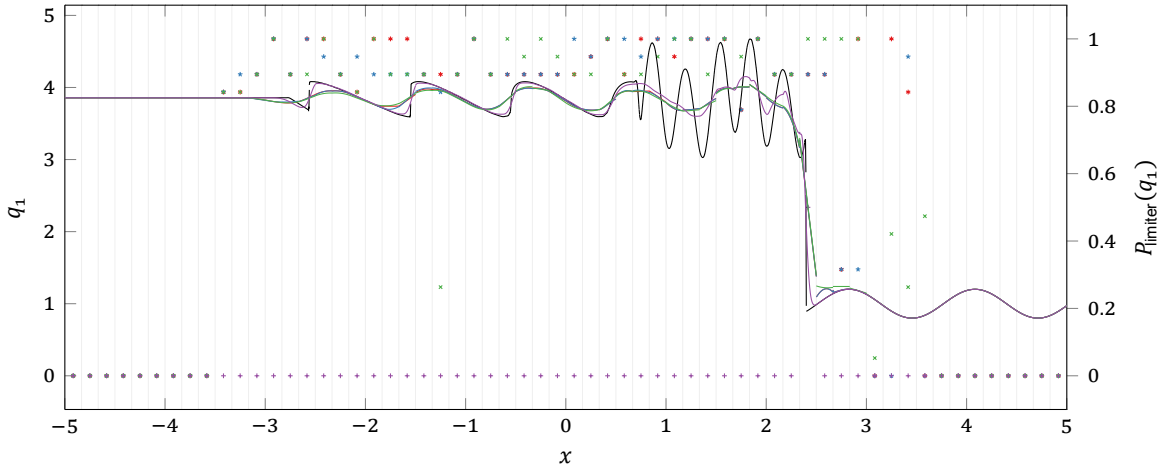


(c) AP-TVD sensor (runs 21 to 24); see also table 14.12f

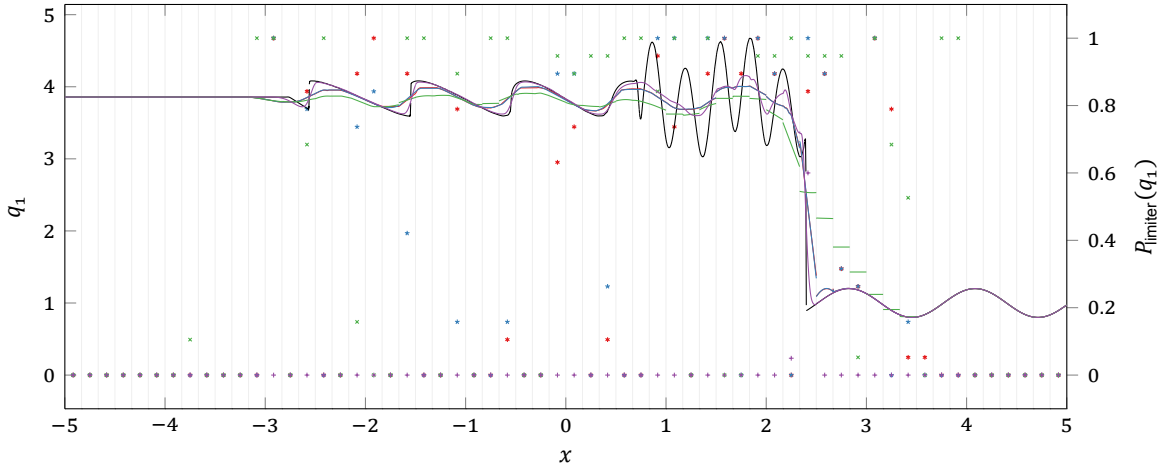
Figure 14.10: Idem, for all $\mathrm{J} = 6$ runs in table 14.5.



(a) No sensor (runs 25 to 28); see also table 14.12g

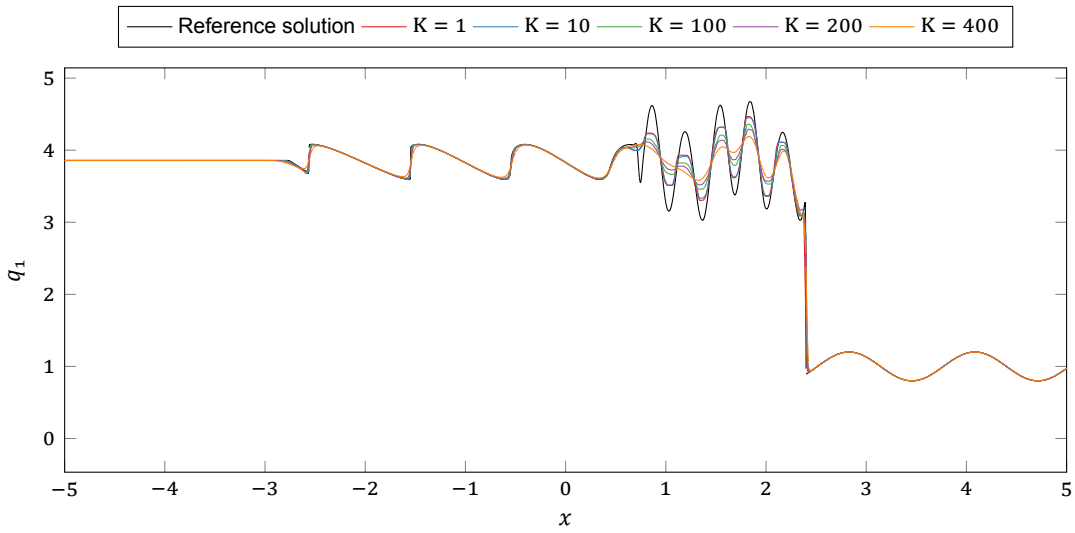


(b) KXRCF sensor (runs 29 to 32); see also table 14.12h

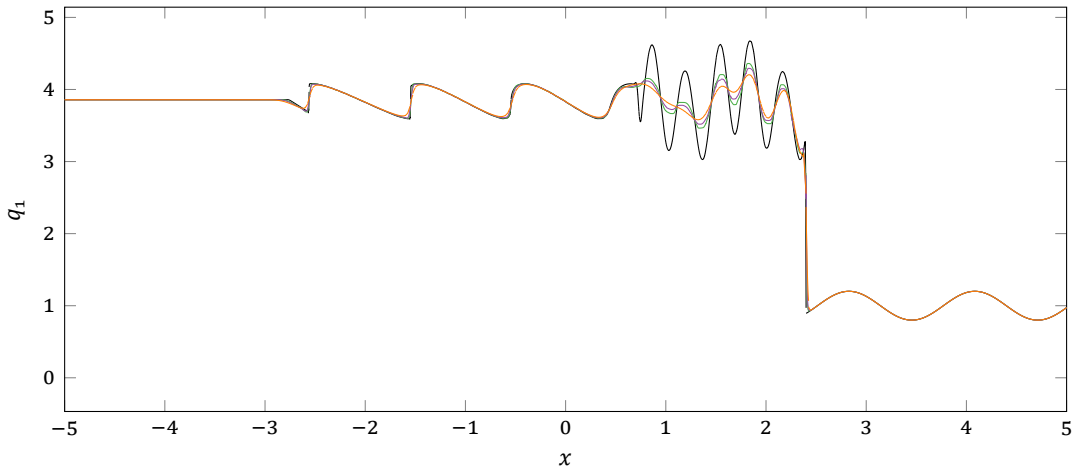


(c) AP-TVD sensor (runs 33 to 36); see also table 14.12i

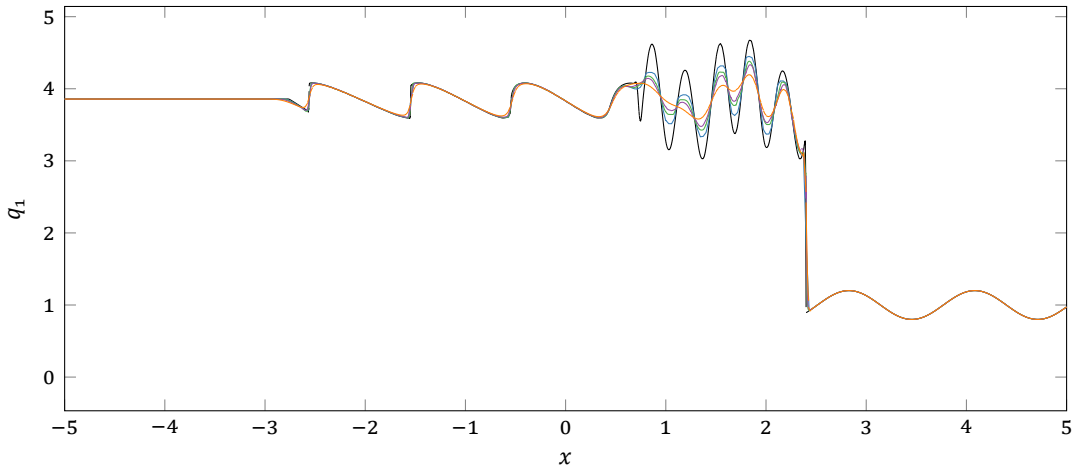
Figure 14.11: Idem, for all $\mathrm{J} = 20$ runs in table 14.5.



(a) No sensor (runs 1 to 5); see also table 14.13a

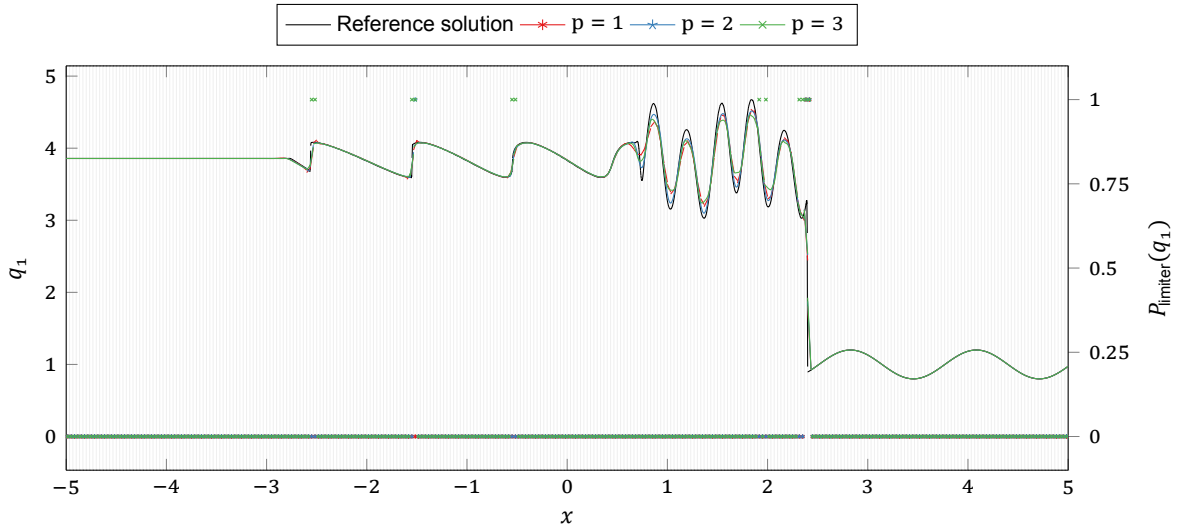


(b) KXRCF sensor (runs 6 to 9); see also table 14.13b

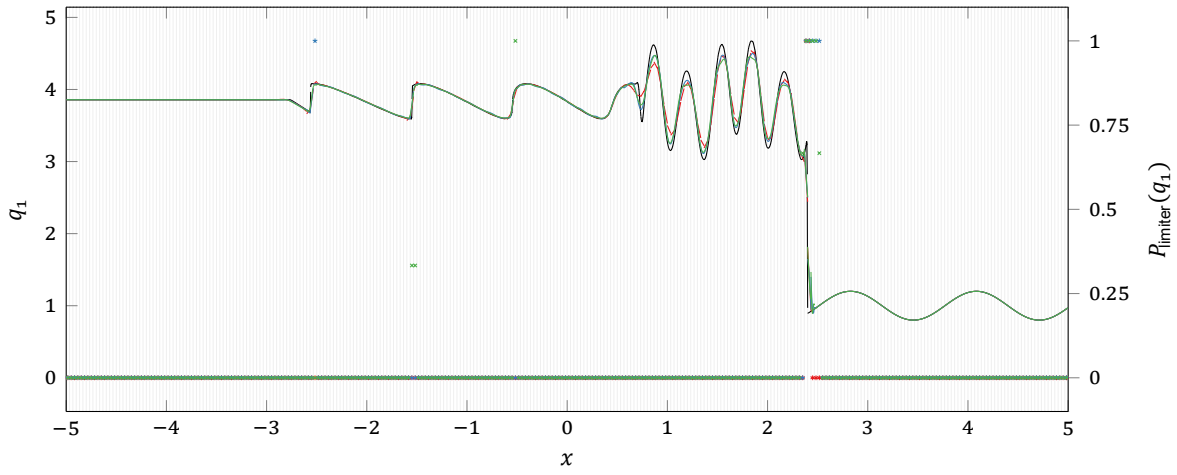


(c) AP-TVD sensor (runs 10 to 13); see also table 14.13c

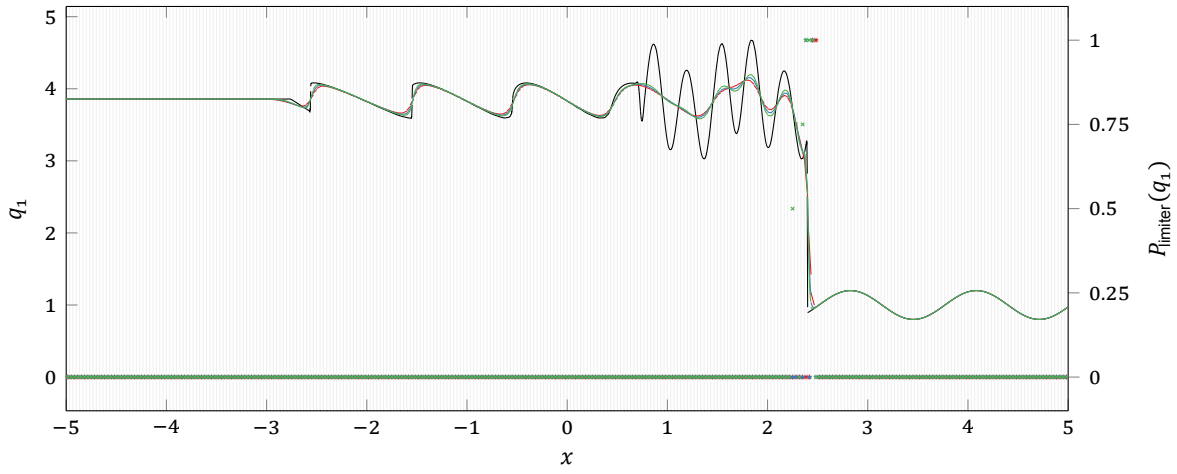
Figure 14.12: Solution for all runs in table 14.6.



(a) TVB limiter + KXRCF sensor (runs 46 to 48); see also table 14.15a



(b) Krivodonova's limiter + KXRCF sensor (runs 49 to 51); see also table 14.15b



(c) AFC/FCT limiter + KXRCF sensor (runs 52 to 54); see also table 14.15c

Figure 14.13: Solution and instantaneous limiter activation for all runs in table 14.5.

Table 14.8: Additional quantitative comparison among the various limiters (test matrix 14.1).

(a) Jiang-Shu problem, $\mathrm{p} = 2$ (figure 14.2a)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$2.6 \cdot 10^{-2}$	1.118	0.0	108
TVB	$8.5 \cdot 10^{-2}$	0.707	39.8	207
BDF	$5.6 \cdot 10^{-2}$	0.830	25.4	211
BSB	$6.6 \cdot 10^{-2}$	0.768	19.2	214
Kriv.	$3.4 \cdot 10^{-2}$	0.941	10.5	199
HWENO	$9.0 \cdot 10^{-2}$	0.801	64.2	200

(b) Jiang-Shu problem, $\mathrm{p} = 5$ (figure 14.3a)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$6.7 \cdot 10^{-3}$	1.188	0.0	289
TVB	$8.8 \cdot 10^{-2}$	0.676	35.0	570
BDF	$5.6 \cdot 10^{-2}$	0.829	44.9	625
BSB	$5.7 \cdot 10^{-2}$	0.824	46.2	639
Kriv.	$2.2 \cdot 10^{-2}$	0.973	36.3	637
HWENO	$2.1 \cdot 10^{-1}$	0.491	80.8	641

(c) Toro's transonic shock tube, $\mathrm{p} = 2$ (figure 14.2b)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$3.6 \cdot 10^{-3}$	1.866	0.0	48
TVB	$4.0 \cdot 10^{-3}$	1.042	26.0	65
BDF	$3.7 \cdot 10^{-3}$	1.023	25.5	63
BSB	$3.7 \cdot 10^{-3}$	1.027	21.2	61
Kriv.	$3.6 \cdot 10^{-3}$	1.109	20.0	62
HWENO	$7.2 \cdot 10^{-3}$	1.055	34.0	61

(d) Toro's transonic shock tube, $\mathrm{p} = 5$ (figure 14.3b)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$2.9 \cdot 10^{-3}$	2.654	0.0	133
TVB	$4.0 \cdot 10^{-3}$	1.041	30.9	180
BDF	$3.7 \cdot 10^{-3}$	1.025	33.4	176
BSB	$3.8 \cdot 10^{-3}$	1.024	32.2	178
Kriv.	$3.4 \cdot 10^{-3}$	1.060	30.7	176
HWENO	$8.5 \cdot 10^{-3}$	1.169	40.4	174

(e) The 1-2-3 problem, $\mathrm{p} = 2$ (figure 14.2c)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$1.3 \cdot 10^{-2}$	1.035	0.0	33
TVB	$6.2 \cdot 10^{-3}$	1.019	29.2	48
BDF	$5.7 \cdot 10^{-3}$	1.006	34.3	45
BSB	$5.9 \cdot 10^{-3}$	1.006	25.0	44
Kriv.	$5.8 \cdot 10^{-3}$	1.008	18.5	44
HWENO	$9.3 \cdot 10^{-3}$	1.007	43.3	46

(f) The 1-2-3 problem, $\mathrm{p} = 5$ (figure 14.3c)

Limiter	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$1.1 \cdot 10^{-2}$	1.039	0.0	93
TVB	$6.2 \cdot 10^{-3}$	1.019	34.8	130
BDF	$5.4 \cdot 10^{-3}$	1.007	45.5	140
BSB	$5.5 \cdot 10^{-3}$	1.007	41.3	139
Kriv.	$6.2 \cdot 10^{-3}$	1.005	30.6	140
HWENO	$1.4 \cdot 10^{-2}$	1.099	57.0	141

(g) Blast wave interaction, $\mathrm{p} = 2$ (figure 14.2d)

Limiter	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	23.06	0.0	545
TVB	12.41	34.9	678
BDF	12.50	24.4	683
BSB	12.58	24.7	660
Kriv.	13.19	19.6	675
HWENO	10.26	48.8	650

(h) Blast wave interaction, $\mathrm{p} = 5$ (figure 14.3d)

Limiter	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	33.39	0.0	1,153
TVB	12.31	42.5	1,767
BDF	12.21	45.9	1,687
BSB	12.66	44.1	1,869
Kriv.	13.31	38.8	1,590
HWENO	12.48	60.5	1,428

Table 14.9: Additional quantitative comparison among the various DGSEM sensors (test matrix 14.2).

(a) Jiang-Shu problem, $\mathrm{p} = 2$ (figure 14.4a)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$3.4 \cdot 10^{-2}$	0.941	10.5	176
KXRCF	$3.3 \cdot 10^{-2}$	0.954	7.3	198
AP-TVD	$3.4 \cdot 10^{-2}$	0.939	10.6	210

(b) Jiang-Shu problem, $\mathrm{p} = 5$ (figure 14.5a)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$2.2 \cdot 10^{-2}$	0.973	36.3	505
KXRCF	$2.2 \cdot 10^{-2}$	0.979	36.2	643
AP-TVD	$2.0 \cdot 10^{-2}$	0.977	4.9	619

(c) Toro's transonic shock tube, $\mathrm{p} = 2$ (figure 14.4b)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$3.6 \cdot 10^{-3}$	1.109	20.0	51
KXRCF	$3.5 \cdot 10^{-3}$	1.065	6.5	45
AP-TVD	$3.6 \cdot 10^{-3}$	1.057	13.9	59

(d) Toro's transonic shock tube, $\mathrm{p} = 5$ (figure 14.5b)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$3.4 \cdot 10^{-3}$	1.060	30.7	147
KXRCF	$3.5 \cdot 10^{-3}$	1.055	23.7	132
AP-TVD	$3.5 \cdot 10^{-3}$	1.100	17.7	160

(e) The 1-2-3 problem, $\mathrm{p} = 2$ (figure 14.4c)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$5.8 \cdot 10^{-3}$	1.008	18.5	38
KXRCF	$7.8 \cdot 10^{-3}$	1.011	7.0	35
AP-TVD	$5.9 \cdot 10^{-3}$	1.007	7.4	41

(f) The 1-2-3 problem, $\mathrm{p} = 5$ (figure 14.5c)

Sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	$6.2 \cdot 10^{-3}$	1.005	30.6	110
KXRCF	$6.1 \cdot 10^{-3}$	1.009	29.7	112
AP-TVD	$6.7 \cdot 10^{-3}$	1.004	11.0	117

(g) Blast wave interaction, $\mathrm{p} = 2$ (figure 14.4d)

Sensor	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	13.19	19.6	562
KXRCF	12.95	6.2	495
AP-TVD	13.13	13.1	608

(h) Blast wave interaction, $\mathrm{p} = 5$ (figure 14.5d)

Sensor	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (s)
–	13.31	38.8	1,447
KXRCF	13.30	34.3	1,479
AP-TVD	13.25	19.0	1,673

Table 14.10: Quantitative comparison between limiters for DGIGA, with and without sensors (test matrix 14.3).

(a) Jiang-Shu problem, $\mathrm{J} = 3$ (figure 14.6a)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$2.6 \cdot 10^{-2}$	1.118	0.0	109
Kriv.	$3.4 \cdot 10^{-2}$	0.939	10.6	912
FCT	$2.9 \cdot 10^{-1}$	0.160	9.7	383
KXRCF	$2.9 \cdot 10^{-1}$	0.161	6.8	398
AP-TVD	$2.9 \cdot 10^{-1}$	0.161	0.3	452

(b) Jiang-Shu problem, $\mathrm{J} = 6$ (figure 14.7a)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$7.1 \cdot 10^{-3}$	1.247	0.0	237
Kriv.	$2.4 \cdot 10^{-2}$	0.965	35.5	3,287
FCT	$2.1 \cdot 10^{-1}$	0.461	8.0	1,858
KXRCF	$2.1 \cdot 10^{-1}$	0.461	8.0	1,819
AP-TVD	$2.1 \cdot 10^{-1}$	0.467	1.4	1,921

(c) Toro's transonic shock tube, $\mathrm{J} = 3$ (figure 14.6b)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$1.6 \cdot 10^{-2}$	2.123	0.0	48
Kriv.	$3.8 \cdot 10^{-3}$	1.048	21.5	155
FCT	$7.5 \cdot 10^{-3}$	1.056	6.3	138
KXRCF	$7.5 \cdot 10^{-3}$	1.057	5.4	122
AP-TVD	$7.4 \cdot 10^{-3}$	1.079	3.4	130

(d) Toro's transonic shock tube, $\mathrm{J} = 6$ (figure 14.7b)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$2.9 \cdot 10^{-2}$	3.641	0.0	96
Kriv.	$3.3 \cdot 10^{-3}$	1.076	30.0	513
FCT	$5.1 \cdot 10^{-3}$	1.138	4.6	627
KXRCF	$5.1 \cdot 10^{-3}$	1.138	4.6	604
AP-TVD	$5.1 \cdot 10^{-3}$	1.148	4.5	635

(e) The 1-2-3 problem, $\mathrm{J} = 3$ (figure 14.6c)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$1.2 \cdot 10^{-2}$	1.046	0.0	36
Kriv.	$5.4 \cdot 10^{-3}$	1.008	20.0	119
FCT	$1.7 \cdot 10^{-2}$	1.120	1.9	98
KXRCF	$1.8 \cdot 10^{-2}$	1.121	2.0	83
AP-TVD	$1.7 \cdot 10^{-2}$	1.120	1.8	89

(f) The 1-2-3 problem, $\mathrm{J} = 6$ (figure 14.7c)

Limiter/ sensor	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	$1.1 \cdot 10^{-2}$	1.035	0.0	79
Kriv.	$6.1 \cdot 10^{-3}$	1.005	34.1	394
FCT	$9.2 \cdot 10^{-3}$	1.146	0.9	348
KXRCF	$1.3 \cdot 10^{-2}$	1.147	1.4	319
AP-TVD	$9.2 \cdot 10^{-3}$	1.146	0.8	329

(g) Blast wave interaction, $\mathrm{J} = 3$ (figure 14.6d)

Limiter/ sensor	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	24.26	0.0	566
Kriv.	12.90	21.9	1,681
FCT	12.07	7.5	1,531
KXRCF	12.07	5.7	1,469
AP-TVD	12.21	4.7	1,514

(h) Blast wave interaction, $\mathrm{J} = 6$ (figure 14.7d)

Limiter/ sensor	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
–	23.02	0.0	1,219
Kriv.	13.06	38.3	4,851
FCT	15.34	6.0	5,972
KXRCF	15.38	5.8	5,681
AP-TVD	15.42	5.3	5,240

Table 14.11: Quantitative results for the IGA-AFC runs of test matrix 14.4.

(a) Jiang-Shu problem (figure 14.8a)

Degree	Smoothness class	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
1	C^0	$1.0 \cdot 10^{-1}$	0.655	60.8	161
2	C^0	$1.1 \cdot 10^{-1}$	0.652	56.2	218
2	C^1	$9.0 \cdot 10^{-2}$	0.716	44.4	524
3	C^0	$1.3 \cdot 10^{-1}$	0.605	53.6	301
3	C^1	$1.1 \cdot 10^{-1}$	0.668	48.5	359
3	C^2	$9.9 \cdot 10^{-2}$	0.694	44.1	595

(b) Toro's transonic shock tube (figure 14.8b)

Degree	Smoothness class	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
1	C^0	$6.9 \cdot 10^{-3}$	1.001	4.8	130
2	C^0	$6.4 \cdot 10^{-3}$	1.032	5.0	194
2	C^1	$5.5 \cdot 10^{-3}$	1.001	6.5	394
3	C^0	$6.9 \cdot 10^{-3}$	1.014	6.6	252
3	C^1	$4.5 \cdot 10^{-3}$	1.010	5.6	273
3	C^2	$6.5 \cdot 10^{-3}$	1.000	7.0	452

(c) The 1-2-3 problem (figure 14.8c)

Degree	Smoothness class	$\left\|q_1^h - q_1\right\|_1$	$\frac{\mathrm{TV}\left(q_1^h\right)}{\mathrm{TV}\left(q_1\right)}$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
1	C^0	$4.1 \cdot 10^{-3}$	0.999	1.4	71
2	C^0	$5.9 \cdot 10^{-3}$	0.994	2.6	103
2	C^1	$5.1 \cdot 10^{-3}$	0.996	2.0	138
3	C^0	$6.8 \cdot 10^{-3}$	0.992	3.2	106
3	C^1	$5.7 \cdot 10^{-3}$	0.995	2.3	122
3	C^2	$5.1 \cdot 10^{-3}$	0.998	1.9	145

(d) Blast wave interaction (figure 14.8d)

Degree	Smoothness class	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (s)
1	C^0	12.09	11.5	644
2	C^0	11.88	13.6	1,031
2	C^1	13.17	11.6	1,425
3	C^0	11.59	16.5	1,376
3	C^1	12.32	13.2	1,014
3	C^2	12.60	12.0	1,130

Table 14.12: Quantitative comparison between basis types, with compatible limiters and sensors (test matrix 14.5).

(a) No sensor, $J = 3$ (figure 14.9a)

Basis	$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
DGSEM	18.99	10.6	0.4
FR/CPR	18.98	10.3	0.4
DGIGA	18.80	9.6	1
DGIGA-AFC	10.19	1.5	0.9

(b) KXRCF, $J = 3$ (figure 14.9b)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
19.74	0.8	0.3
20.02	0.8	0.3
19.63	0.8	0.5
10.31	1.0	0.8

(c) AP-TVD, $J = 3$ (figure 14.9c)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
19.64	6.1	0.4
19.84	5.9	0.4
19.64	5.6	1.4
10.27	0.9	1

(d) No sensor, $J = 6$ (figure 14.10a)

Basis	$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
DGSEM	15.04	22.2	0.3
FR/CPR	14.92	22.1	0.3
DGIGA	14.63	22.2	0.9
DGIGA-AFC	11.22	1.8	1

(e) KXRCF, $J = 6$ (figure 14.10b)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
15.40	5.2	0.2
15.29	5.2	0.2
14.83	5.8	0.3
11.30	1.7	1

(f) AP-TVD, $J = 6$ (figure 14.10c)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
14.97	8.8	0.3
14.63	9.2	0.3
14.38	9.9	0.8
11.76	1.5	0.9

(g) No sensor, $J = 20$ (figure 14.11a)

Basis	$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
DGSEM	7.50	41.0	0.2
FR/CPR	7.51	40.3	0.2
DGIGA	7.06	40.7	6.8
DGIGA-AFC	9.62	1.6	7.3

(h) KXRCF, $J = 20$ (fig. 14.11b)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
7.47	34.4	0.2
7.60	33.7	0.2
7.03	35.0	3.3
9.63	1.5	8.2

(i) AP-TVD, $J = 20$ (fig. 14.11c)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
7.36	21.2	0.2
7.42	20.4	0.2
5.66	27.2	4.8
9.75	1.6	7.6

Table 14.13: Quantitative comparison of third order DGIGA-AFC discretizations, with and without sensors, for varying numbers of patches (test matrix 14.6).

(a) No sensor (figure 14.12a)

K	$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
10	14.61	1.2	1.7
100	13.44	1.4	1.5
200	12.20	1.4	1.5
400	10.10	1.1	1.6

(b) KXRCF (figure 14.12b)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
-	-	-
13.38	1.3	1.4
12.28	1.4	1.4
10.19	0.8	1.1

(c) AP-TVD (figure 14.12c)

$TV\left(q_1^h\right)$	$\overline{P}_{\text{limiter}}$ (%)	Elapsed time (h)
14.62	1.2	1.7
13.89	1.3	1.3
13.03	1.3	1.5
10.18	0.6	1.4

Table 14.14: L^1 error, its measured rate of convergence (with increasing $\mathrm{N_{dofs}}$) and averaged limiter activation; runs 1 to 45 of test matrix 14.7.

(a) DGIGA, TVB limiter; p = 1

$\mathrm{N_{dofs}}$	$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
48	$3.7 \cdot 10^{-2}$		18.8
72	$1.5 \cdot 10^{-2}$	−2.28	12.7
96	$7.4 \cdot 10^{-3}$	−2.34	9.6

(b) Idem, p = 2

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$6.9 \cdot 10^{-2}$		40.3
$3.5 \cdot 10^{-2}$	−1.67	26.1
$1.9 \cdot 10^{-2}$	−2.13	20.1

(c) Idem, p = 3

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$1.2 \cdot 10^{-1}$		50.1
$6.0 \cdot 10^{-2}$	−1.74	35.7
$3.4 \cdot 10^{-2}$	−1.93	26.6

(d) DGIGA, Krivodonova's limiter; p = 1

$\mathrm{N_{dofs}}$	$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
48	$3.7 \cdot 10^{-2}$		18.8
72	$1.5 \cdot 10^{-2}$	−2.28	12.7
96	$7.4 \cdot 10^{-3}$	−2.34	9.6

(e) Idem, p = 2

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$3.5 \cdot 10^{-3}$		6.3
$8.6 \cdot 10^{-4}$	−3.51	4.3
$3.4 \cdot 10^{-4}$	−3.18	3.3

(f) Idem, p = 3

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$1.1 \cdot 10^{-3}$		9.8
$1.9 \cdot 10^{-4}$	−4.4	6.9
$5.7 \cdot 10^{-5}$	−4.18	5.6

(g) DGIGA-AFC, FCT limiter; p = 1

$\mathrm{N_{dofs}}$	$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
48	$1.9 \cdot 10^{-1}$		20.8
72	$1.5 \cdot 10^{-1}$	−0.58	13.1
96	$1.3 \cdot 10^{-1}$	−0.65	9.0

(h) Idem, p = 2

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$2.1 \cdot 10^{-1}$		25.4
$1.6 \cdot 10^{-1}$	−0.56	16.0
$1.4 \cdot 10^{-1}$	−0.63	10.9

(i) Idem, p = 3

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$2.1 \cdot 10^{-1}$		25.6
$1.7 \cdot 10^{-1}$	−0.57	15.1
$1.4 \cdot 10^{-1}$	−0.63	10.0

(j) DGSEM, Krivodonova's limiter; p = 1

$\mathrm{N_{dofs}}$	$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
48	$3.7 \cdot 10^{-2}$		18.8
72	$1.5 \cdot 10^{-2}$	−2.28	12.7
96	$7.4 \cdot 10^{-3}$	−2.34	9.6

(k) Idem, p = 2

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$3.5 \cdot 10^{-3}$		6.3
$8.6 \cdot 10^{-4}$	−3.51	4.3
$3.4 \cdot 10^{-4}$	−3.18	3.3

(l) Idem, p = 3

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$1.1 \cdot 10^{-3}$		9.8
$1.9 \cdot 10^{-4}$	−4.4	6.9
$5.7 \cdot 10^{-5}$	−4.18	5.6

(m) IGA-AFC, FCT limiter; p = 1

$\mathrm{N_{dofs}}$	$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
48	$5.2 \cdot 10^{-2}$		60.1
72	$2.4 \cdot 10^{-2}$	−1.89	36.1
96	$1.3 \cdot 10^{-2}$	−2.07	23.0

(n) Idem, p = 2

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$4.8 \cdot 10^{-2}$		48.2
$2.0 \cdot 10^{-2}$	−2.15	27.7
$9.8 \cdot 10^{-3}$	−2.48	17.6

(o) Idem, p = 3

$\left\|q_1^h - q_1\right\|_1$	Rate	$\overline{P}_{\mathrm{limiter}}$ (%)
$5.5 \cdot 10^{-2}$		48.2
$2.4 \cdot 10^{-2}$	−2.08	29.2
$1.2 \cdot 10^{-2}$	−2.49	17.1

Table 14.15: Quantitative comparison of the numerical solutions of (9.16), obtained with a fixed number of second, third and fourth-order DGIGA/DGIGA-AFC patches (runs 46 to 54 of test matrix 14.7).

(a) TVB ($M = 0$) + KXRCF (fig. 14.13a)

p	$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (min)
1	18.31	0.5	5.3
2	18.66	0.9	11.7
3	16.50	2.3	21.2

(b) Kriv. + KXRCF (fig. 14.13b)

$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (min)
18.31	0.5	6.5
19.63	0.9	13.5
18.41	1.4	17.1

(c) FCT + KXRCF (fig. 14.13c)

$\mathrm{TV}\left(q_1^h\right)$	$\overline{P}_{\mathrm{limiter}}$ (%)	Elapsed time (min)
8.95	0.8	6.8
9.60	1.2	21.1
10.12	1.2	48.9

15

Conclusions

This thesis has been concerned with the potential of high-order methods for application to scale-resolving simulations of high-speed, possibly turbulent, flows. Rather than targeting a turbulent high-speed flow in itself, I have instead studied the most fundamental aspects of three discontinuous finite element methods (DGSEM, FR/CPR and DGIGA) in the relatively simple context of one-dimensional hyperbolic transport equations (linear advection, inviscid Burgers equation and Euler equations). The suitability for LES and DNS of the latter scheme, in relation to the former two, has thus been evaluated *indirectly*.

In what follows, I express my conclusions based on the results discussed in each of the previous chapters. Additionally, I highlight the most important shortcomings of the work and provide recommendations for future research. This chapter is divided into five sections, the first four corresponding to each of the research sub-questions posed in the introduction chapter. These double as an account, chapter by chapter, of all tasks carried out. I then end with a final answer to the main research question of this project that takes into account all previous considerations.

15.1. Order of accuracy

I started by verifying that all three spatial schemes, applied to Burgers equation, were indeed high order. DGSEM, FR/CPR and (most configurations of) DGIGA are able to achieve their formal order of accuracy, at least up to degree 4 and order 5. I was also able to detect an exponential type of convergence when refining in polynomial/B-spline degree. For FR/CPR, I confirmed that the correction parameter $\eta > 0$ can only reduce the global order of accuracy with respect to that of DGSEM. I also checked that all optimal explicit time-schemes of the SSP-RK type converged at their expected rates as Δt was reduced. The fact that all these results are consistent with the literature and/or theory is an indication that, to some extent, the implementation used is error-free.

In relation to the research questions of this project, I found that DGIGA is able to achieve high order in the same sense that DGSEM and FR/CPR do, albeit with the following caveats:

- This is only true for the modal treatment of DGIGA; nodal DGIGA reaches only second order accuracy, regardless of its degree.
- Even though all DGIGA configurations show the expected order of accuracy when refining in Δx, increasing the number of breakpoint spans (i.e. refining “in a CG way”) improves said order even further, in some cases.
- Some configurations of DGIGA simply fail—in the sense that the algorithm executing them is unstable. Neither linear stability limits nor condition number issues are able to explain this phenomenon in all cases that it occurs.

For DGIGA, results suggest that the best way of extracting the most accuracy per degree of freedom is to increase both B-spline degree and smoothness simultaneously. The next best options would be to increase degree only (at a fixed smoothness), then number of breakpoint spans, and, lastly, number of patches.

15.1.1. Limitations

Only orders up to 5 could be confirmed. To measure the order of convergence, I qualitatively compared the slope of a rectilinear (in logarithmic axes) portion of the error vs. degrees of freedom plot with the formal order of accuracy of the scheme in question, based on its degree. However, I found that for orders higher than 5 there were no linear portions in said plot. I am under the impression that this is because the error stagnates (it does not reduce any further) before this linear portion can appear.

Also, no indication of superconvergence (orders of accuracy higher than $\mathrm{p}+1$) was found for DGSEM nor FR/CPR. I attribute this to the way in which I measure the numerical error and its norm.

15.1.2. Recommendations

All things considered, these results were evidence that higher order can lead to higher accuracy for a fixed number of degrees of freedom, even in the nonlinear case of Burgers equation, as long as no limiters and/or discontinuous solution are involved. Yet, it remains to be seen if, and under which conditions, this translates into lower overall cost for a given accuracy; this is a possible direction of future research.

An alternative way to define the numerical error would have been as the difference between the approximate solution and the *projection* of the exact solution to the same finite dimensional space as the former. Because the exact solution is generally not a piecewise polynomial, it would make sense that the definition of the error used in this work can only ever reach orders of convergence up to $\mathrm{p}+1$. In the alternative definition, however, the error would only be dependent on the dispersive and dissipative characteristics of the discretization. The fact that the theoretical order of accuracy estimate (A.65), which predicts better-than-optimal orders, is based on spectral properties suggests that this way to define the error would have led to the detection of superaccuracy. Moreover, this approach would allow the computation of the error norm to be made directly by subtracting nodal or control point–wise values (because both functions involved would then share a common discretization), thus avoiding the need to numerically integrate the error. Nevertheless, one would still need to use quadrature to project the exact solution into the approximate one's trial space.

It might also be interesting to clarify the complicated relation that order of convergence seems to have with the various refinement directions in DGIGA. Any insight on the unexplained nonlinear instability could prove valuable as well.

15.2. Dispersion, dissipation and linear stability

Next, I studied the wave propagation behavior of the three schemes. I did this through an *a priori* modified wavenumber analysis, valid only for a linear discretization. Time-step size bounds for liner stability were also obtained.

In DGSEM, the higher the order, the wider the range of wavenumbers that are well resolved. The catch, however, is that, simultaneously, there is a localized increase in dispersion error in the middle portion of the underresolved wavenumber range. FR/CPR has one extra free parameter, the correction factor η, which I simply sampled across its full range for each degree. Results for DGSEM and FR/CPR once more matched the literature, indicating that the methodology used and its implementation was sound.

Characterizing DGIGA was challenging due to it having three (constrained) parameters for a given number of basis function per patch: degree, number of breakpoint spans and smoothness. I opted to study the effect that each of these parameters had while keeping the other two fixed. While not exhaustive, I consider that this has allowed me to provide a representative picture of DGIGA.

DGIGA presents the same trends as DGSEM when degree and smoothness are increased simultaneously; they even coincide for the Bernstein case (single breakpoint span). Balancing degree, smoothness and/or number of breakpoint spans, I realized that it was possible to influence the dispersion and dissipation curves of a given J more effectively than in FR/CPR. I also discovered a curious "bubbling" phenomenon for DGIGA: as the number of degrees of freedom per patch increases (also influenced by the rest of basis parameters) its Fourier footprint, usually a single contour, splits into three; the two new ones split again, and so on. These "bubble" eigenmodes are reminiscent of those observed when using a non-dissipative numerical flux across elements in DGSEM.

In all three schemes, a higher polynomial/B-spline degree was found to be associated with a more restrictive Courant number. This bound can be relaxed mildly in FR/CPR, by increasing the correction

factor. In DGIGA, since the number of degrees of freedom per patch is no longer dependent on degree only, the potential for a much laxer linear stability constraint exists.

A striking feature of DGIGA is that its dispersion and dissipation relations for equal number of degrees of freedom, yet different degree, are comparable. This has powerful implications for the potential of high-order schemes in LES and DNS: for a fixed number of degrees of freedom, lower-order DGIGA offers dispersion and dissipation characteristics comparable to those of higher-order DGSEM.

15.2.1. Limitations and recommendations

The spectral characteristics reported focused mostly on the so-called primary or physical eigenmode, which is representative only in the lower range of wavenumbers. In general, all eigenmodes contribute to the wave propagation characteristics. Combined-mode analysis would have been a more reliable way to measure the spectral response of high order schemes.

A more exhaustive exploration of the DGIGA configuration space is still possible. I doubt, however, that it would add much additional insight. More general kinds of FR/CPR correction functions could have been considered as well; recall that only the VCJH class was considered in this work.

15.3. Optimal FR/CPR and DGIGA configurations

The realization that DGIGA (also FR/CPR, but only to a minor extent) allowed the freedom to influence dispersion, dissipation and linear stability independently from degree and/or order of accuracy, led me to the inclusion of an additional chapter (and associated research sub-question) in the thesis. My goal was simple: determine, for a given number of degrees of freedom per element/patch, the best (in some sense) possible configuration of DGIGA and FR/CPR.

I had to establish a criterion through which to measure the quality of a discretization. This ended up being the overall spectral error (both dispersive and dissipative contributions) across the entire range of resolved wavenumbers, after an arbitrary simulation time, determined via a combined-mode analysis approach. In addition, alternative criteria (theoretical order of accuracy, resolving efficiency, ratio between dispersion and dissipation, and computational cost estimation) were also considered. These, however, were not used as objective functions; they were only evaluated afterwards.

Optimized DGIGA turned out to possess advantageous spectral characteristics with respect to DGSEM, starting from 6 degrees of freedom per patch. Interestingly, from that point on, the optimal DGIGA patch happens to always be the one with a single breakpoint inside it. For FR/CPR, on the other hand, no optima had barely any advantage over DGSEM in terms of the chosen objective function; a result consistent with the literature, once more giving validity to the approach employed. In addition, the following features of DGIGA optima were revealed:

- The ratio between their numerical dispersion and dissipation appears to be particularly favorable for LES and/or DNS applications (the scheme has an increased inherent tendency to dissipate its own dispersion errors).
- Their resolving efficiency (largest well-resolved wavenumber) is higher than DGSEM's—despite the former having a *lower* order of accuracy.
- They posses significantly (up to 70 %) larger stability limits than DGSEM.

These findings suggest that DGIGA possesses inherently superior spectral characteristics compared to DGSEM, at least based on the indirect metrics considered in this work. Said advantage is clearly greater for DGIGA than for FR/CPR. It appears that that the true strength of high-order methods might not actually be their order in itself, but rather the reduced dissipation and diffusion *over a range of wavenmumbers* that usually—but not necessarily—comes with it. DGIGA, unlike FR/CPR (of the VCJH type) and DGSEM, possesses the necessary freedom to exploit this. At the same time, nevertheless, the higher resolving efficiency and stability limits of DGIGA seem not to be enough to compensate for its extra cost in terms of floating point operations compared to DGSEM.

15.3.1. Limitations

Despite the promising features of optimal DGIGA schemes discovered, it must be pointed out that these only occur for relatively high values of J, which could be impractical. Moreover, as in every optimization

problem, these results were entirely dependent on the chosen objective function. Furthermore, the cost model employed is based on number of floating point operations; this has two major limitations:

- It is entirely dependent on the implementation in question.
- More FLOPs are not necessarily correlated with higher computational cost (CPU-hours).

15.3.2. Recommendations

A high resolving efficiency was assumed to be beneficial in scale resolving simulations. Rigorously testing the extent to which this is the case could be a promising direction for future work. The same applies to the ratio between dispersion and dissipation as an indication of LES/DNS potential. It would also be interesting to further clarify the role and relative importance of order of accuracy and spectral properties in scale resolving simulations.

A more meaningful measure of computational cost could be based on the idea in [129, §3.2]. It consists on expressing the cost associated with a specific implementation of a method as a nondimensional value; the reference value used to nondimensionalize said cost is the one associated with a particular run of the benchmark code TauBench [1]. Cost, in this approach, is actual wall-clock time (e.g. CPU-hours) necessary to obtain a result with certain accuracy. The influence of hardware is minimized by making the magnitude dimensionless.

15.4. Nonlinear physics

Both a priori modified wavenumber analysis and combined-mode analysis are semi-analytical tools that quantify linear wave propagation physics. Next, I set out to show that the error in the solution is influenced by the spectral properties predicted via combined-mode analysis. I was able to confirm this for the linear advection equation. For Burgers equation, however, the advantage was barely perceptible, if at all. This aspect of the thesis, i.e. whether the linear wave propagation characteristics of DGIGA, FR/CPR and DGSEM can be used to predict suitability for scale resolving simulations, remains inconclusive.

A posteriori modified wavenumber was generalized from finite volumes to DG (FR/CPR and DGIGA included), and applied to the Euler system. I had hoped that this would allow a characterization of the spectral properties of a discretization directly for the nonlinear case. However, I was able to show that its predictions are incompatible with those of combined-mode analysis in the higher-than-first-order case (whenever there is a multiplicity of eigenmodes). I therefore decided not to pursue this idea further.

15.4.1. Limitations and Recommendations

The fact that I was unable to detect a clear advantage for DGIGA in the nonlinear case is not evidence that said advantage is not there. I would suggest the Taylor-Green vortex (a two-dimensional problem for the Euler equations) as a more conclusive test case in that regard, based on the amount of attention it has received in the high-order literature.

15.5. Discontinuous solutions, limiters and sensors

In the last chapter of this thesis, I finally turned my attention to the Euler equations and the issue of limiting. I compared the TVB slope limiter, three hierarchical modal limiters, a simple WENO one, as well as algebraic flux correction (for DGIGA). The hierarchical moment limiter of Krivodonova was found to be the best of these. Among Legendre-based limiters, HWENO performed worst by far.

When used in combination with Krivodonova's limiter, optimal DGIGA results were comparable to those of DGSEM and optimal FR/CPR.

Using the proposed formulation, no DGIGA-AFC configuration is able to even compare against Krivodonova's hierarchical limiter. The least diffusive configuration corresponded to using a single patch throughout the whole domain (IGA-AFC). Even then, however, accuracy was sub-par; comparable to that of the TVDM slope limiter, which is regarded as outright unusable in the literature, due to its tendency of needlessly diffusing smooth extrema.

I additionally tested two sensors, and found one of them, KXRCF, to be beneficial in relation to using none at all. However, this seemed to depend on the specific test case being solved. For a test case roughly mimicking a shock wave–turbulence interaction, I found that the combination of DGSEM with

the KXRCF sensor and the TVDM limiter was about as accurate as the same scheme and sensor but with Krivodonova's limiter instead. Yet, DGIGA-AFC in combination with this same sensor, was much less accurate.

Lastly, I showed that, even with the best sensor-limiter combinations, increasing the order of accuracy alone is not an effective way to increase accuracy in solutions that contain shocks or other discontinuous features. I blame this on the lack of h-refinement that would compensate the p-coarsening around discontinuities.

15.5.1. Limitations and recommendations

No limiter currently seems to exists that is entirely satisfying; this is clear among the numerically tested ones, and there is consensus in the literature in this regard. Popular approaches not considered in this work include artificial viscosity, as well as more elaborate WENO limiters. An even more promising approach, albeit significantly more involved, would be to capture discontinuities by adapting both discretization degree and mesh cell size around them.

15.6. Is DGIGA-AFC well-suited to LES of high-speed flows?

Regarding the DGIGA-AFC combination, results are conclusive: the proposed formulation is a failure. AFC diffuses sharp features excessively and it does not preserve accuracy at smooth extrema. While not as pronounced, this also happens in IGA (i.e. single patch DGIGA) and is, therefore, not a consequence of my extension of AFC from IGA to DGIGA only. My impression, in light of the fact that AFC is much more successful when used as a means of constraining the L^2 projection of an initial condition, is that it is the linearized version of AFC that is at fault.

When it comes to DGIGA on its own, however, my answer is more nuanced: I have found no strong reason to discard it as a valid alternative to either DGSEM nor FR/CPR. In fact, DGIGA does offer the promise of significant advantages for application in LES and DNS if we take at face value its numerical dissipation and dispersion properties. Keeping in mind that lower degree implies reduced bandwidth in the discrete operator matrices and larger maximum stable time-step sizes, the possibility of DGIGA being able to use a lower degree than DGSEM and FR without compromising wave propagation accuracy could, in principle, be used to justify it as a better method. Recall, too, that the condition number of B-spline mass and control Vandermonde matrices grows exponentially with B-spline degree—the previous consideration thus also mitigates this problem. That being said, the only evidence I could find in support of the idea that the good wave propagation properties observed in the linear advection case actually carry over to nonlinear ones, was the fact that all three schemes (even with DGIGA being of lower degree) produce practically identical results in the Shu-Osher test case when stabilized with the hierarchical limiter of Krivodonova. Furthermore, in my implementation at least, DGIGA remains significantly more costly per degree of freedom than both alternatives considered.

The weak inter-patch coupling proposed for IGA in this work has been shown to be effective for the most part—the unexplained nonlinear instability issue notwithstanding. This is also very relevant, because of the ill-conditioning of the control Vandermonde matrix[1] becoming exponentially more pronounced as the number of breakpoints per patch grows (figure 6.11; note that this is *not* the case for the mass matrix, though, as seen in figure 6.10). By subdividing the domain into C^{-1}-coupled (DG) patches, themselves divided in turn into C^0–$C^{\mathrm{p}-1}$-coupled breakpoint spans, moderately high approximation degrees ($\mathrm{p} \leq 10$) and associated orders can be achieved while maintaining reasonable condition numbers, both patch-wide and mesh-wide.

15.7. Recommendations for future work

The possibility remains that aspects not studied in this thesis, most relevant the advantages of an isogeometric formulation in relation to meshing, could outweigh its cost overhead—which is the most salient limitation encountered in this work for DGIGA itself. It is for this reason that I encourage future research on IGA to be carried out in two or three dimensions, and for Euler or even Navier-Stokes physics directly: so that focus is placed on its geometric representation advantages, and that any comparisons with more conventional schemes can be made directly for the relevant, nonlinear physics. Alternatively,

[1] The control Vandermonde matrix is a necessary ingredient of the modal formulation of IGA I have proposed; recall that I have also shown that the nodal treatment is first-order-accurate only.

if interest is on large eddy simulation, study of the one-dimensional *viscous* Burgers equation could also be a very adequate next step.

A

Modified Wavenumber Analysis

This appendix describes the procedure used in this work to characterize the wave propagation (i.e. diffusion and dispersion) and linear stability properties of the three methods studied in detail, namely: discontinuous Galerkin spectral element method (DGSEM), flux reconstruction (FR/CPR) and discontinuous Galerkin isogeometric analysis (DGIGA). It is largely based on the excellent account of the approach given in [116, appendix B], with the notion of a *modified wavenumber* incorporated from [48, appendix C]. The same approach particularized for modal DG and FR/CPR, along with results, can be found in [53] and [120], respectively.

A.1. Discrete wavenumbers

Fundamentally, the approach described in this appendix can be regarded as a generalization of Von Neumann analysis [50, §7.2] from a low-order finite volume context to a high-order finite element one. Accordingly, the aim is to study the evolution in time of a monochromatic Fourier wave (in a fully linear setting and with no influence from boundary conditions) by comparing the exact (i.e. continuous) and approximate (i.e. discrete) situations. In the continuous case, one may consider an infinite one-dimensional domain $x \in [-\infty, \infty]$ in which any function satisfying certain conditions (see e.g. [19, section 2.1]) can be represented as:

$$q(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \widehat{q}(\kappa) \mathrm{e}^{\mathrm{i}\kappa x} \, \mathrm{d}\kappa \,, \tag{A.1}$$

where $\mathrm{i} = \sqrt{-1}$. The previous can be regarded as defining $q(x) : \mathbb{R} \to \mathbb{R}$ as the *inverse Fourier transform* of $\widehat{q}(\kappa) : \mathbb{C} \to \mathbb{C}$. The wavenumber, defined as the number of wavelengths (λ) in a 2π period, takes any real value, $\kappa \in \mathbb{R}$, in accordance to the wavelengths ranging continuously from infinity to zero, and every Fourier wave having two possible orientations (each associated to a sign) except for $\kappa = 0$, which represents a constant mode.

A discrete version of the situation described in the previous paragraph is required. The first step is to construct a version of the previous infinite domain in such a way that does not require an infinite number of degrees of freedom. This is possible if we instead consider a particular domain, Ω, and let it repeat periodically to cover the entire real line. From here on, $q(x)$ is defined over $x \in \Omega$. As a consequence, only a discrete number (yet still infinitely many) of wavenumbers are allowed, with a maximum wavelength and minimum wavenumber (both corresponding to the 1st wavemode) equal to:

$$\lambda_1 = L \iff \kappa_1 = \frac{2\pi}{L} \,, \tag{A.2}$$

where L is the size of Ω. All other modes will be multiples of this fundamental wavenumber[1], except for the trivial case of a constant $q(x)$—for which we reserve the zeroth wavemode:

$$\lambda_0 = \infty \iff \kappa_0 = 0 \,. \tag{A.3}$$

[1] Note that the present derivation discards any wave that does not repeat itself in a period of L. This is not the same as having periodic boundary conditions on the edges of Ω—in the latter situation, wavelengths $2L, {}^2/_3 L, {}^2/_5 L, \ldots$ would be allowed.

Equation (A.1) is thus replaced by the *Fourier series*:

$$q(x) = \sum_{n=-\infty}^{\infty} \widehat{q}_n \mathrm{e}^{\mathrm{i}\kappa_n x}\,, \tag{A.4}$$

with $n \in \mathbb{Z}$. Equation A.4 can be readily interpreted as the original function being expanded into the infinite-dimensional space spanned by the complex exponential functions (i.e. Fourier basis functions). Still, there is no lower bound for the wavelength—i.e. upper bound for the wavenumber.

Next, a *uniform* discontinuous *finite element discretization* is applied. For all discretization methods considered in this work, the fundamental repeating unit is the sub-domain Ω_k, discontinuously coupled to its two neighbors through a numerical flux. Any such element or patch is designated the *generating pattern* of the discretization. For the discretization to be amenable to a Fourier representation, uniformity is required at the generating pattern level. It must be both geometrical (i.e. all generating patterns need to have the same size) as well as in terms of the finite dimensional spaces used in each one of them. Within a generating pattern, however, no such restriction applies; therefore, *nonuniform* knot vectors are not excluded from the analysis. Accordingly, Ω is subdivided into K patches; not only is every patch (Ω_k) geometrically identical the the rest, but also each employs the same trial and test function spaces as every other. The number of dimensions of these is J. All in all, there are $\mathrm{N}_{\mathrm{dofs}} \coloneqq \mathrm{JK}$ degrees of freedom in every spatial period of the discrete signal, $L = \mathrm{K}\Delta x$, and $\Delta x/\mathrm{J}$ represents a characteristic length associated with the distance between two degrees of freedom.

Once more, the allowable wavenumbers in the approximate solution have been restricted. This time, however, an upper bound to the wavelength appears: by the Nyquist-Shannon sampling theorem, a minimum of *two degrees of freedom* are required to capture any given frequency [117]. As a consequence, the smallest (positive) wavelength and largest wavenumber resolvable are:

$$\lambda_{\mathrm{Nyq}} = \frac{2\Delta x}{\mathrm{J}} \iff \kappa_{\mathrm{Nyq}} = \frac{\mathrm{J}\pi}{\Delta x}\,. \tag{A.5}$$

At this point, finally, κ has become *both* bounded and discrete. The wavenumbers that can be resolved within a spatial period (domain Ω), split into an *even* number of patches K, each of them employing J basis functions, are:

$$\kappa_n = n\frac{2\pi}{L} \equiv n\frac{2\pi}{\mathrm{K}\Delta x} \equiv n\frac{\mathrm{J}\pi}{\Delta x \frac{\mathrm{N}_{\mathrm{dofs}}}{2}}, \quad n \in \left[-\frac{\mathrm{N}_{\mathrm{dofs}}}{2}, \frac{\mathrm{N}_{\mathrm{dofs}}}{2}\right] \subset \mathbb{Z}\,. \tag{A.6}$$

A conventional discrete version of A.4, assuming $\mathrm{N}_{\mathrm{dofs}}$ is even, is [19, p. 48]:

$$q^h(x) = \sum_{n=-\mathrm{N}_{\mathrm{dofs}}/2}^{\mathrm{N}_{\mathrm{dofs}}/2-1} \widehat{q}^h_n \mathrm{e}^{\mathrm{i}\kappa_n x}\,, \tag{A.7}$$

or, if $\mathrm{N}_{\mathrm{dofs}}$ is odd:

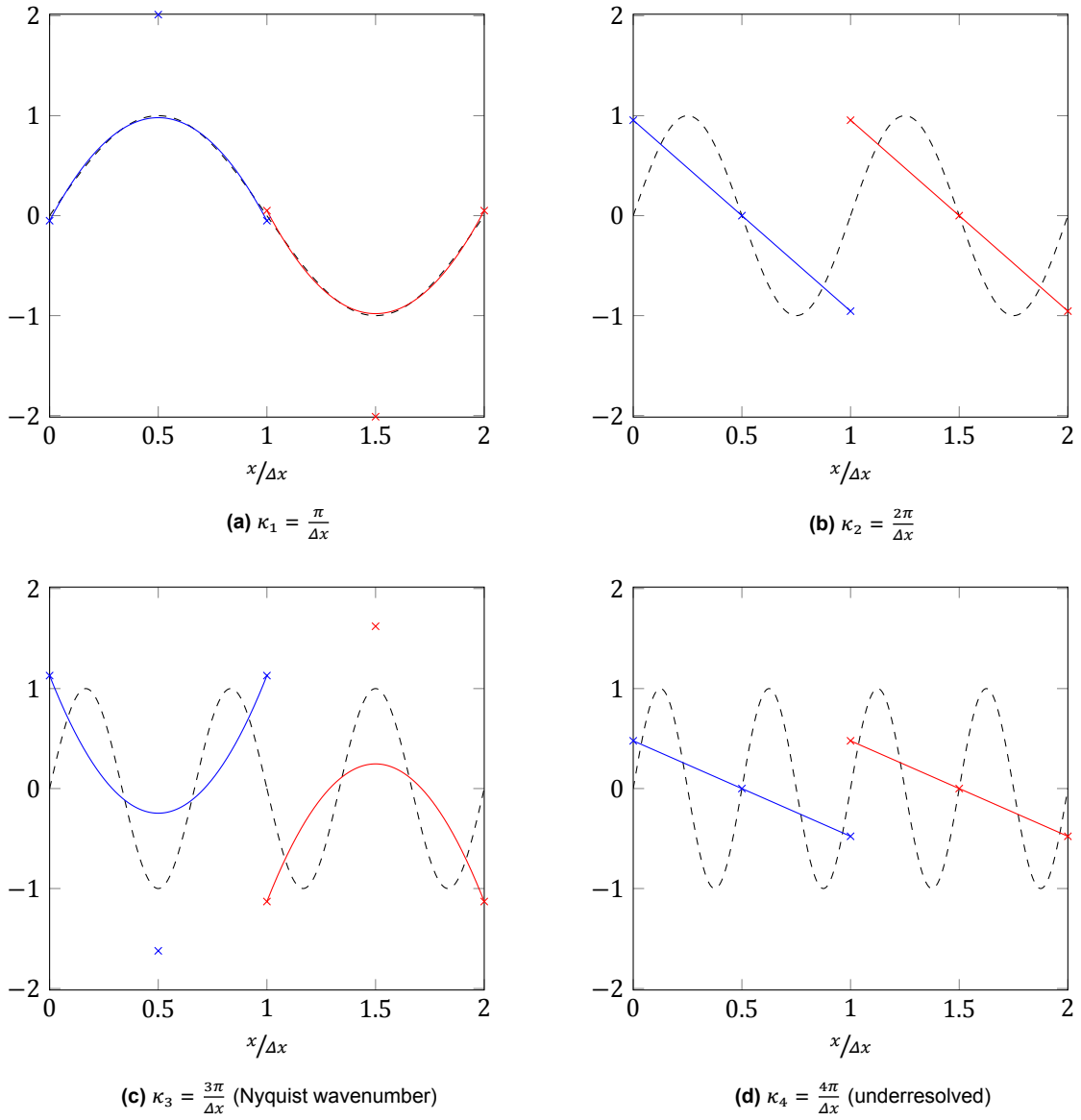
$$q^h(x) = \sum_{n=-(\mathrm{N}_{\mathrm{dofs}}-1)/2}^{(\mathrm{N}_{\mathrm{dofs}}-1)/2} \widehat{q}^h_n \mathrm{e}^{\mathrm{i}\kappa_n x}\,. \tag{A.8}$$

In these, the total number of degrees of freedom, $\mathrm{N}_{\mathrm{dofs}}$, is *preserved*: the total number of distinct wavemodes (including both orientations, positive and negative) that can be resolved in the periodically bounded domain is also equal to $\mathrm{N}_{\mathrm{dofs}}$. Equations (A.7) and (A.8) express the approximate solution in Ω as the linear combination of $\mathrm{N}_{\mathrm{dofs}}$ Fourier waves.

A.2. Wave propagation

Let us consider the simplest hyperbolic problem—the advection equation (2.19)—written in dimensionless form as follows:

$$\frac{\partial q}{\partial t^*} + \frac{\partial q}{\partial x^*} = 0\,, \tag{A.9}$$



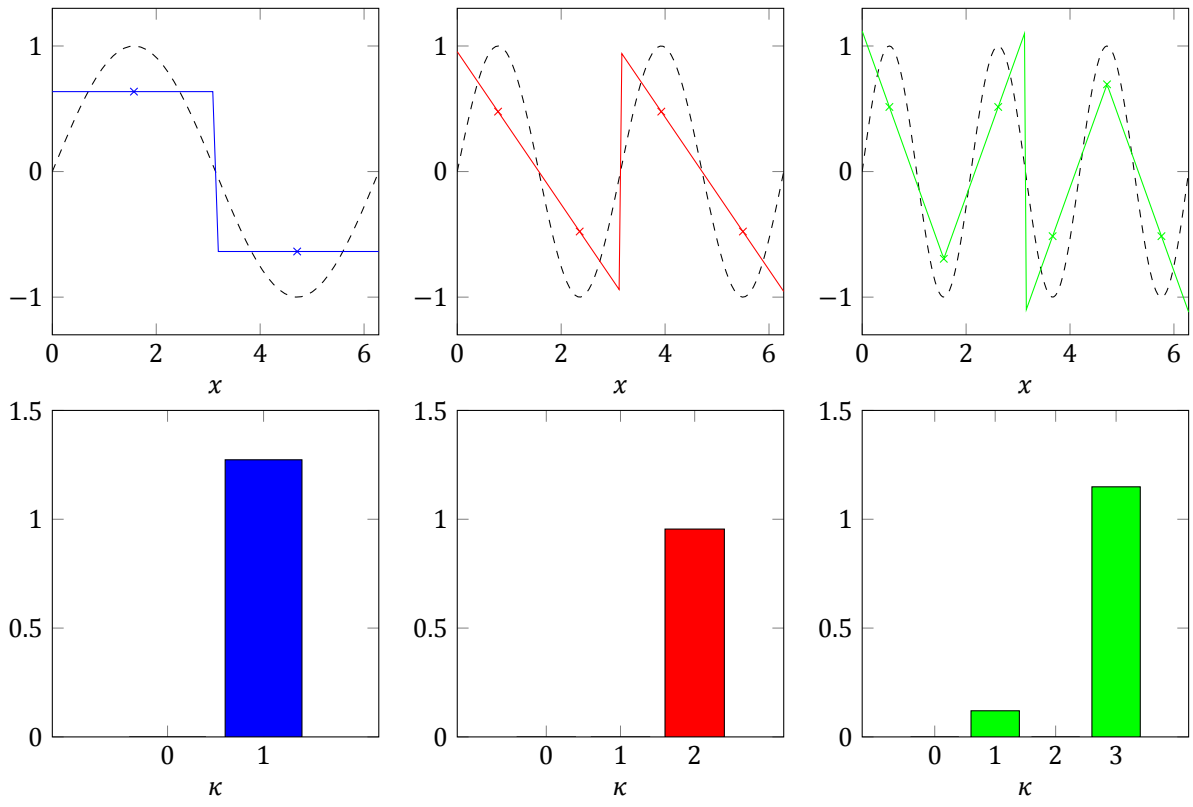
(a) $\kappa_1 = \frac{\pi}{\Delta x}$

(b) $\kappa_2 = \frac{2\pi}{\Delta x}$

(c) $\kappa_3 = \frac{3\pi}{\Delta x}$ (Nyquist wavenumber)

(d) $\kappa_4 = \frac{4\pi}{\Delta x}$ (underresolved)

Figure A.1: Fourier waves (dashed, black lines) and L^2-projected discrete counterparts, for a periodic domain discretized into 2 DGIGA patches (blue and red) of 3 degrees of freedom—in this case, control points—each (cross markers). The fourth mode, which cannot be resolved with the available degrees of freedom, is aliasing the second. The knot vector of each patch is: $\Xi = \begin{bmatrix} -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}$.



(a) $\Xi = [-1 \quad 1]$ **(b)** $\Xi = [-1 \quad -1 \quad 1 \quad 1]$ **(c)** $\Xi = [-1 \quad -1 \quad 0 \quad 1 \quad 1]$

Figure A.2: Smallest wavelength Fourier mode resolvable (top, dashed) and its L^2 projection into finite-element space (top, solid), along with the latter's single-sided discrete Fourier spectrum[2] (bottom); for three DGIGA discretizations of $\Omega = [0, 2\pi]$, each using 2 patches with knot vector Ξ.

with a nondimensionalization such that:

$$t = \frac{\Delta x}{a} t^* , \qquad x = \Delta x x^* , \tag{A.10}$$

Δx being the size of each patch in the uniform discretization, a the advection velocity, and $q(x^*, t^*)$ the scalar quantity being advected (assumed dimensionless without significant loss of generality).

A.2.1. Exact dispersion relation

By the principle of superposition, it is sufficient to focus on an exact solution function that consists of a single (real-valued) Fourier wave in space:

$$q(t^*, x^*) = \hat{q}(t^*, \kappa^*) \mathrm{e}^{\mathrm{i}\kappa^* x^*} , \tag{A.11}$$

all information regarding *amplitude* and *phase* as a function of time is contained in $\hat{q} : (\mathbb{R}, \mathbb{R}) \to \mathbb{C}$. One way to encode this dependence is to define $\widehat{\hat{q}} : \mathbb{R} \to \mathbb{C}$ such that:

$$\hat{q}(t^*, \kappa^*) = \widehat{\hat{q}}(\kappa^*) \mathrm{e}^{\theta^* t^*} , \tag{A.12}$$

with $\theta^* \in \mathbb{C}$ (at least, a priori). Plugging (A.11) into (A.9) gives:

$$\theta^* + \mathrm{i}\kappa^* = 0 , \tag{A.13}$$

a result known as the *exact dispersion relation* (in dimensionless form). Cast back in dimensional terms, it reads $\theta = -\mathrm{i}\kappa a \iff \kappa = \mathrm{i}\theta/a$. The exact behavior of a Fourier mode of wavenumber κ subject to

[2] Obtained by applying the *fast Fourier transform* (FFT) algorithm to a uniform J points-per-patch sampling of the approximate solution, $q^h(x)$.

linear advection is, therefore, described by the function:

$$q(t,x) = \widehat{\widetilde{q}}(\kappa)\mathrm{e}^{\mathrm{i}\kappa(x-at)}\,. \tag{A.14}$$

In the literature, $\Re\left(\theta\right)$ is known as the *exact dissipation rate*, and it being zero implies that the exact solution does not change in amplitude as time goes on; this is the ideal *non-dissipative* behavior. In turn, $-\Im\left(\theta\right)$ is the *exact angular frequency*; (A.14) reveals that it being equal to $a\kappa$ implies that the exact solution is advected with speed a (regardless of the wavenumber)—the ideal *non-dispersive* case. The previous considerations give rise to the *modified wavenumber analysis*, in which the imaginary and real parts of $\tilde{\kappa}^* = \mathrm{i}\tilde{\theta}^*$ (the analogue to κ^*, corresponding to the solution obtained with a particular spatial discretization method) are compared with these exact ones. This quantity, $\tilde{\kappa}$, is the *modified wavenumber*.

A.2.2. Modified wavenumber

Let us consider the arbitrary instant t, and the wavenumber κ_n (the n subscript indicates the specific instance associated to this wavemode). The exact solution (A.14), projected[3] into the finite-dimensional trial space $S_k^h(\widetilde{\Omega})$, will define a vector of time-dependent degrees of freedom, $\hat{\boldsymbol{q}}_k^{L^2} : \mathbb{R} \to \mathbb{R}^{\mathrm{J}\times 1}$, such that:

$$\int_{-1}^{1} \widehat{\widetilde{q}}(\kappa_n^*)\mathrm{e}^{\mathrm{i}\kappa_n^*\left(k+\frac{1+\xi}{2}-t^*\right)}\boldsymbol{\phi}\,\mathrm{d}\xi = \widetilde{\boldsymbol{\mathcal{M}}}\hat{\boldsymbol{q}}_k^{L^2}(t^*)\,, \tag{A.15}$$

where $\boldsymbol{\phi} = \left[\phi_1(\xi),\ldots,\phi_{\mathrm{J}}(\xi)\right]^{\mathsf{T}}$ is the (column) vector of basis functions that span $S_k^h(\widetilde{\Omega})$. The mapping to/from reference element space is explained in §3.1.1; the element index appears in the exponent because $x_k/\Delta x = k$ (hence assuming, without loss of generality, that $x_1 = \Delta x$). We can then define the following vectors of *degree of freedom–wise amplitudes* (time-dependent and not, respectively):

$$\hat{\boldsymbol{q}}_n^{L^2}(t^*) \coloneqq \widehat{\hat{\boldsymbol{q}}}_n^{L^2}\mathrm{e}^{-\mathrm{i}\kappa_n^* t^*}\,, \qquad \widehat{\hat{\boldsymbol{q}}}_n^{L^2} \coloneqq \widehat{\widetilde{q}}(\kappa_n^*)\left(\widetilde{\boldsymbol{\mathcal{M}}}\right)^{-1}\int_{-1}^{1}\mathrm{e}^{\mathrm{i}\kappa_n^*\left(\frac{1+\xi}{2}\right)}\boldsymbol{\phi}\,\mathrm{d}\xi\,, \tag{A.16}$$

and use them to express the degrees of freedom of the *projection of the exact solution into finite-element space*, in direct analogy with (A.11) and (A.14), as:

$$\hat{\boldsymbol{q}}_k^{L^2}(t^*) = \hat{\boldsymbol{q}}_n^{L^2}(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^* k} = \widehat{\hat{\boldsymbol{q}}}_n^{L^2}\mathrm{e}^{\mathrm{i}\kappa_n^*(k-t^*)}\,. \tag{A.17}$$

It is reasonable to seek an approximate solution similar to (A.17), in the sense of its degrees of freedom satisfying:

$$\hat{\boldsymbol{q}}_k(t^*) = \hat{\boldsymbol{q}}_n(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^* k}\,, \tag{A.18}$$

i.e. such that any errors due to the discrete spatial derivative operator[4] are concentrated in the still time-dependent (and now also wavenumber-dependent) but no longer element-local, array of degrees of freedom, $\hat{\boldsymbol{q}}_n(t^*)$. This makes it possible to represent the spatial residual operator as a matrix (recall that we are dealing with a linear problem) $\boldsymbol{\mathcal{R}}_n^* \in \mathbb{C}^{\mathrm{J}\times\mathrm{J}}$; its derivation is detailed in §A.3. With it, the semi-discrete advection equation (A.9) becomes:

$$\frac{\mathrm{d}\hat{\boldsymbol{q}}_k}{\mathrm{d}t^*} = \boldsymbol{\mathcal{R}}_n^*\hat{\boldsymbol{q}}_k\,. \tag{A.19}$$

The eigendecomposition of $\boldsymbol{\mathcal{R}}_n^*$ results in a matrix of (right, column) eigenvectors and a diagonal matrix of eigenvalues,

$$\boldsymbol{V}_n^* = \begin{bmatrix}\boldsymbol{v}_{1\,n}^* & \boldsymbol{v}_{2\,n}^* & \ldots & \boldsymbol{v}_{\mathrm{J}\,n}^*\end{bmatrix}\,, \qquad \boldsymbol{\Theta}_n^* = \begin{bmatrix}\tilde{\theta}_{1\,n}^* & & 0\\ & \ddots & \\ 0 & & \tilde{\theta}_{\mathrm{J}\,n}^*\end{bmatrix}\,, \tag{A.20}$$

such that:

$$\boldsymbol{\mathcal{R}}_n^*\boldsymbol{V}_n^* = \boldsymbol{V}_n^*\boldsymbol{\Theta}_n^*\,. \tag{A.21}$$

[3] Details are in §§3.2 and 3.5; for FR/CPR, trial functions (instead of test functions) are used in the projection, for the reasons given in §5.3.1.

[4] Note that time is kept continuous in the present *semi-discrete* wave propagation analysis, i.e. it approximates the limit $\Delta t \to 0$.

Let us use (A.18) and this factorization, (A.20), to re-state (A.19) as:

$$\frac{\mathrm{d}}{\mathrm{d}t^*}\left(\left(\boldsymbol{V}_n^*\right)^{-1}\widehat{\boldsymbol{q}}_n(t^*)\right) = \boldsymbol{\Theta}_n^*\left(\left(\boldsymbol{V}_n^*\right)^{-1}\widehat{\boldsymbol{q}}_n(t^*)\right), \tag{A.22}$$

which is actually a trivial system of ODEs, in which each equation is decoupled from the rest; a solution vector that satisfies it, and has the form of (A.12), is:

$$\left(\boldsymbol{V}_n^*\right)^{-1}\widehat{\boldsymbol{q}}_n(t^*) = \begin{bmatrix} \widehat{\widehat{q}}_{1n}\mathrm{e}^{\tilde{\theta}_{1n}^* t^*} \\ \vdots \\ \widehat{\widehat{q}}_{\mathrm{J}n}\mathrm{e}^{\tilde{\theta}_{\mathrm{J}n}^* t^*} \end{bmatrix}, \tag{A.23}$$

with $\widehat{\widehat{q}}_{jn} \in \mathbb{C}$. At some initial instant ($t^* = 0$, without loss of generality), it must hold that $\hat{\boldsymbol{q}}_k^{L^2}(0) = \hat{\boldsymbol{q}}_k(0)$. The array of *eigenmode-wise amplitudes*, therefore, must be:

$$\widehat{\widehat{\boldsymbol{q}}}_n \coloneqq \begin{bmatrix} \widehat{\widehat{q}}_{1n} \\ \vdots \\ \widehat{\widehat{q}}_{\mathrm{J}n} \end{bmatrix} = \left(\boldsymbol{V}_n^*\right)^{-1}\widehat{\boldsymbol{q}}_n^{L^2}. \tag{A.24}$$

Comparing equations (A.23) and (A.12) leads to the conclusion that each of the eigenvalues of the residual matrix, $\tilde{\theta}_{jn}^*$, plays a similar role as θ^* does in the exact dispersion relation (§A.2.1). This justifies the definition of a set of *modified wavenumbers* (one for each eigenvalue) that contain all information related to the wave propagation properties of a given spatial semi-discretization. Each modified wavenumber is defined, by analogy with (A.13), so that:

$$\tilde{\kappa}_{jn}^* \coloneqq \mathrm{i}\tilde{\theta}_{jn}^*. \tag{A.25}$$

All in all, we now have an expression in which the degrees of freedom of the approximate solution are the result of J eigenvector-shaped contributions (hence my use of the term *eigenmodes*[5]), each of the form of (A.17) and (A.14):

$$\hat{\boldsymbol{q}}_k(t^*) = \sum_{j=1}^{\mathrm{J}} \boldsymbol{v}_{jn}^* \widehat{\widehat{q}}_{jn}\mathrm{e}^{\mathrm{i}\left(\kappa_n^* k - \tilde{\kappa}_{jn}^* t^*\right)}, \tag{A.26}$$

and so, we see that the degree of freedom–wise Fourier coefficients in (A.18) must satisfy:

$$\widehat{\boldsymbol{q}}_n(t^*) = \boldsymbol{V}_n^* \widehat{\boldsymbol{q}}_n(t^*), \tag{A.27}$$

if we define $\tilde{\boldsymbol{\kappa}}_n^* \coloneqq \begin{bmatrix} \tilde{\kappa}_{1n}^* & \dots & \tilde{\kappa}_{\mathrm{J}n}^* \end{bmatrix}^\intercal$, and:

$$\widehat{\boldsymbol{q}}_n(t^*) \coloneqq \widehat{\widehat{\boldsymbol{q}}}_n \odot \mathrm{e}^{-\mathrm{i}\tilde{\boldsymbol{\kappa}}_n^* t^*}, \tag{A.28}$$

where $\odot$ represents the Hadamard (i.e. element-wise) product between two vectors or matrices, and the exponential function applied to a vector is to be understood as the vector of element-wise exponentials. It is worth pointing out that:

$$\left(\boldsymbol{V}_n^* \widehat{\widehat{\boldsymbol{q}}}_n\right) \odot \mathrm{e}^{-\mathrm{i}\tilde{\boldsymbol{\kappa}}_n^* t^*} \neq \boldsymbol{V}_n^*\left(\widehat{\widehat{\boldsymbol{q}}}_n \odot \mathrm{e}^{-\mathrm{i}\tilde{\boldsymbol{\kappa}}_n^* t^*}\right). \tag{A.29}$$

A.2.3. Multiplicity of eigenmodes

Van den Abeele [116, p. 229] explains the presence of multiple modified wavenumbers in the discrete version of an exact solution that only contains a single one, with the fact that the characteristic polynomial of the residual operator matrix has periodicity 2π. The dimensionless wavenumber ranges (in line with the bounds addressed in §A.1) within the interval $-\mathrm{J}\pi \leq \kappa_n^* \leq \mathrm{J}\pi$. This allows for J modified wavenumbers being associated to each baseline wavenumber, κ_n, such that $\tilde{\kappa}_n^* = \kappa_n^* + 2l\pi$, with $l \in \mathbb{Z}$.

Moura et al. [92] point out that a single Fourier basis function will not—in general—correspond to a single basis function of the semi-discretization's trial space; the projection of the initial condition can

[5] Note that there are two kinds of "modes" present in this analysis: eigenmodes (index j) and wavemodes (index n).

therefore energize various modes in a finite element's eigenspace, and each of them will disperse and dissipate in accordance to each own's modified wavenumber. The combined effect of all these eigenmodes could therefore be defined as the "true" spectral response of the method [5].

In the present report, I use the term *physical eigenmode* to refer to the single $\tilde{\kappa}_n^*$ (among all $\tilde{\kappa}_{jn}^*$) that best approximates the exact dispersion relation in the well-resolved[6] range of wavenumbers. A simple way to obtain it for every $\kappa_n^* > 0$ is as follows:

1. Set $n = 0$ (zeroth wavemode).
2. Compute $\kappa_n^* = n\frac{2\pi}{\mathrm{K}}$.
3. Compute all $\tilde{\kappa}_{jn}^*$ associated to κ_n^*.
4. Set $\tilde{\kappa}_n^*$ to the $\tilde{\kappa}_{jn}^*$ for which $|\tilde{\kappa}_{jn}^* - \kappa_n^*|$ is closest to zero (assume it is the physical eigenmode).
5. Set $n = n + 1$ (next wavemode).
6. Repeat steps 2 and 3.
7. Set $\tilde{\kappa}_n^*$ to the $\tilde{\kappa}_{jn}^*$ for which $|\tilde{\kappa}_{jn}^* - \tilde{\kappa}_{n-1}^*|$ is closest to zero.
8. Repeat from step 5 while $n \leq \frac{\mathrm{N_{dofs}}}{2}$, then stop.

a strategy similar in spirit to that used by Hu et al. [53].

A.2.4. Dominant eigenmode

With some discretizations—such as DG with *centered* numerical fluxes ($\beta = 0$)—the notions of physical and spurious (or parasitic[7]) modes are not particularly useful [7, 92]. Instead, it is more convenient to distinguish between *primary* and *secondary* eigenmodes, with the addendum that *it is not always the case that the primary mode dominates the spectral response of the scheme*.

Asthana and Jameson [7] propose a simple way to determine which eigenmode is the dominant one, at any given wavenumber, by measuring how the approximate solution's energy is distributed among them. The energy of a Fourier mode is associated with its amplitude squared; by similarity with (A.17), the amplitudes associated with each eigenmode are the $\hat{\tilde{q}}_{jn}$ coefficients given by (A.24). The relative contribution of each eigenmode to the approximate solution's n-th wavemode can then be expected to be equal to each one's relative energy content, defined as:

$$\Gamma_{jn} \coloneqq \frac{\left|\hat{\tilde{q}}_{jn}\right|^2}{\sum_{r=1}^{\mathrm{J}} \left|\hat{\tilde{q}}_{rn}\right|^2} \in [0, 1] \ . \tag{A.30}$$

Alhawwary and Wang [5] report results supporting the hypothesis that the most energized mode is representative of the "true" spectral behavior of the semidiscretization at wavenumbers such that $\Gamma_{jn} \gg \Gamma_{rn}$, for $r \neq j$.

A.2.5. Combined mode semi-discrete analysis

A definitive answer to the question of which eigenmode, if any, should be taken as representative of the spectral behavior of the discretization can be found in the approach of Vanharen et al. [117]. These researchers forgo the notion of modified wavenumbers altogether; instead they focus directly on the *effects* of dispersion and dissipation—*phase shift* and *energy loss*, respectively, of the numerical solution in relation to its exact counterpart—for any resolvable wavenumber, at some given time instant. For a matter of consistency with the rest of this appendix, I shall present the semi-discrete version of this approach [5], which neglects any dispersion or dissipation due to the temporal discretization.

[6] Section A.5.3 establishes the criterion used in this work to define this range.

[7] Designating secondary modes as parasitic is particularly misleading, since these can actually be beneficial [92]—analyzing the physical or primary mode exclusively, can lead to an underestimation of the discretization's performance.

We have seen so far that the numerical solution to the linear advection problem with an initial condition of wavenumber κ_n^*, over the arbitrary generating pattern Ω_k, may be expressed as:

$$\widetilde{q}_k^h(t^*,\xi) = \boldsymbol{\phi}^\intercal(\xi)\widehat{\boldsymbol{q}}_n(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^* k} = \boldsymbol{\phi}^\intercal(\xi)\boldsymbol{V}_n^*\left(\widehat{\widetilde{\boldsymbol{q}}}_n \odot \mathrm{e}^{-\mathrm{i}\tilde{\boldsymbol{\kappa}}_n^* t^*}\right)\mathrm{e}^{\mathrm{i}\kappa_n^* k}\,, \tag{A.31}$$

while the L^2 projection, into this same discrete solution space, of its exact counterpart is:

$$\widetilde{q}_k^{L^2}(t^*,\xi) = \boldsymbol{\phi}^\intercal(\xi)\widehat{\boldsymbol{q}}_n^{L^2}(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^* k} = \boldsymbol{\phi}^\intercal(\xi)\widehat{\widetilde{\boldsymbol{q}}}_n^{L^2}(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^*(k-t^*)}\,. \tag{A.32}$$

Recall that the actual (i.e. not projected) exact solution, restricted to $x \in \Omega_k$, was:

$$\widetilde{q}_k(t^*,\xi) = \widehat{q}(t^*,\kappa_n^*)\mathrm{e}^{\mathrm{i}\kappa_n^*\frac{\xi+1}{2}}\mathrm{e}^{\mathrm{i}\kappa_n^* k} = \widehat{\widetilde{q}}(\kappa_n^*)\mathrm{e}^{\mathrm{i}\kappa_n^*\frac{\xi+1}{2}}\mathrm{e}^{\mathrm{i}\kappa_n^*(k-t^*)}\,. \tag{A.33}$$

All three, in general, are functions of dimensionless time and space (in reference element coordinates). In fact, let us say that they belong to the space $L^2(\widetilde{\Omega})$, of all complex-valued functions over the reference element. Let us then introduce the following *complex scalar product* operator between two functions of this kind:

$$\forall (f,g) \in \left(L^2\left(\widetilde{\Omega}\right)\right)^2 \quad \langle f,g\rangle \coloneqq \int_{-1}^{1} f(\xi)\,\mathrm{conj}\left(g\left(\xi\right)\right)\,\mathrm{d}\xi\,. \tag{A.34}$$

This operator (A.34), on one hand, can be used to determine the phase shift between two functions. In particular, the (dimensionless) *phase lag*[8] between approximate and projected solutions is [5]:

$$\Delta\Psi_n^*(t^*) \coloneqq \arg\left(\left\langle \widetilde{q}_k^h(t^*,\xi), \widetilde{q}_k^{L^2}(t^*,\xi)\right\rangle\right)\,. \tag{A.35}$$

From it, we may define *phase lead* as $-\Delta\Psi_n^*$, and *phase shift* as $|\Delta\Psi_n^*|$. On the other hand, (A.34) defines an energy norm for $L^2(\widetilde{\Omega})$ that can be used to evaluate the *amplification factor* of the approximate solution [5]:

$$G_n(t^*) \coloneqq \frac{\left\|\widetilde{q}_k^h(t^*,\xi)\right\|}{\left\|\widetilde{q}_k^{L^2}(t^*,\xi)\right\|} = \sqrt{\frac{\left\langle \widetilde{q}_k^h(t^*,\xi), \widetilde{q}_k^h(t^*,\xi)\right\rangle}{\left\langle \widetilde{q}_k^{L^2}(t^*,\xi), \widetilde{q}_k^{L^2}(t^*,\xi)\right\rangle}}\,. \tag{A.36}$$

These two quantities represent one additional way to measure numerical dispersion and dissipation errors (respectively). The advantage of this approach is that it takes into account the influence of all eigenmodes in "the right way" [117], and can thus be considered to describe the *true* spectral response of the spatial scheme [5]. For comparison purposes, the phase shift and amplification factor for a single eigenmode (e.g. physical or dominant) of modified wavenumber $\tilde{\kappa}_n^*$ are given by:

$$\Delta\Psi_n^*(t^*) = \left(\kappa_n^* - \Re\left(\tilde{\kappa}_n^*\right)\right)t^*\,, \qquad G_n(t^*) = \mathrm{e}^{\Im(\tilde{\kappa}_n^*)t^*}\,, \tag{A.37}$$

while those of the exact solution would be:

$$\Delta\Psi_n^*(t^*) = 0\,, \qquad G_n(t^*) = 1\,. \tag{A.38}$$

It is worth highlighting that scalar products of the type (A.34) between two discrete functions can be conveniently evaluated as follows. Consider, for example, (A.35). Note that:

$$\left\langle \widetilde{q}_k^h(t^*,\xi), \widetilde{q}_k^{L^2}(t^*,\xi)\right\rangle = \int_{-1}^{1}\left(\sum_{r=1}^{\mathrm{J}}\phi_r(\xi)\hat{q}_{rk}(t^*)\right)\mathrm{conj}\left(\sum_{j=1}^{\mathrm{J}}\phi_j(\xi)\hat{q}_{jk}^{L^2}(t^*)\right)\mathrm{d}\xi\,. \tag{A.39}$$

The complex conjugate operator is distributive both in addition and multiplication; therefore:

$$\mathrm{conj}\left(\sum_{j=1}^{\mathrm{J}}\phi_j(\xi)\hat{q}_{jk}^{L^2}(t^*)\right) = \sum_{j=1}^{\mathrm{J}}\phi_j(\xi)\,\mathrm{conj}\left(\hat{q}_{jn}^{L^2}(t^*)\right)\mathrm{e}^{-\mathrm{i}\kappa^* k}\,. \tag{A.40}$$

[8] For any given $z \in \mathbb{C}$, the operation $\arg(z)$ extracts its phase angle; it is equivalent to $\Im\left(\ln(z)\right)$.

Moreover, the product of sums can be expanded as the double sum over all term-by-term products, i.e.:

$$\left(\sum_{r=1}^{\mathrm{J}} \phi_r(\xi)\widehat{q}_{rn}^h(t^*)\mathrm{e}^{\mathrm{i}\kappa^* k}\right)\left(\sum_{j=1}^{\mathrm{J}} \phi_j(\xi)\,\mathrm{conj}\left(\widehat{q}_{jn}^{L^2}(t^*)\right)\mathrm{e}^{-\mathrm{i}\kappa^* k}\right) = \sum_{r=1}^{\mathrm{J}}\sum_{j=1}^{\mathrm{J}} \phi_r(\xi)\widehat{q}_{rn}^h(t^*)\phi_j(\xi)\,\mathrm{conj}\left(\widehat{q}_{jn}^{L^2}(t^*)\right). \tag{A.41}$$

Hence, the scalar product between these two discrete functions ends up being equivalent to:

$$\left\langle \widetilde{q}_k^h(t^*,\xi), \widetilde{q}_k^{L^2}(t^*,\xi)\right\rangle \equiv \sum_{r=1}^{\mathrm{J}}\sum_{j=1}^{\mathrm{J}} \widehat{q}_{rn}^h(t^*)\,\mathrm{conj}\left(\widehat{q}_{jn}^{L^2}(t^*)\right)\int_{-1}^{1}\phi_r(\xi)\phi_j(\xi)\,\mathrm{d}\xi\,, \tag{A.42}$$

and an analogous result holds for the other two pairs of functions that appear in (A.36).

A.3. Residual operators in matrix form

Any spatial semi-discretization can be seen as an operator—the residual—being applied to the degrees of freedom (each a function of time only) to give the time-derivative of the same. In the particular case of a linear differential equation and a linear discretization (including a linear numerical flux function and no limiting), this residual can be written as a linear combination of the degrees of freedom within the spatial stencil. Furthermore, if the approximate solution happens to be periodic and one employs a perfectly uniform discretization to represent it, it becomes possible to relate the degrees of freedom across various generating patterns; the result is that a single matrix is sufficient to encode the entire spatial semi-discretization operator. This is essentially an extension of the Fourier/von Neumann method to high-order methods.

In what follows, a semi-discretization matrix $\boldsymbol{\mathcal{R}}_n^*$—to be used in (A.19)—for each of the research objects of this work is deduced. Said matrix, in all three cases, can be factored as:

$$\boldsymbol{M}\boldsymbol{\mathcal{R}}_n^* = 2\boldsymbol{C} + \boldsymbol{E} + \mathrm{e}^{-\mathrm{i}\kappa_n^*}\boldsymbol{E}^- + \mathrm{e}^{\mathrm{i}\kappa_n^*}\boldsymbol{E}^+\,, \tag{A.43}$$

with $\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{E}, \boldsymbol{E}^{\pm} \in \mathbb{R}^{\mathrm{J}\times\mathrm{J}}$ defined by (A.51) in DG, and (A.55) in FR/CPR.

A.3.1. DGSEM

The DGSEM semi-discretization (4.25), particularized to the advection equation, reads:

$$\frac{\mathrm{d}\check{\boldsymbol{q}}_k^{\intercal}}{\mathrm{d}t}\boldsymbol{\mathcal{M}}_k + \left[a\check{q}\boldsymbol{l}_k^{\intercal}\right]_{\partial\Omega_k} = a\check{\boldsymbol{q}}_k^{\intercal}\boldsymbol{\mathcal{C}}_k\,, \tag{A.44}$$

where $\check{\boldsymbol{q}}_k^{\intercal} \in \mathbb{R}^{1\times\mathrm{J}}$ is the vector of unknown Lagrange basis function coefficients, as this is a scalar conservation law. Taking advantage of the fact that the spatial discretization is uniform (see §A.1), and applying (A.10), the previous is equivalent[9] to:

$$\frac{1}{2}\widetilde{\boldsymbol{\mathcal{M}}}\frac{\mathrm{d}\check{\boldsymbol{q}}_k}{\mathrm{d}t^*} + [\check{q}\boldsymbol{l}]_{\partial\Omega_k} = \boldsymbol{\mathcal{C}}^{\intercal}\check{\boldsymbol{q}}_k\,. \tag{A.45}$$

The boundary flux term is:

$$[\check{q}\boldsymbol{l}]_{\partial\Omega_k} = \check{q}_k^R\boldsymbol{l}(1) - \check{q}_k^L\boldsymbol{l}(-1)\,, \tag{A.46}$$

with $\check{q}$ being associated with a *linear* numerical flux function, such as the following:

$$\check{q}\left(q^L, q^R\right) = \frac{1+\beta}{2}q^L + \frac{1-\beta}{2}q^R\,, \tag{A.47}$$

in which the value of $\beta \in \mathbb{R}$ determines the amount of upwinding. All methods studied in this report employ the fully upwind Riemann flux[10], which is reproduced by setting $\beta = 1$. In favor of generality,

[9] Note that $\frac{\Delta x_k}{2}\widetilde{\boldsymbol{\mathcal{M}}}_k = \boldsymbol{\mathcal{M}}_k$.

[10] Typically, $0 \le \beta \le 1$ (central to upwind).

β will be kept arbitrary in this appendix; for an arbitrary element Ω_k, the left and right numerical fluxes are:

$$\check{q}_k^R = \frac{1+\beta}{2}\boldsymbol{l}^\intercal(1)\check{\boldsymbol{q}}_k + \frac{1-\beta}{2}\boldsymbol{l}^\intercal(-1)\check{\boldsymbol{q}}_{k+1}\,, \qquad \check{q}_k^L = \frac{1+\beta}{2}\boldsymbol{l}^\intercal(1)\check{\boldsymbol{q}}_{k-1} + \frac{1-\beta}{2}\boldsymbol{l}^\intercal(-1)\check{\boldsymbol{q}}_k\,. \tag{A.48}$$

It is now possible to write the semi-discrete advection equation as:

$$\widetilde{\boldsymbol{\mathcal{M}}}\frac{\mathrm{d}\check{\boldsymbol{q}}_k}{\mathrm{d}t^*} = \left(2\boldsymbol{\mathcal{C}}^\intercal + \boldsymbol{E}\right)\check{\boldsymbol{q}}_k + \boldsymbol{E}^-\check{\boldsymbol{q}}_{k-1} + \boldsymbol{E}^+\check{\boldsymbol{q}}_{k+1}\,. \tag{A.49}$$

Note that, in general, all degrees of freedom in the three patches within the stencil of the discretization appear in the previous system of ODEs. In order to simplify the problem further, it is necessary to exploit the periodicity of the discretization such that a relationship between the degrees of freedom across multiple patches can somehow be encoded in the matrix of coefficients of the system. Recalling (A.18),

$$\check{\boldsymbol{q}}_k(t^*) = \widehat{\boldsymbol{q}}_n(t^*)\mathrm{e}^{\mathrm{i}\kappa_n^* k}\,, \tag{A.50}$$

and by the properties of the exponential function, $\check{\boldsymbol{q}}_{k\pm1} \equiv \check{\boldsymbol{q}}_k\mathrm{e}^{\pm\mathrm{i}\kappa_n^*}$. Equation (A.49) can therefore be rewritten in the form of (A.43), defining:

$$\boldsymbol{M} \coloneqq \widetilde{\boldsymbol{\mathcal{M}}}\,, \tag{A.51a}$$
$$\boldsymbol{C} \coloneqq \boldsymbol{\mathcal{C}}^\intercal\,, \tag{A.51b}$$
$$\boldsymbol{E} \coloneqq (1-\beta)\boldsymbol{l}(-1)\boldsymbol{l}^\intercal(-1) - (1+\beta)\boldsymbol{l}(1)\boldsymbol{l}^\intercal(1)\,, \tag{A.51c}$$
$$\boldsymbol{E}^- \coloneqq (1+\beta)\boldsymbol{l}(-1)\boldsymbol{l}^\intercal(1)\,, \tag{A.51d}$$
$$\boldsymbol{E}^+ \coloneqq (-1+\beta)\boldsymbol{l}(1)\boldsymbol{l}^\intercal(-1)\,. \tag{A.51e}$$

A.3.2. FR/CPR

Equation (5.14), when particularized for the scalar advection equation, becomes[11]:

$$\frac{\mathrm{d}\check{\boldsymbol{q}}_k^\intercal}{\mathrm{d}t} + \frac{2a}{\Delta x_k}\left(\check{\boldsymbol{q}}_k^\intercal\boldsymbol{\mathcal{D}} + \Delta\check{q}_k^L\boldsymbol{\alpha}_L^\intercal + \Delta\check{q}_k^R\boldsymbol{\alpha}_R^\intercal\right) = 0\,, \tag{A.52}$$

which, once transposed and in dimensionless form, simplifies to:

$$\frac{\mathrm{d}\check{\boldsymbol{q}}_k}{\mathrm{d}t^*} + 2\left(\boldsymbol{\mathcal{D}}^\intercal\check{\boldsymbol{q}}_k + \boldsymbol{\alpha}_L\Delta\check{q}_k^L + \boldsymbol{\alpha}_R\Delta\check{q}_k^R\right) = 0\,. \tag{A.53}$$

This time, it is the difference between corrected and uncorrected fluxes at each edges that needs to be expanded in terms of the (nodal) degrees of freedom of the local stencil:

$$\Delta\check{q}_k^L = \check{q}_k^L - \widetilde{q}_k^h(t^*,-1) = \frac{1+\beta}{2}\boldsymbol{l}^\intercal(1)\check{\boldsymbol{q}}_{k-1} - \frac{1+\beta}{2}\boldsymbol{l}^\intercal(-1)\check{\boldsymbol{q}}_k\,, \tag{A.54a}$$
$$\Delta\check{q}_k^R = \check{q}_k^R - \widetilde{q}_k^h(t^*,1) = \frac{1-\beta}{2}\boldsymbol{l}^\intercal(-1)\check{\boldsymbol{q}}_{k+1} - \frac{1-\beta}{2}\boldsymbol{l}^\intercal(1)\check{\boldsymbol{q}}_k\,. \tag{A.54b}$$

Proceeding as in the previous subsection, $\boldsymbol{\mathcal{R}}_n^*$ in the FR/CPR case is found to adhere to (A.43) if:

$$\boldsymbol{M} \coloneqq \widetilde{\boldsymbol{\mathcal{M}}} \equiv \boldsymbol{I}\,, \tag{A.55a}$$
$$\boldsymbol{C} \coloneqq -\boldsymbol{\mathcal{C}}^\intercal \equiv -\boldsymbol{\mathcal{D}}^\intercal\,, \tag{A.55b}$$
$$\boldsymbol{E} \coloneqq (1+\beta)\boldsymbol{\alpha}_L\boldsymbol{l}^\intercal(-1) + (1-\beta)\boldsymbol{\alpha}_R\boldsymbol{l}^\intercal(1)\,, \tag{A.55c}$$
$$\boldsymbol{E}^- \coloneqq -(1+\beta)\boldsymbol{\alpha}_L\boldsymbol{l}^\intercal(1)\,, \tag{A.55d}$$
$$\boldsymbol{E}^+ \coloneqq -(1-\beta)\boldsymbol{\alpha}_R\boldsymbol{l}^\intercal(-1)\,. \tag{A.55e}$$

[11] For economy of notation, let $\boldsymbol{\alpha}^\intercal \coloneqq g'(\boldsymbol{\xi}^\intercal) \in \mathbb{R}^{1\times J}$.

A.3.3. DGIGA

Equation (6.14)—which I refer to as the *modal* flux treatment—becomes, for the advection equation:

$$\hat{\boldsymbol{F}}_k = \left(a\hat{\boldsymbol{Q}}_k \boldsymbol{N}(\boldsymbol{\xi}^\intercal)\right)\left(\boldsymbol{N}(\boldsymbol{\xi}^\intercal)\right)^{-1} \equiv a\hat{\boldsymbol{Q}}_k \,, \tag{A.56}$$

i.e. it is identical to (6.15), the *nodal* flux treatment. Once particularized to the current case, the general semi-discrete conservation law for DGIGA, (6.16), becomes precisely analogous to that of DGSEM—the only difference being the type of basis functions employed (B-splines, in this case):

$$\frac{1}{2}\widetilde{\boldsymbol{\mathcal{M}}}\frac{\mathrm{d}\hat{\boldsymbol{q}}_k}{\mathrm{d}t^*} + [\check{q}\boldsymbol{N}]_{\partial\Omega_k} = \boldsymbol{C}^\intercal\hat{\boldsymbol{q}}_k \,. \tag{A.57}$$

Consequently, the matrices in (A.43) for DGIGA are defined analogously to the DGSEM ones, (A.51).

A.4. Linear stability

Consider (A.22) again. Each of its components encodes advection for a single eigenmode, with all information associated with the spatial derivative of the solution contained in $\tilde{\theta}^*_{jn}$. Linear stability analysis, for high-order schemes, consists on determining the conditions (i.e. time-step size) under which the contribution to the solution due to the j-th eigenmode reduces (stability), increases (instability) or remains the same (equilibrium) in magnitude, as time advances. In the method of lines (see §7.1), the role of a time integration scheme is to approximate the term at the right-hand-side of:

$$\widehat{q}_{jn}(t^* + \Delta t^*) - \widehat{q}_{jn}(t^*) = \tilde{\theta}^*_{jn}\int_{t^*}^{t^*+\Delta t^*} \widehat{q}_{jn}(t^*)\,\mathrm{d}t^* \,, \tag{A.58}$$

an equation that arises when integrating a component of (A.22) in time, from the current instant, t^* (in which the solution is known), to the next, $t^* + \Delta t^*$.

As an example, consider the *implicit* Euler time-scheme (which also corresponds to the first-order implicit finite-difference approximation to the first time derivative). The integral in question is approximated as:

$$\int_{t^*}^{t^*+\Delta t^*} \widehat{q}_{jn}(t^*)\,\mathrm{d}t^* \approx \Delta t^*\widehat{q}_{jn}(t^* + \Delta t^*) \,. \tag{A.59}$$

Using this approximation in (A.58), we obtain:

$$\frac{\widehat{q}_{jn}(t^* + \Delta t^*)}{\widehat{q}_{jn}(t^*)} = \frac{\widehat{\widehat{q}}_{jn}\mathrm{e}^{\tilde{\theta}^*_{jn}(t^*+\Delta t^*)}}{\widehat{\widehat{q}}_{jn}\mathrm{e}^{\tilde{\theta}^*_{jn}t^*}} = \mathrm{e}^{\tilde{\theta}^*_{jn}\Delta t^*} \approx \frac{1}{1 - \Delta t^*\tilde{\theta}^*_{jn}} \,. \tag{A.60}$$

This shows that:

- Nontrivial initial conditions ($\widehat{\widehat{q}}_{jn} \neq 0$) have no effect on linear stability.
- Exact time integration modifies the solution by a factor of $\mathrm{e}^{\tilde{\theta}^*_{jn}\Delta t^*}$, per time-step.
- The use of an approximate time integration scheme results in some approximation to the previous.

A.4.1. Amplification factor

It is always possible (for any linear time-scheme) to follow these same steps and define an *amplification factor function*, $G : \mathbb{C} \to \mathbb{C}$, such that:

$$G(z) \approx \mathrm{e}^z \,. \tag{A.61}$$

This function only depends on the time scheme (not on the spatial operator) as long as z is allowed to take any value in the complex plane, to which G associates a complex value. In fact, the role of the spatial scheme is to restrict such values to $z = \Delta t^*\tilde{\theta}^*_{jn}$.

By treating $z \in \mathbb{C}$ as an independent variable, it is possible to study the stability of each semi-discretization separately. An iso-contour of $|G(z)|$ joins together the values of z that result in certain amplification. The set of all z for which $|G(z)| = 1$ enclose the *stability region* of the scheme, which is the collection of all z for which $|G(z)| < 1$, and for which the time-integration is stable. For implicit

Euler, the amplification factor function gives a stability region encompassing the entire complex plane except for the circle tangent to the imaginary axis with center at $z = 1$. An equivalent expression can be obtained for any linear scheme, including those with multiple stages (e.g. Runge-Kutta) or steps (e.g. Adams-Bashforth). Amplification factor functions of several SSP Runge-Kutta schemes are listed in §7.3.

A.4.2. Fourier footprint

The collection of all eigenvalues of a spatial discretization—all eigenmodes, all wavenumbers—is frequently referred to as its *Fourier footprint*. These may be thought of as curves[12] in the complex plane[13]. The (nondimensional) time-step size acts as a scaling factor for said curves. All such figures in this report are scaled so that all eigenvalues are within the $|G(z)| = 1$ contour (also shown) for a particular time scheme (3-stage, 3rd order strong-stability-preserving Runge-Kutta).

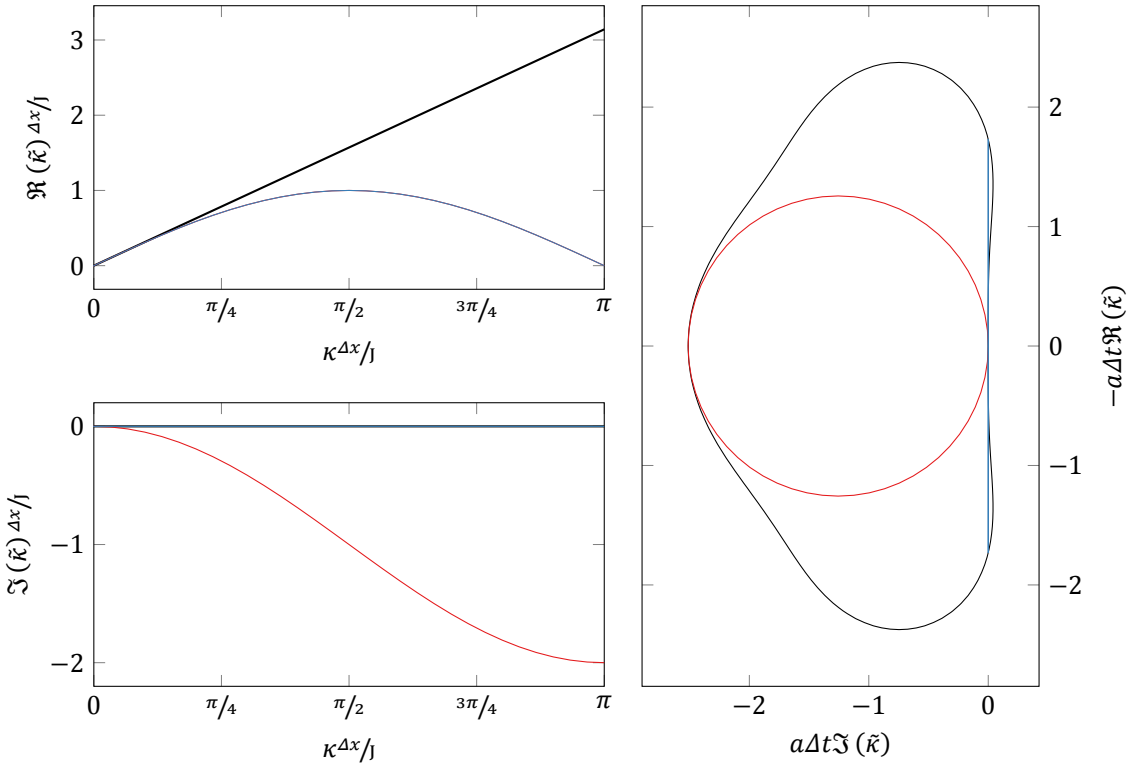


Figure A.3: Dispersion relation (top, left), dissipation relation (bottom, left) and Fourier footprint (right) of the upwind (——) and centered (——), $\mathrm{p} = 0$ discretizations. Their footprints are shown scaled by the Courant numbers 1.256 and 1.732, respectively.

A.4.3. Time-step size limits

The influence that the spatial semi-discretization scheme has on stability is thus encoded in the set of points in the complex plane—the eigenvalues of its spatial operator—which, in combination with an amplification factor function, will result in certain amplification factor magnitude. For a given Fourier mode to be stable, none of the eigenmodes associated to it can have an amplitude factor greater than one. Linear stability of a full discretization is therefore guaranteed whenever the entire Fourier footprint of the spatial semi-discretization, *scaled by the dimensionless time-step size*, lies within the stability region of the time-integration scheme it is paired with.

[12] These curves correspond to the limit as $\Delta x \to 0$. In practice, they are sampled at KJ wavenumbers, uniformly distributed around (and including) $\kappa = 0$; K is the number of generating patterns (i.e. patches or elements) within Ω, and J is the number of basis function in each.

[13] Alternatively, one can regard the previous as *projections* onto the complex plane of the three-dimensional curves shown in figure A.4.

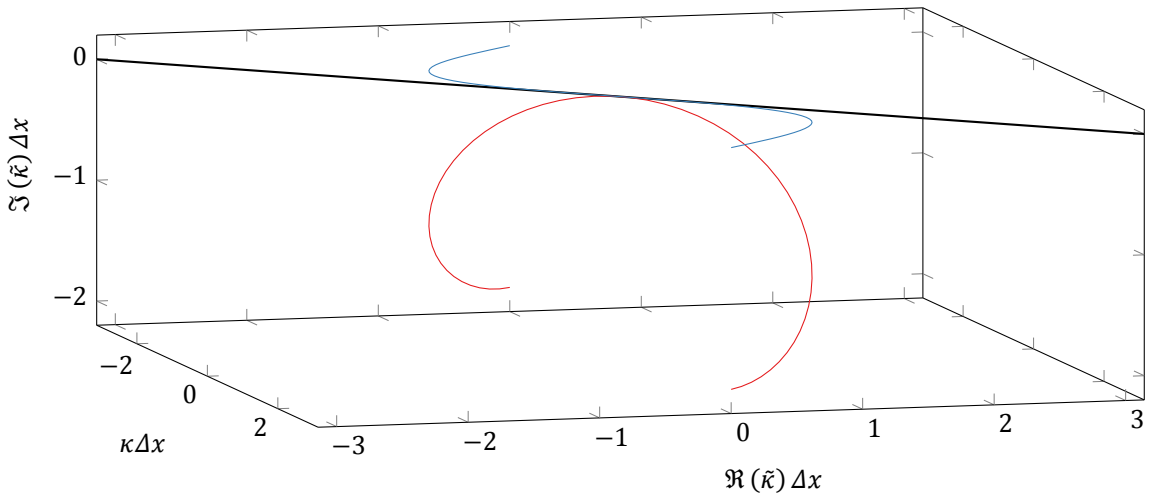


Figure A.4: Modified wavenumbers of figure A.3 (dimensionless, but with no scaling), now seen as so-called *Bloch waves* [121].

This dimensionless time-step size expresses a *ratio between the physical and numerical speeds of propagation of information*, and thus can be seen as a *Courant number* (ς). More precisely, the current nondimensionalization gives:

$$\Delta t^* = \frac{a}{\Delta x/\Delta t} \equiv \varsigma\,, \tag{A.62}$$

which matches the definition of the Courant number used in this report. Every combination of time and space semi-discretizations will have a *maximum stable Courant number*, defined as

$$\varsigma_{\max} \equiv \Delta t^*_{\max} \coloneqq \inf\left\{\Delta t^* \geq 0 \;\;:\;\; \forall j,n \;\; \left|G\left(\Delta t^* \tilde{\theta}^*_{jn}\right)\right| = 1\right\}, \tag{A.63}$$

i.e. the largest ς for which no point of the Fourier footprint of the spatial scheme falls outside the stability region of the time scheme. It is possible, nevertheless, that a discretization has $\varsigma_{\max} = 0$; it is then said to be *unconditionally unstable*.

A.5. Quantifying dispersion and dissipation

With (A.25) as the definition of the modified wavenumber, the definitions of *modified dissipation rate* and *modified angular frequency* arise naturally as analogues to their exact counterparts, derived from (A.13). Assuming that a single representative $\tilde{\kappa}^*_n$ modified wavenumber can be found for every wave-mode, it is conventional in the literature to qualitatively judge the spectral performance of a scheme by plotting $\Re(\tilde{\kappa}^*_n)$ and $\Im(\tilde{\kappa}^*_n)$ against κ^*_n, for $n \geq 0$. This leads to the definition of $\tilde{\kappa}^* : \mathbb{R} \to \mathbb{C}$ as a (dimensionless) modified wavenumber associated to the (also dimensionless) exact wavenumber κ^*. With high-order discretizations, it is typical to normalize the exact and modified (dimensionless) wavenumbers by J, the number of basis functions (i.e. degrees of freedom) per patch/element; this is because $\kappa^* \in [-\pi\mathrm{J}, \pi\mathrm{J}]$. Moreover, the following symmetry around the origin holds: $\tilde{\kappa}^*(\kappa^*) = -\operatorname{conj}\tilde{\kappa}^*(-\kappa^*)$ (real parts are antisymmetric, imaginary parts are symmetric).

A.5.1. Order of accuracy

The ability of a method to approximate the exact dispersion and dissipation relations can be measured by taking the absolute value of the (complex) difference between exact and modified wavenumbers:

$$E_T(\kappa^*) \coloneqq \left|\tilde{\kappa}^*(\kappa^*) - \kappa^*\right|\,. \tag{A.64}$$

Following [55] and [120], one may compute the following theoretical order of accuracy associated to the dispersion and dissipation errors of a spatial semi-discretization by evaluating (A.64) at a pair of

well-resolved wavenumbers, κ^* and $\frac{\kappa^*}{2}$:

$$A_T \coloneqq \frac{\ln\left(E_T\left(\kappa^*\right)\right) - \ln\left(E_T\left(\frac{\kappa^*}{2}\right)\right)}{\ln(2)} - 1\,. \tag{A.65}$$

A piecewise polynomial solution discretization of degree p, in general, can be expected to converge such that $\left|q^h - q\right| = \mathcal{O}\left(\Delta x^{\mathrm{p}+1}\right)$. In practice, however, it is sometimes the case that quantities such as the period of oscillation in an unsteady wake, or even—hypotetically—a boundary layer profile around a sufficiently smooth geometry, are *superaccurate* [8], in the sense that their error reduces at a higher-than-expected rate as the mesh is refined—typically, for DG, $\mathcal{O}\left(\Delta x^{2\mathrm{p}+1}\right)$. The quantity A_T is a theoretical estimate of said *superconvergence* order [55, 120][14].

A.5.2. Resolving efficiency

As the discretization is refined—or, equivalently, for low enough wavenumbers—$\tilde{\kappa}^* \approx \kappa^*$. The highest wavenumber that a given discretization can resolve *accurately* is another important indicator of its performance (the higher the better). Lele [81] proposed to define as the so-called *resolving efficiency* of a scheme as the fraction between its highest well-resolved wavenumber, and its highest resolvable one. This, however, still requires some criterion to distinguish between well-resolved and badly-resolved wavenumbers.

Typically, the highest well-resolved (dimensionless) wavenumber is simply defined as:

$$\kappa_f^* \coloneqq \sup\left\{\kappa^* > 0 \quad : \quad \frac{E_T(\kappa^*)}{\kappa^*} \leq \varepsilon\right\}, \tag{A.66}$$

with an error threshold in the range $0.001 \leq \varepsilon \leq 0.1$ [7, 81]—I use $\varepsilon = 0.01$. Any $\kappa^* < \kappa_f^*$ is considered to be well-resolved, and the resolving efficiency is:

$$e_1 \coloneqq \frac{\kappa_f^*}{\pi \mathrm{J}}\,. \tag{A.67}$$

The larger e_1 is, the more wavelengths can be resolved *accurately* per degree of freedom.

A.5.3. Numerical cutoff wavenumber

Different from the resolution threshold set by κ_f^*, but also of interest, is the wavenumber at which numerical dissipation starts to become significant. Numerical dissipation can act as an implicit sub-grid-scale model in LES, damping any scales beyond some specific wavenumber.

Moura et al. [92] argue for what they refer to as the "1% rule", a rule-of-thumb criterion used to define this cutoff threshold. Given a semi-discretization, let:

$$\kappa_{1\%}^* \coloneqq \sup\left\{\kappa^* > 0 \quad : \quad 1 - \mathrm{e}^{\frac{\Im(\tilde{\kappa}^*)}{\mathrm{J}}} \leq 0.01\right\}, \tag{A.68}$$

i.e. the wavenumber at which any propagating wave (regardless of its speed) has its amplitude scaled to 99%, per degree of freedom crossed[15]. Numerical evidence suggests that this criterion is able to accurately predict the beginning of the numerically induced dissipation range in underresolved turbulent flow simulations, at least for DG schemes [93].

A.5.4. Dispersion to dissipation ratio

Within the well-resolved range, it is desirable for a semi-discretization to minimize E_T. As dispersion errors start to become important, however, it is no longer optimal to have the least amount of dissipation possible (at least in LES, there will—by definition—be some underresolved wavenumbers present in the simulation). Instead, it would be desirable to have large numerical dissipation acting only on those waves that are being advected at erroneous speeds, leaving the well-resolved ones untouched. One may quantify this as follows, a slight generalization of the ideas in [3].

[14] Assuming that the Courant number is kept constant as Δx is refined [8, p. 6].
[15] See §A.5.4.

Consider, as the exact solution, a monochromatic wave of the form of (A.33), with (dimensionless) wavenumber κ^*. In any such wave, energy (and, equivalently, information) propagates through the domain at the so-called *group velocity* [85, 115, 119]:

$$v_g^* \coloneqq \frac{\mathrm{d}\kappa^*}{\mathrm{d}\kappa^*} \equiv 1 \,. \tag{A.69}$$

This speed, defined as the derivative of the angular frequency with respect to the wavenumber, is identically equal to unity for the exact advection problem under the current non-dimensionalization (A.10). In the exact solution, any perturbation (regardless of κ^*) travels along the domain at a rate of "J degrees of freedom per dimensionless unit of time", with no change in amplitude.

In the discrete case, we have seen that multiple eigenmodes arise (§A.2.2). Any energy present initially in the numerical solution will be distributed among all these modes (see §A.2.4). It is reasonable to assume, then, that the fraction of the initial energy associated with a given eigenmode—e.g. the j-th one—will affect the solution independently of the others, and in accordance to this eigenmode's wave equation. Each of these "eigenwaves" will then carry a fraction of the total energy initially available in the solution at its own *modified* group velocity:

$$\tilde{v}_{g,j}^* \coloneqq \Re\left(\frac{\mathrm{d}\tilde{\kappa}_j^*}{\mathrm{d}\kappa^*}\right) . \tag{A.70}$$

If the numerical scheme is dispersive, $v_g^* \neq \tilde{v}_{g,j}^*$. This means that, as time progresses, the energy content of the j-th eigenmode gets further and further away from its desired position (that which would correspond to the exact solution). We may, in that case (significant dispersion), associate said energy to an *undesired perturbation* traveling at a dimensionless speed of $v_g^* - \tilde{v}_{g,j}^*$ along the domain. In terms of number of degrees of freedom—of which, on average, there are J per every dimensionless unit of distance (i.e. patch width)—the reach of this spurious perturbation, as a function of (dimensionless) time and wavenumber, is: $\mathrm{J}|v_g^* - \tilde{v}_{g,j}^*|t^*$.

Nevertheless, if, in addition to being dispersive, a given semi-discretization is also dissipative, the amplitude of the aforementioned perturbation will also be subjected to damping. This is an exponential decay process ($\Im\left(\tilde{\kappa}_j^*\right) < 0$ is assumed); the time it takes for the amplitude to reduce to a given fraction $1/r$ of its initial value is:

$$\frac{1}{r} = \mathrm{e}^{\Im\left(\tilde{\kappa}_j^*\right) t_{1/r}^*} \implies t_{1/r}^* \coloneqq \frac{-\ln(r)}{\Im\left(\tilde{\kappa}_j^*\right)} \,, \tag{A.71}$$

and, for simplicity, let us define $t_{1/\mathrm{e}}$ ($r = \mathrm{e}$) to be the *lifetime*[16] of this perturbation. In conclusion, the mean lifetime—in terms of number of degrees of freedom "polluted"—of any spurious energy content in the numerical solution associated to the j-th eigenmode, is:

$$\chi_j(\kappa^*) \coloneqq \frac{\mathrm{J}\left|1 - \Re\left(\frac{\mathrm{d}\tilde{\kappa}_j^*}{\mathrm{d}\kappa^*}\right)\right|}{-\Im\left(\tilde{\kappa}_j^*\right)} , \tag{A.72}$$

a function of the (dimensionless) wavenumber only.

Furthermore, since each eigenmode may have a different amount of energy, let us define the overall effect of all eigenmodes as the weighted average between the relative energy content of each perturbation, and its lifetime:

$$\overline{\chi}(\kappa^*) \coloneqq \sum_{j=1}^{\mathrm{J}} \Gamma_j \chi_j(\kappa^*) \,. \tag{A.73}$$

The largest the spurious energy content, and the largest its lifetime, the worse. Therefore, I would argue that $\overline{\chi}$ should be as low as possible in the underresolved wavenumber range. A concise way to measure this effect is through the following quantity—the mean lifetime, averaged over all underresolved wavenumbers:

$$\left\|\overline{\chi}\right\| \coloneqq \frac{1}{\mathrm{J}\pi - \kappa_f^*} \int_{\kappa_f^*}^{\mathrm{J}\pi} \overline{\chi}(\kappa^*)\, \mathrm{d}\kappa^* \,, \tag{A.74}$$

[16] This definition is a standard convention, alternative to the perhaps more familiar *half-life*, $t_{1/2}$ ($r = 2$).

using κ_f^* from §A.5.2. Equation (A.74) purposefully excludes the well-resolved wavenumber range, for which $\overline{\chi}$ is meaningless; in practice, this means that any indetermination in χ_j as $\kappa^* \to 0$ is avoided. Adaptive quadrature can be used to compute (A.74); however, because the integrand happens to be highly non-smooth, I use the rather coarse adaptation tolerances 1×10^{-5} (absolute) and 1×10^{-3} (relative).

B

Time Complexity Estimation

This appendix details the estimation of cost per step (in FLOPs) used in §12.4. The approach, based on my implementation in particular, simply consists on selecting those methods representative of the cost of a time step (indicated in listing B.1), and tracking the number of floating point operations involved in every subroutine invoked by these. Its outcome is the cost model encoded by (12.15).

In what follows, I list every method involved in advancing the numerical solution from one step to the next, highlight those nested functions the cost of which I take into account, and indicate the number of floating-point operations made in every line of their source code—summation, subtraction, multiplication or division between two floating-point scalars; each of these counts as one FLOP. These estimates assume that the PDE being solved is the linear advection equation (2.19), such that the solution over a given element or patch is encoded by a *single* vector of J entries. Moreover, the time-scheme is SSP-RK3(3) from §7.3.3 (every time-step is subdivided into *three* stages), and periodic boundary conditions are employed at both ends of the domain.

Listing B.1: Fragment of the `Solver.stepForward` method (inherited by `SSP_RK3`) that advances the solution by one time-step. Only the two highlighted methods (and their nested function calls) will be taken into account in the cost estimation.

```
1  function STOP = stepForward(this,mesh)
2      ⟨...⟩
3      % Advance one time-step:
4      this.stageNow = 0; % reset stage counter
5      while this.stageNow < this.stageCount
6          % Advance stage counter:
7          this.stageNow = this.stageNow + 1;
8          % Evaluate solution residuals:
9          mesh.computeResiduals(this.physics,this) See B.2
10         % Advance solution by one stage:
11         for element = mesh.elements
12             this.applyStage(element) 4J FLOPs (B.16)
13         end
14         % Apply limiter after each stage:
15         this.limiter.applyStage(mesh,this)
16     end
17     % Apply limiter after a full time step (one additional time):
18     this.limiter.applyStep(mesh,this)
19     ⟨...⟩
20 end
```

The software these estimates will be deduced for (see §9.1) makes use of the object-oriented paradigm, meaning that the step update has been designed into separate and non-overlapping logical entities. As a consequence, all differences between DGIGA, FR/CPR and DGSEM (all of them

derived from a common parent class) are concentrated into the one method that computes the numerical residual's expansion coefficients. All computations made within a single time-scheme *stage* update can be grouped into the following three categories:

- Discontinuous coupling of the current solution across element interfaces
- Computation of the current numerical residual
- Computation of the numerical solution at the following time-stage

Any other costs, most of which occurring only once per simulation (e.g. projection of initial conditions, assembly of mass and gradient matrices), will be neglected. So will any overheads associated with function calls, memory access, pointer arithmetic, and similar.

B.1. Discontinuous coupling

In DG methods the coupling between elements is made via a numerical flux, typically obtained by solving a Riemann problem (see §3.3.3). This section details the cost of all methods involved in computing said interface fluxes—including the treatment of boundary conditions, which are enforced weakly (§3.6). Note that most methods in this section require knowledge of each element's two immediate neighbors, as well as the edge that each pair of elements share.

Listing B.2: Method `computeResiduals` of the `Mesh` class. Called in B.1, once per stage.

```
function computeResiduals(this,physics,solver)
    % Updates the residuals of all cells in a mesh, using the
    % spatial discretization scheme assigned to the mesh.
    %
    % Compute the state at both edges of each element:
    [this.elements.localTimeDelta] = deal(inf);
    this.elements.interpolateStateAtEdges 4JK FLOPs (B.3)
    % Also for ghost elements:
    this.boundaries.apply(physics,solver) 8J FLOPs (B.4)
    % Compute the Riemann flux at each edge:
    this.edges.computeFlux(physics) 5(K + 1) FLOPs (B.6)
    % Apply the spatial discretization operator to each element:
    this.elements.computeResiduals(physics) See B.9
end
```

Listing B.3: Method `interpolateStateAtEdges` of the `Element` class. Called in B.2 and B.5.

```
function interpolateStateAtEdges(these)
    % Evaluates and stores the state vector at these elements' edges.
    % Vector input.
    for this = these
        this.stateL = this.states*this.basis.left; 2J FLOPs
        this.stateR = this.states*this.basis.right; 2J FLOPs
    end
end
```

Listing B.4: Method `apply` of the `Boundary` class. Its role is to enforce boundary conditions in a generic manner. Called by B.2.

```
function apply(these,physics,solver)
    % Updates the ghost element of each of these boundaries
    % according to each's scalar apply method.
    these(1).apply_scalar(physics,solver,true) 4J FLOPs (B.5)
    these(2).apply_scalar(physics,solver,false) 4J FLOPs (B.5)
end
```

Listing B.5: Method `apply_scalar` of the `Periodic` class, a particular type of `Boundary`. It particularizes the ghost element approach of imposing boundary conditions to the periodic case. Used twice in B.4—once for each boundary of the mesh.

```
function apply_scalar(this,varargin)
    % Updates the ghost element's states with those of its ”real”
    % counterpart (i.e. the copy of it owned by the mesh).
    this.ghostElement.states = this.oppositeBoundElement.states;
    this.ghostElement.interpolateStateAtEdges 4J FLOPs (B.3)
end
```

Listing B.6: Method `computeFlux` of the `Edge` class, called in B.2.

```
function computeFlux(these,physics)
    % Computes and sets numerical fluxes (i.e. solutions of the
    % Riemann problem at an edge) on both elements sharing each of
    % these edges, including a conventional sign. Also sets a local
    % time-step size based on each Riemann problem's characteristic
    % speeds. Vector input.
    for this = these
        [flux,waveSpeeds] = physics.riemannFlux(...
            this.elementL.stateR,this.elementR.stateL); 1 FLOPs (B.7)
        this.elementL.riemannR = flux; % 'normal vector' = +1
        this.elementR.riemannL = -flux; % 'normal vector' = -1 1 FLOPs
        this.computeTimeDeltas(waveSpeeds); 3 FLOPs (B.8)
    end
end
```

Listing B.7: Method `riemannFlux` of the `Advection` class. It particularizes the generic Riemann solver (the outputs of which are a numerical flux and a vector of wave speeds) to the simple case of linear advection. It is called in B.6, for every edge in the mesh (boundaries included).

```
function [flux,S] = riemannFlux(this,stateL,stateR)
    S = this.advSpeed;
    if S > 0
        flux = S*stateL; 1 FLOPs
    else
        flux = S*stateR; 1 FLOPs
    end
end
```

Listing B.8: Method `computeTimeDeltas` of the `Edge` class. I use it to determine candidate time-step sizes for the next time-step. It is called in B.6, once per edge.

```
function computeTimeDeltas(this,waveSpeeds)
    % Sets local time-step sizes based on characteristic speeds and
    % element sizes, such that each local Courant number is unity.
    % Does not override more conservative existing values.
    this.elementL.localTimeDelta = min(...
        [this.elementL.localTimeDelta
        -this.elementL.dx./waveSpeeds(waveSpeeds < 0)]); 2 FLOPs
    this.elementR.localTimeDelta = min(...
        [this.elementR.localTimeDelta
        this.elementR.dx./waveSpeeds(waveSpeeds > 0)]); 1 FLOPs
end
```

Listing B.9: Method `computeResiduals` of the `Element` class. Used in B.2.

```
function computeResiduals(these,physics)
    for this = these
        this.basis.computeResiduals(this,physics); See B.10, B.13 and B.15
    end
end
```

B.2. Residual evaluation (spatial schemes)

Once the numerical fluxes ensuring a coupled solution have been made available to each element, its residuals can be evaluated completely independently from the rest. In fact, from this point onward, all methods are applied element-wise.

B.2.1. DGSEM

The first of the three types of spatial schemes under consideration is DGSEM (§4). Since its trial and test spaces are spanned by an orthogonal basis, the inverse of its diagonal mass matrix is trivial (it is applied as a vector of weighting factors). Its `derivatives` matrix, however, is full—hence the cost indicated for the matrix-vector product in line 3 of listing B.10.

Listing B.10: Method `computeResiduals` of the `DGSEM` class, a subtype of the `Basis` superclass. Called in B.9, for every element.

```
function computeResiduals(this,element,physics)
    element.computeFluxesFromStates(physics); J FLOPs (B.11)
    element.residuals =
        element.fluxes*this.derivatives 2J^2 FLOPs
        + J FLOPs
        (element.riemannR.*this.right' J FLOPs
        + J FLOPs
        element.riemannL.*this.left') J FLOPs
        ./this.nodeWeights'; J FLOPs
    element.residuals = - 2/element.dx*element.residuals; J + 1 FLOPs
end
```

Listing B.11: Method `computeFluxesFromStates` of the `Element` class. Called by B.10 and B.13.

```
function computeFluxesFromStates(this,physics)
    this.fluxes = physics.flux(this.states); J FLOPs (B.12)
end
```

Listing B.12: Method `flux` of the `Advection` class, a particular type of `Physics`. Used in B.11 and B.15.

```
function flux = flux(this,state)
    flux = this.advSpeed*state; J FLOPs
end
```

B.2.2. FR/CPR

The second type of spatial semi-discretization, FR/CPR (detailed in §5), takes it one step further: not only is its mass matrix diagonal, it is the identity. Its discrete gradient operator is, nevertheless, a full matrix (as for DGSEM). However, due to the need of evaluating the uncorrected fluxes at the element edges (these are required for the flux correction procedure), FR ends up requiring $2(J+1)$ FLOPs more than DGSEM for every residual evaluation, making this scheme slightly more costly[1]. This can be observed by comparing lisitings B.13 and B.10.

[1] Note that this would not have been the case if e.g. the distribution of nodes were Gauss-Lobatto; if the value of the fluxes was available at element edges by design, both in DGSEM and FR, the latter would be the cheapest of the two.

Listing B.13: Method `computeResiduals` of the `FR` class, another descendant of `Basis`. Called in B.9, for every element.

```
1  function computeResiduals(this,element,physics)
2      element.computeFluxesFromStates(physics); J FLOPs (B.11)
3      element.interpolateFluxAtEdges; 4J FLOPs (B.14)
4      element.residuals =
5          element.fluxes*this.derivatives 2J^2 FLOPs
6          + J FLOPs
7          (-element.riemannL - element.fluxL)*this.correctionsL J + 1 FLOPs
8          + J FLOPs
9          (element.riemannR - element.fluxR)*this.correctionsR; J + 1 FLOPs
10     element.residuals = -2/element.dx*element.residuals; J + 1 FLOPs
11 end
```

Listing B.14: Method `interpolateFluxAtEdges` of the `Element` class. Used in B.13.

```
1  function interpolateFluxAtEdges(this)
2      % Evaluates and stores the flux vector at the element's edges.
3      this.fluxL = this.fluxes*this.basis.left; 2J FLOPs
4      this.fluxR = this.fluxes*this.basis.right; 2J FLOPs
5  end
```

B.2.3. DGIGA

Lastly, the costs of the DGIGA scheme (presented in §6) are shown in listing B.15. Unlike the previous two, this semi-discretization's operators can be sparse—their half-bandwidth is at most $\mathrm{p} \leq \mathrm{J} - 1$. This is also the case for the mass matrix, which is therefore *banded* (rather than diagonal) for DGIGA. A sparse matrix-vector product requires twice as many FLOPs as the number of nonzero entries in said matrix [91]. For a B-spline basis of $\mathrm{k} \geq 1$ breakpoint spans, degree $\mathrm{p} > 0$ and smoothness $\mathcal{C}^{\varkappa} : \varkappa < \mathrm{p}$ (all three of them integers), it can be shown that the *number of non-zeroes* in both `massMatrix` and `gradientMatrix` (i.e. $\boldsymbol{\mathcal{M}}$ and $\boldsymbol{\mathcal{C}}$, respectively) is:

$$\mathrm{nnz}\left(\boldsymbol{\mathcal{M}}\right) = \mathrm{nnz}\left(\boldsymbol{\mathcal{C}}\right) = \mathrm{k}\left(\mathrm{p}+1\right)^2 - \left(\mathrm{k}-1\right)\left(\varkappa+1\right)^2 . \tag{B.1}$$

The fact that DGIGA employs a modal basis adds complexity to the computation of fluxes in the interior of the element. I do so—as explained in §6.3.1—by first evaluating the state function at the control point locations (line 2 of listing B.15), then evaluating the flux function there (line 3 of listing B.15), and finally deducing the modal flux coefficients that would produce the control point flux values just obtained (lines 4 to 7 of listing B.15). As shown in the listing, this is encoded by a multiplication with the `controlVandermonde` matrix, $\boldsymbol{N}(\boldsymbol{\xi}^{\intercal})$ (defined by equation 6.13) and its inversion. The number of nonzero entries in $\boldsymbol{N}(\boldsymbol{\xi}^{\intercal})$ bounded by (this underestimates the number of zeros by roughly up to 35 % for $\mathrm{p} \leq 20$ and $\mathrm{k} \leq 5$):

$$\mathrm{nnz}\left(\boldsymbol{N}(\boldsymbol{\xi}^{\intercal})\right) \leq \mathrm{k}\left(\mathrm{p}^2+1\right) - \left(\mathrm{k}-1\right)\left(\varkappa^2+1\right) . \tag{B.2}$$

I estimate the cost of the inverse projection—from control location samples to modal expansion coefficients—by assuming that said `controlVandermonde` matrix has previously been factored into lower and upper triangular *full*[2] matrices, `upperControlVandermonde` and `lowerControlVandermonde` respectively. Each of the two triangular system inversions associated with these matrices, therefore, requires J^2 FLOPs—J being, as in DGSEM and FR, the total number of rows or columns of $\boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{C}}$ and $\boldsymbol{N}(\boldsymbol{\xi}^{\intercal})$. As a side note: recall that for linear advection this projection to/from control locations is unnecessary altogether (see §6.3.1, equation 6.15); however, I consider that not taking the cost of these steps into account would overestimate the performance of DGIGA in a misleading way.

DGIGA's trial and test basis functions are not orthogonal. This manifests in yet another added cost, this time associated with inverting its mass matrix. On the upside, `massMatrix` is guaranteed to be

[2] Unfortunately, `controlVandermonde` is non-symmetric; pivoting might hence be required to ensure a stable factorization, which will lead to the lower/upper bandwidth of its lower/upper triangular factors being increased, possibly, all the way up to $\mathrm{J} - 1$.

symmetric-positive-definite; it is possible, therefore, to factorize it (via the Cholesky algorithm, without pivoting) into a pair of triangular matrices (`choleskyMassMatrix` and its transpose) which *preserve the bandwidth* of `massMatrix` (i.e. upper/lower bandwidth equal to p). Assuming, as previously, that said Cholesky factor has been computed beforehand, it can quite easily be shown[3] that the cost of each of the two sparse triangular system solves (lines 12 and 13 of listing B.15) is as indicated.

Listing B.15: Method `computeResiduals` of `DGIGA`, also derived from `Basis`. Called in B.9, once per element.

```
1  function computeResiduals(this,element,physics)
2      element.residuals = element.states*this.controlVandermonde;
           ≤ 2k(p^2 + 1) − 2(k − 1)(κ^2 + 1) FLOPs
3      element.fluxes = physics.flux(element.residuals); J FLOPs (B.12)
4      % Overcomplicated control Vandermonde matrix inversion...
5      element.fluxes = element.fluxes(:,this.colPivotIdsControlVandermonde)/
           this.upperControlVandermonde; J^2 FLOPs
6      element.fluxes(:,this.rowPivotIdsControlVandermonde) = element.fluxes/
           this.lowerControlVandermonde; J^2 FLOPs
7      % ...done.
8      element.residuals = element.fluxes*this.gradientMatrix;
           2k(p + 1)^2 − 2(k − 1)(κ + 1)^2 FLOPs
9      element.residuals(:,[1 end]) = element.residuals(:,[1 end])...
10         - [element.riemannL element.riemannR]; 2 FLOPs
11     % Overcomplicated mass matrix inversion...
12     element.residuals = element.residuals/this.choleskyMassMatrix;
           J(2p + 1) − p(p + 1) FLOPs
13     element.residuals = element.residuals/this.choleskyMassMatrix.';
           J(2p + 1) − p(p + 1) FLOPs
14     % ...done.
15     element.residuals = element.residuals*2/element.dx; J + 1 FLOPs
16 end
```

B.3. Solution update (time scheme)

Once both solution and residuals are available for each element, the former can be advanced one stage towards the next time-step. This is done by the time-scheme, e.g. SSP-RK3(3). From listing B.16, it is clear that each stage update requires 4J FLOPs (on average).

Listing B.16: Method `applyStage` of the `SSP_RK3` class, implementing this particular type of `Solver` (its parent class). It is called in B.1, three times (once per stage) for each element.

```
1  function applyStage(this,element)
2      switch this.stageNow
3          case 1
4              element.extraStates = element.states;
5              element.states = element.states + this.timeDelta*
                   element.residuals; 2J FLOPs
6          case 2
7              element.states = 0.75*element.extraStates + 0.25*(
                   element.states + this.timeDelta*element.residuals); 5J FLOPs
8          otherwise
9              element.states = element.extraStates/3 + (element.states +
                   this.timeDelta*element.residuals)/1.5; 5J FLOPs
10     end
11 end
```

[3] Consider an upper triangular matrix $\boldsymbol{A} \in \mathbb{R}^{\mathrm{J}\times\mathrm{J}}$ of upper bandwidth p, i.e. $a_{ij} = 0$ if $j - i > \mathrm{p} \cup j < i$. We wish to solve $\boldsymbol{Ax} = \boldsymbol{b}$. We start with $x_{\mathrm{J}} = {}^{b_{\mathrm{J}}}/_{a_{\mathrm{JJ}}}$ (1 FLOP). Next, $x_{\mathrm{J}-1} = {}^{(b_{\mathrm{J}-1} - a_{\mathrm{J}-1\mathrm{J}}x_{\mathrm{J}})}/_{a_{\mathrm{J}-1\mathrm{J}-1}}$ (3 FLOPs); then 5, 7, and so on—until reaching $i = \mathrm{J} - \mathrm{p}$. From then on, each row has $\mathrm{p} + 1$ non-zeros; the cost is $1 + 2\mathrm{p}$ FLOPs per remaining row.

Bibliography

[1] Integrated Performance Analysis of Computer Systems. URL http://www.ipacs-benchmark.org.

[2] Milton Abramowitz and Irene Ann Stegun, editors. *Handbook of Mathematical Functions with Figures, Graphs and Tables*. Applied Mathematics. National Bureau of Standards, Washington, D. C., tenth edition, December 1972.

[3] Nikolaus Adams, Xiangyu Hu, and Volker Tritschler. Dispersion-dissipation condition for finite difference schemes. 2015. doi:10.13140/RG.2.1.5181.7122.

[4] Slimane Adjerid, Karen D. Devine, Joseph E. Flaherty, and Lilia Krivodonova. A posteriori error estimation for discontinuous Galerkin solutions of hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1097–1112, January 2002. ISSN 00457825. doi:10.1016/S0045-7825(01)00318-8.

[5] Mohammad Alhawwary and Z. J. Wang. Fourier analysis and evaluation of DG, FD and compact difference methods for conservation laws. *Journal of Computational Physics*, 373:835–862, November 2018. ISSN 00219991. doi:10.1016/j.jcp.2018.07.018.

[6] George B. Arfken and Hans J. Weber. *Mathematical Methods for Physicists*. Elsevier Academic Press, sixth edition, 2005.

[7] Kartikey Asthana and Antony Jameson. High-Order Flux Reconstruction Schemes with Minimal Dispersion and Dissipation. *Journal of Scientific Computing*, 62(3):913–944, March 2015. ISSN 0885-7474, 1573-7691. doi:10.1007/s10915-014-9882-5.

[8] Harold Atkins and Brian Helenbrook. Super-Convergence of Discontinuous Galerkin Method Applied to the Navier-Stokes Equations. In *19th AIAA Computational Fluid Dynamics*, San Antonio, Texas, June 2009. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-137-3. doi:10.2514/6.2009-3787.

[9] Garrett E. Barter and David L. Darmofal. Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation. *Journal of Computational Physics*, 229(5):1810–1827, March 2010. ISSN 00219991. doi:10.1016/j.jcp.2009.11.010.

[10] Timothy Barth and Dennis Jespersen. The design and application of upwind schemes on unstructured meshes. In *27th Aerospace Sciences Meeting*, Reno,NV,U.S.A., January 1989. American Institute of Aeronautics and Astronautics. doi:10.2514/6.1989-366.

[11] Andrea D. Beck, Thomas Bolemann, David Flad, Hannes Frank, Gregor J. Gassner, Florian Hindenlang, and Claus-Dieter Munz. High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations. *International Journal for Numerical Methods in Fluids*, 76(8):522–548, November 2014. ISSN 02712091. doi:10.1002/fld.3943.

[12] Carl M. Bender and Steven A. Orszag. *Advanced Mathematical Methods for Scientists and Engineers I*. Springer New York, New York, NY, 1999. ISBN 978-1-4419-3187-0 978-1-4757-3069-2. doi:10.1007/978-1-4757-3069-2.

[13] Rupak Biswas, Karen D. Devine, and Joseph E. Flaherty. Parallel, adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14(1-3):255–283, April 1994. ISSN 01689274. doi:10.1016/0168-9274(94)90029-9.

[14] Jiri Blazek. *Computational Fluid Dynamics: Principles and Applications*. Butterworth-Heinemann, Amsterdam, 3rd edition, 2015. ISBN 978-0-08-099995-1.

[15] 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, January 1973. ISSN 00219991. doi:10.1016/0021-9991(73)90147-2.

[16] Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer New York, New York, NY, 2008. ISBN 978-0-387-75933-3 978-0-387-75934-0. doi:10.1007/978-0-387-75934-0.

[17] A. Burbeau, P. Sagaut, and Ch.-H. Bruneau. A Problem-Independent Limiter for High-Order Runge–Kutta Discontinuous Galerkin Methods. *Journal of Computational Physics*, 169(1):111–150, May 2001. ISSN 00219991. doi:10.1006/jcph.2001.6718.

[18] Cameron Taylor. Finite Difference Coefficients Calculator. URL `https://web.media.mit.edu/~crtaylor/calculator.html`.

[19] C. Canuto, editor. *Spectral Methods: Fundamentals in Single Domains*. Scientific Computation. Springer-Verlag, Berlin ; New York, 2006. ISBN 978-3-540-30725-9.

[20] Claudio Canuto, M. Yousuff Hussaini, Alfio Maria Quarteroni, and Thomas A. Zang, Jr. *Spectral Methods in Fluid Dynamics*. Springer Science & Business Media, December 2012. ISBN 978-3-642-84108-8.

[21] Jesse Chan and John A. Evans. Multi-patch discontinuous Galerkin isogeometric analysis for wave propagation: Explicit time-stepping and efficient mass matrix inversion. *Computer Methods in Applied Mechanics and Engineering*, 333:22–54, May 2018. ISSN 00457825. doi:10.1016/j.cma.2018.01.022.

[22] 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. ISSN 0025-5718. doi:10.2307/2008474.

[23] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta local projection P1-discontinuous-Galerkin finite element method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, 25(3):337–361, 1991.

[24] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta Discontinuous Galerkin Method for Conservation Laws V. *Journal of Computational Physics*, 141(2):199–224, April 1998. ISSN 00219991. doi:10.1006/jcph.1998.5892.

[25] Bernardo Cockburn and Chi-Wang Shu. Runge–Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems. *Journal of Scientific Computing*, 16(3):173–261, September 2001. ISSN 1573-7691. doi:10.1023/A:1012873910884.

[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, September 1989. ISSN 00219991. doi:10.1016/0021-9991(89)90183-6.

[27] 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, April 1990. ISSN 00255718. doi:10.2307/2008501.

[28] J. Austin Cottrell, Thomas J. R. Hughes, and Yuri Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, August 2009. ISBN 978-0-470-74909-8.

[29] Jie Du, Chi-Wang Shu, and Mengping Zhang. A simple weighted essentially non-oscillatory limiter for the correction procedure via reconstruction (CPR) framework. *Applied Numerical Mathematics*, 95:173–198, September 2015. ISSN 0168-9274. doi:10.1016/j.apnum.2014.01.006.

[30] Michael Dumbser and Raphaël Loubère. A simple robust and accurate a posteriori sub-cell finite volume limiter for the Discontinuous Galerkin method on unstructured meshes. *Journal of Computational Physics*, 319, May 2016. doi:10.1016/j.jcp.2016.05.002.

[31] R. Duvigneau. Isogeometric analysis for compressible flows using a Discontinuous Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 333:443–461, May 2018. ISSN 00457825. doi:10.1016/j.cma.2018.01.039.

[32] Régis. Duvigneau. CAD-consistent adaptive refinement using a NURBS-based discontinuous Galerkin method. *International Journal for Numerical Methods in Fluids*, page fld.4819, February 2020. ISSN 0271-2091, 1097-0363. doi:10.1002/fld.4819.

[33] B. Einfeldt, C. D. Munz, P. L. Roe, and B. Sjögreen. On Godunov-type methods near low densities. *Journal of Computational Physics*, 92(2):273–295, February 1991. ISSN 0021-9991. doi:10.1016/0021-9991(91)90211-3.

[34] Bernd Einfeldt. On Godunov-Type Methods for Gas Dynamics. *SIAM Journal on Numerical Analysis*, 25(2):294–318, 1988. URL http://www.jstor.org/stable/2157317.

[35] European Commission. High-Fidelity LES/DNS Data for Innovative Turbulence Models. URL https://cordis.europa.eu/project/id/814837.

[36] David Flad, Andrea Beck, and Claus-Dieter Munz. Simulation of underresolved turbulent flows by adaptive filtering using the high order discontinuous Galerkin spectral element method. *Journal of Computational Physics*, 313:1–12, May 2016. ISSN 00219991. doi:10.1016/j.jcp.2015.11.064.

[37] C. A.J. Fletcher. The group finite element formulation. *Computer Methods in Applied Mechanics and Engineering*, 37(2):225–244, April 1983. ISSN 00457825. doi:10.1016/0045-7825(83)90122-6.

[38] Gregor Gassner and David A. Kopriva. A Comparison of the Dispersion and Dissipation Errors of Gauss and Gauss–Lobatto Discontinuous Galerkin Spectral Element Methods. *SIAM Journal on Scientific Computing*, 33(5):2560–2579, January 2011. ISSN 1064-8275, 1095-7197. doi:10.1137/100807211.

[39] Gregor J. Gassner and Andrea D. Beck. On the accuracy of high-order discretizations for underresolved turbulence simulations. *Theoretical and Computational Fluid Dynamics*, 27(3-4): 221–237, June 2013. ISSN 0935-4964, 1432-2250. doi:10.1007/s00162-011-0253-7.

[40] Gregor J. Gassner, Andrew R. Winters, Florian J. Hindenlang, and David A. Kopriva. The BR1 Scheme is Stable for the Compressible Navier–Stokes Equations. *Journal of Scientific Computing*, 77(1):154–200, October 2018. ISSN 0885-7474, 1573-7691. doi:10.1007/s10915-018-0702-1. Something is going on with this. There are two papers and one technical report; one paper is an erratum of the first. The technical report is more recent than the papers and does not have exactly the same content.

[41] S. Gottlieb, D. I. Ketcheson, and C.-W. Shu. High order strong stability preserving time discretizations. *Journal of Scientific Computing*, 38(3):251–289, 2009. doi:10.1007/s10915-008-9239-z.

[42] Sigal Gottlieb and Chi-Wang Shu. Total variation diminishing Runge-Kutta schemes. *Mathematics of Computation of the American Mathematical Society*, 67(221):73–85, January 1998. ISSN 0025-5718, 1088-6842. doi:10.1090/S0025-5718-98-00913-2.

[43] 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. ISSN 0021-9991. doi:10.1016/0021-9991(87)90031-3.

[44] R. Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 51(9-10): 1131–1156, July 2006. ISSN 0271-2091, 1097-0363. doi:10.1002/fld.1134.

[45] Ralf Hartmann and Paul Houston. Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations. *Journal of Computational Physics*, 183(2):508–532, December 2002. ISSN 0021-9991. doi:10.1006/jcph.2002.7206.

[46] Ralf Hartmann and Tobias Leicht. Higher order and adaptive DG methods for compressible flows. *VKI LS*, 3:156, 2014.

[47] Jan S. Hesthaven and Tim Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*. Number 54 in Texts in Applied Mathematics. Springer, New York, 2008. ISBN 978-0-387-72065-4 978-0-387-72067-8.

[48] Stefan Hickel, Christian P. Egerer, and Johan Larsson. Subgrid-scale modeling for implicit large eddy simulation of compressible flows and shock-turbulence interaction. *Physics of Fluids*, 26 (10):106101, October 2014. ISSN 1070-6631, 1089-7666. doi:10.1063/1.4898641.

[49] Florian Hindenlang, Gregor J. Gassner, Christoph Altmann, Andrea Beck, Marc Staudenmaier, and Claus-Dieter Munz. Explicit discontinuous Galerkin methods for unsteady problems. *Computers & Fluids*, 61:86–93, May 2012. ISSN 00457930. doi:10.1016/j.compfluid.2012.03.006.

[50] Charles Hirsch. *Numerical Computation of Internal and External Flows: Fundamentals of Computational Fluid Dynamics*. Elsevier/Butterworth-Heinemann, Oxford ; Burlington, MA, 2nd ed edition, 2007. ISBN 978-0-7506-6594-0.

[51] Charles Hirsch, Koen Hillewaert, Ralf Hartmann, Vincent Couaillier, Jean-Francois Boussuge, Frederic Chalot, Sergey Bosniakov, and Werner Haase, editors. *TILDA: Towards Industrial LES/DNS in Aeronautics: Paving the Way for Future Accurate CFD - Results of the H2020 Research Project TILDA, Funded by the European Union, 2015 -2018*. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer International Publishing, 2021. ISBN 978-3-030-62047-9. doi:10.1007/978-3-030-62048-6.

[52] F. Q. Hu, M. Y. Hussaini, and J. L. Manthey. Low-Dissipation and Low-Dispersion Runge–Kutta Schemes for Computational Acoustics. *Journal of Computational Physics*, 124(1):177–191, March 1996. ISSN 00219991. doi:10.1006/jcph.1996.0052.

[53] 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, May 1999. ISSN 0021-9991. doi:10.1006/jcph.1999.6227.

[54] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41):4135–4195, 2005. doi:10.1016/j.cma.2004.10.008.

[55] H. T. Huynh. A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods. In *18th AIAA Computational Fluid Dynamics Conference*, Miami, Florida, June 2007. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-129-8. doi:10.2514/6.2007-4079.

[56] H. T. Huynh, Z. J. Wang, and P. E. Vincent. High-order methods for computational fluid dynamics: A brief review of compact differential formulations on unstructured grids. *Computers & Fluids*, 98:209–220, July 2014. ISSN 00457930. doi:10.1016/j.compfluid.2013.12.007.

[57] Andrzej Jaeschke and Matthias Möller. High-Order Isogeometric Methods for Compressible Flows. I. Scalar Conservation Laws. *arXiv:1809.10896 [math]*, September 2018. URL http://arxiv.org/abs/1809.10896. Comment: Accepted for publication in the Proceedings of the 19th International Conference on Finite Elements in Flow Problems (FEF 2017).

[58] J. Jaffre, C. Johnson, and A. Szepessy. Convergence of the discontinuous galerkin finite element method for hyperbolic conservation laws. *Mathematical Models and Methods in Applied Sciences*, 05(03):367–386, May 1995. ISSN 0218-2025. doi:10.1142/S021820259500022X.

[59] A. Jameson. Analysis and design of numerical schemes for gas dynamics, 1: Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Fluid Dynamics*, 4(3-4):171–218, January 1995. ISSN 1061-8562, 1029-0257. doi:10.1080/10618569508904524.

[60] Antony Jameson. Formulation of kinetic energy preserving conservative schemes for gas dynamics and direct numerical simulation of one-dimensional viscous compressible flow in a shock tube using entropy and kinetic energy preserving schemes. *Journal of Scientific Computing*, 34 (2):188–208, 2008. URL http://www.springerlink.com/index/U256264735254303.pdf.

[61] Antony Jameson. Computational Fluid Dynamics: Past, Present and Future, August 2012. URL https://www.cespr.fsu.edu/people/myh/CFD-Conference/Session-1/Tony-Jameson-Presentation.pdf.

[62] Guang-Shan Jiang and Chi-Wang Shu. Efficient Implementation of Weighted ENO Schemes. *Journal of Computational Physics*, 126(1):202–228, June 1996. ISSN 00219991. doi:10.1006/jcph.1996.0130.

[63] Forrester Johnson, Edward Tinoco, and Jong Yu. Thirty Years of Development and Application of CFD at Boeing Commercial Airplanes, Seattle. In *16th AIAA Computational Fluid Dynamics Conference*, Orlando, Florida, June 2003. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-086-4. doi:10.2514/6.2003-3439.

[64] M. Kermani and E. Plett. Modified entropy correction formula for the Roe scheme. In *39th Aerospace Sciences Meeting and Exhibit*, Reno,NV,U.S.A., January 2001. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2001-83.

[65] Robert M. Kirby and George Em Karniadakis. De-aliasing on non-uniform grids: Algorithms and applications. *Journal of Computational Physics*, 191(1):249–264, October 2003. ISSN 00219991. doi:10.1016/S0021-9991(03)00314-0.

[66] David A. Kopriva. *Implementing Spectral Methods for Partial Differential Equations*. Scientific Computation. Springer Netherlands, Dordrecht, 2009. ISBN 978-90-481-2260-8 978-90-481-2261-5. doi:10.1007/978-90-481-2261-5.

[67] David A. Kopriva and Gregor Gassner. On the Quadrature and Weak Form Choices in Collocation Type Discontinuous Galerkin Spectral Element Methods. *Journal of Scientific Computing*, 44(2): 136–155, August 2010. ISSN 0885-7474, 1573-7691. doi:10.1007/s10915-010-9372-3.

[68] L. Krivodonova, J. Xin, J. F. Remacle, N. Chevaugeon, and J. E. Flaherty. Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Applied Numerical Mathematics*, 48(3):323–338, March 2004. ISSN 0168-9274. doi:10.1016/j.apnum.2003.11.002.

[69] Lilia Krivodonova. Limiters for high-order discontinuous Galerkin methods. *Journal of Computational Physics*, 226(1):879–896, September 2007. ISSN 0021-9991. doi:10.1016/j.jcp.2007.05.011.

[70] Norbert Kroll, editor. *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications: Results of a Collaborative Research Project Funded by the European Union, 2006-2009*. Number v. 113 in Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer, Berlin, 2010. ISBN 978-3-642-03706-1 978-3-642-03707-8.

[71] Norbert Kroll, Charles Hirsch, Francesco Bassi, Craig Johnston, and Koen Hillewaert, editors. *IDIHOM: Industrialization of High-Order Methods - A Top-Down Approach*, volume 128 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*. Springer International Publishing, Cham, 2015. ISBN 978-3-319-12885-6 978-3-319-12886-3. doi:10.1007/978-3-319-12886-3.

[72] Martin Kronbichler. The Discontinuous Galerkin Method: Derivation and Properties. In Martin Kronbichler and Per-Olof Persson, editors, *Efficient High-Order Discretizations for Computational Fluid Dynamics*, volume 602, pages 1–55. Springer International Publishing, Cham, 2021. ISBN 978-3-030-60609-1 978-3-030-60610-7. doi:10.1007/978-3-030-60610-7_1.

[73] Martin Kronbichler and Per-Olof Persson, editors. *Efficient High-Order Discretizations for Computational Fluid Dynamics*, volume 602 of *CISM International Centre for Mechanical Sciences*. Springer International Publishing, Cham, 2021. ISBN 978-3-030-60609-1 978-3-030-60610-7. doi:10.1007/978-3-030-60610-7.

[74] Dmitri Kuzmin. A vertex-based hierarchical slope limiter for p-adaptive discontinuous Galerkin methods. *Journal of Computational and Applied Mathematics*, page 9, 2010.

[75] Dmitri Kuzmin. Algebraic Flux Correction I. In Dmitri Kuzmin, Rainald Löhner, and Stefan Turek, editors, *Flux-Corrected Transport*, pages 145–192. Springer Netherlands, Dordrecht, 2012. ISBN 978-94-007-4037-2 978-94-007-4038-9. doi:10.1007/978-94-007-4038-9_6.

[76] Dmitri Kuzmin and John N. Shadid. Gradient-based nodal limiters for artificial diffusion operators in finite element schemes for transport equations: Gradient-based nodal limiters for artificial diffusion operators in finite element schemes for transport equations. *International Journal for Numerical Methods in Fluids*, 84(11):675–695, August 2017. ISSN 02712091. doi:10.1002/fld.4365.

[77] Dmitri Kuzmin, Matthias Möller, John N. Shadid, and Mikhail Shashkov. Failsafe flux limiting and constrained data projections for equations of gas dynamics. *Journal of Computational Physics*, 229(23):8766–8779, November 2010. ISSN 00219991. doi:10.1016/j.jcp.2010.08.009.

[78] Dmitri Kuzmin, Matthias Möller, and Marcel Gurris. Algebraic Flux Correction II. In Dmitri Kuzmin, Rainald Löhner, and Stefan Turek, editors, *Flux-Corrected Transport*, pages 193–238. Springer Netherlands, Dordrecht, 2012. ISBN 978-94-007-4037-2 978-94-007-4038-9. doi:10.1007/978-94-007-4038-9_7.

[79] Tobias Leicht, Daniel Vollmer, Jens Jägersküpper, Axel Schwöppe, Ralf Hartmann, J. Fiedler, and Tobias Schlauch. DLR-Project Digital-X - Next Generation CFD Solver 'Flucs'. Technical Report 420027, DLR, 2016. URL https://elib.dlr.de/111205/1/420027.pdf.

[80] Robert W. Leland, Mahesh Rajan, and Michael A. Heroux. Performance Efficiency and Effectiveness of Supercomputers. Technical Report SAND2016-3730, 1561471, April 2016.

[81] Sanjiva K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103(1):16–42, November 1992. ISSN 00219991. doi:10.1016/0021-9991(92)90324-R.

[82] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, Cambridge; New York, 2002. ISBN 978-0-511-04219-5 978-0-511-79125-3 978-0-521-81087-6 978-0-511-04507-3 978-0-511-14809-5. URL http://dx.doi.org/10.1017/CBO9780511791253.

[83] Doron Levy, Gabriella Puppo, and Giovanni Russo. On the behavior of the total variation in CWENO methods for conservation laws. *Applied Numerical Mathematics*, 33(1-4):407–414, May 2000. ISSN 01689274. doi:10.1016/S0168-9274(99)00107-5.

[84] Wanai Li, Qian Wang, and Yu-Xin Ren. A p-weighted limiter for the discontinuous Galerkin method on one-dimensional and two-dimensional triangular grids. *Journal of Computational Physics*, 407:109246, April 2020. ISSN 00219991. doi:10.1016/j.jcp.2020.109246.

[85] James Lighthill. *Waves in Fluids*. Cambridge University Press, 1990. ISBN 0-521-29233-6.

[86] J. Loffeld and J. A.F. Hittinger. On the arithmetic intensity of high-order finite-volume discretizations for hyperbolic systems of conservation laws. *The International Journal of High Performance Computing Applications*, 33(1):25–52, January 2019. ISSN 1094-3420, 1741-2846. doi:10.1177/1094342017691876.

[87] Hong Luo, Joseph D. Baum, and Rainald Löhner. A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids. *Journal of Computational Physics*, 225(1):686–713, July 2007. ISSN 00219991. doi:10.1016/j.jcp.2006.12.017.

[88] Johannes Markert, Gregor Gassner, and Stefanie Walch. A Sub-Element Adaptive Shock Capturing Approach for Discontinuous Galerkin Methods. *arXiv:2011.03338 [astro-ph]*, November 2020. URL http://arxiv.org/abs/2011.03338. Comment: 40 pages in total: 30 pages of main text and 7 pages of appendix.

[89] Gianmarco Mengaldo, Daniele De Grazia, Freddie Witherden, Antony Farrington, Peter Vincent, Spencer Sherwin, and Joaquim Peiro. A Guide to the Implementation of Boundary Conditions in Compact High-Order Methods for Compressible Aerodynamics. In *7th AIAA Theoretical Fluid Mechanics Conference*, Atlanta, GA, June 2014. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-293-6. doi:10.2514/6.2014-2923.

[90] Matthias Möller and Andrzej Jaeschke. High-Order Isogeometric Methods for Compressible Flows. II. Compressible Euler Equations. *arXiv:1809.10893 [math]*, September 2018. URL http://arxiv.org/abs/1809.10893.

[91] Euripides Montagne and Edward Aymeric. Let's Agree on Computing Flops for the Symmetric Sparse Matrix Vector Product. In *24th High Performance Computing Symposium*, Pasadena, CA, 2016. Society for Modeling and Simulation International (SCS). ISBN 978-1-5108-2318-1. doi:10.22360/SpringSim.2016.HPC.037.

[92] R. C. Moura, S. J. Sherwin, and J. Peiró. Linear dispersion–diffusion analysis and its application to under-resolved turbulence simulations using discontinuous Galerkin spectral/hp methods. *Journal of Computational Physics*, 298:695–710, October 2015. ISSN 00219991. doi:10.1016/j.jcp.2015.06.020.

[93] R. C. Moura, G. Mengaldo, J. Peiró, and S. J. Sherwin. On the eddy-resolving capability of high-order discontinuous Galerkin approaches to implicit LES / under-resolved DNS of Euler turbulence. *Journal of Computational Physics*, 330:615–623, February 2017. ISSN 00219991. doi:10.1016/j.jcp.2016.10.056.

[94] Scott M. Murman, Nicholas K. Burgess, Laslo T. Diosady, and Anirban Garai. A DGSEM Shock-capturing Scheme for Scale-resolving Simulations. In *23rd AIAA Computational Fluid Dynamics Conference*, AIAA AVIATION Forum. American Institute of Aeronautics and Astronautics, June 2017. doi:10.2514/6.2017-4106.

[95] Jens Niegemann, Richard Diehl, and Kurt Busch. Efficient low-storage Runge–Kutta schemes with optimized stability regions. *Journal of Computational Physics*, 231(2):364–372, January 2012. ISSN 0021-9991. doi:10.1016/j.jcp.2011.09.003.

[96] Per-Olof Persson and Jaime Peraire. Sub-Cell Shock Capturing for Discontinuous Galerkin Methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2006. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-039-0. doi:10.2514/6.2006-112.

[97] Les A. Piegl and Wayne Tiller. *The NURBS Book*. Monographs in Visual Communications. Springer, Berlin ; New York, 2nd ed edition, 1997. ISBN 978-3-540-61545-3.

[98] Les A. Piegl, Wayne Tiller, and Khairan Rajab. It is time to drop the "R" from NURBS. *Engineering with Computers*, 30(4):703–714, October 2014. ISSN 0177-0667, 1435-5663. doi:10.1007/s00366-013-0318-x.

[99] Sergio Pirozzoli. Numerical Methods for High-Speed Flows. *Annual Review of Fluid Mechanics*, 43(1):163–194, January 2011. ISSN 0066-4189, 1545-4479. doi:10.1146/annurev-fluid-122109-160718.

[100] Stephen B. Pope. The computational cost. In *Turbulent Flows*, pages 346–350. Cambridge University Press, Cambridge, 2017. ISBN 978-0-521-59886-6.

[101] 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, January 2004. ISSN 00219991. doi:10.1016/j.jcp.2003.07.026.

[102] 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, January 2005. ISSN 1064-8275, 1095-7197. doi:10.1137/04061372X.

[103] 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, July 2005. ISSN 00457930. doi:10.1016/j.compfluid.2004.05.005.

[104] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics*. Number 37 in Texts in Applied Mathematics. Springer, Berlin ; New York, 2nd ed edition, 2007. ISBN 978-3-540-34658-6.

[105] P. L. Roe. Computational fluid dynamics—retrospective and prospective. *International Journal of Computational Fluid Dynamics*, 19(8):581–594, November 2005. ISSN 1061-8562. doi:10.1080/10618560600585315.

[106] L. F. Shampine. Vectorized adaptive quadrature in MATLAB. *Journal of Computational and Applied Mathematics*, 211(2):131–140, February 2008. ISSN 03770427. doi:10.1016/j.cam.2006.11.021.

[107] Cengke Shi and Chi-Wang Shu. On local conservation of numerical methods for conservation laws. *Computers & Fluids*, 169:3–9, June 2018. ISSN 00457930. doi:10.1016/j.compfluid.2017.06.018.

[108] Chi-Wang Shu. TVB Uniformly High-Order Schemes for Conservation Laws. *Mathematics of Computation*, 49(179):105–121, July 1987.

[109] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, August 1988. ISSN 00219991. doi:10.1016/0021-9991(88)90177-5.

[110] Chi-Wang Shu and Stanley Osher. Efficient Implementation of Essentially Non-oscillatory Shock-Capturing Schemes, II. page 47, 1989.

[111] Jeffrey Slotnick, Abdollah Khodadoust, Juan Alonso, David Darmofal, William Gropp, Elizabeth Lurie, and Dimitri Mavriplis. CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. Contractor Report (CR) NASA/CR–2014-218178, NASA, March 2014. URL https://ntrs.nasa.gov/citations/20140003093.

[112] Joel Smoller. *Shock Waves and Reaction-Diffusion Equations*. Number 258 in Grundlehren Der Mathematischen Wissenschaften. Springer-Verlag, New York, 2nd ed edition, 1994. ISBN 978-0-387-94259-9 978-3-540-94259-7.

[113] G. A. Sod. Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *J. Comput. Phys.; (United States)*, 26:4, April 1978. doi:10.1016/0021-9991(78)90023-2.

[114] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer, Dordrecht ; New York, 3rd ed edition, 2009. ISBN 978-3-540-25202-3 978-3-540-49834-6.

[115] Lloyd N. Trefethen. Group Velocity in Finite Difference Schemes. *SIAM Review*, 24(2):113–136, April 1982. ISSN 0036-1445, 1095-7200. doi:10.1137/1024038.

[116] Kris Van den Abeele. *Development of High-Order Accurate Schemes for Unstructured Grids*. PhD thesis, Vrije Universiteit Brussel, Brussels, 2009.

[117] Julien Vanharen, Guillaume Puigt, Xavier Vasseur, Jean-François Boussuge, and Pierre Sagaut. Revisiting the spectral analysis for high-order spectral discontinuous methods. *Journal of Computational Physics*, 337:379–402, May 2017. ISSN 00219991. doi:10.1016/j.jcp.2017.02.043.

[118] John Vassberg. Expectations for computational fluid dynamics. *International Journal of Computational Fluid Dynamics*, 19(8):549–558, November 2005. ISSN 1061-8562. doi:10.1080/10618560500508375.

[119] Robert Vichnevetsky and John B. Bowles. *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*. Society for Industrial and Applied Mathematics, January 1982. ISBN 978-0-89871-181-3 978-1-61197-087-6. doi:10.1137/1.9781611970876.

[120] P. E. Vincent, P. Castonguay, and A. Jameson. Insights from von Neumann analysis of high-order flux reconstruction schemes. *Journal of Computational Physics*, 230(22):8134–8154, September 2011. ISSN 00219991. doi:10.1016/j.jcp.2011.07.013.

[121] P. E. Vincent, P. Castonguay, and A. Jameson. A New Class of High-Order Energy Stable Flux Reconstruction Schemes. *Journal of Scientific Computing*, 47(1):50–72, April 2011. ISSN 0885-7474, 1573-7691. doi:10.1007/s10915-010-9420-z.

[122] P. E. Vincent, A. M. Farrington, F. D. Witherden, and A. Jameson. An extended range of stable-symmetric-conservative Flux Reconstruction correction functions. *Computer Methods in Applied Mechanics and Engineering*, 296:248–272, November 2015. ISSN 00457825. doi:10.1016/j.cma.2015.07.023.

[123] J. B. Vos, A. Rizzi, D. Darracq, and E. H. Hirschel. Navier–Stokes solvers in European aircraft design. *Progress in Aerospace Sciences*, 38(8):601–697, November 2002. ISSN 0376-0421. doi:10.1016/S0376-0421(02)00050-7.

[124] Michael Yang and Z. J. Wang. A Parameter-Free Generalized Moment Limiter for High-Order Methods on Unstructured Grids. *Advances in Applied Mathematics and Mechanics*, 1(4):451–480, June 2009. ISSN 2070-0733, 2075-1354. doi:10.4208/aamm.09-m0913.

[125] Z. J. Wang, editor. *Adaptive High-Order Methods in Computational Fluid Dynamics*. Number v. 2 in Advances in Computational Fluid Dynamics. World Scientific, Singapore ; Hackensack, N.J., 2011. ISBN 978-981-4313-18-6.

[126] Z. J. Wang. High-order computational fluid dynamics tools for aircraft design. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 372 (2022):20130318, August 2014. doi:10.1098/rsta.2013.0318.

[127] Z. J. Wang and Haiyang Gao. A unifying lifting collocation penalty formulation including the discontinuous Galerkin, spectral volume/difference methods for conservation laws on mixed grids. *Journal of Computational Physics*, 228(21):8161–8186, November 2009. ISSN 00219991. doi:10.1016/j.jcp.2009.07.036.

[128] Z. J. Wang and H. T. Huynh. A review of flux reconstruction or correction procedure via reconstruction method for the Navier-Stokes equations. *Mechanical Engineering Reviews*, 3(1): 15–00475–15–00475, 2016. ISSN 2187-9753. doi:10.1299/mer.15-00475.

[129] Z. J. Wang, Krzysztof Fidkowski, Rémi Abgrall, Francesco Bassi, Doru Caraeni, Andrew Cary, Herman Deconinck, Ralf Hartmann, Koen Hillewaert, H. T. Huynh, Norbert Kroll, Georg May, Per-Olof Persson, Bram van Leer, and Miguel Visbal. High-order CFD methods: Current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013. ISSN 1097-0363. doi:10.1002/fld.3767.

[130] Samuel Williams, Andrew Waterman, and David Patterson. Roofline: An insightful visual performance model for multicore architectures. *Communications of the ACM*, 52(4):65–76, April 2009. ISSN 0001-0782, 1557-7317. doi:10.1145/1498765.1498785.

[131] Paul Woodward and Phillip Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54(1):115–173, April 1984. ISSN 0021-9991. doi:10.1016/0021-9991(84)90142-6.

[132] 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, November 2010. ISSN 00219991. doi:10.1016/j.jcp.2010.08.016.

[133] 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 A: Mathematical, Physical and Engineering Sciences*, 467(2134):2752–2776, October 2011. ISSN 1364-5021, 1471-2946. doi:10.1098/rspa.2011.0153.

[134] 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, January 2013. ISSN 00219991. doi:10.1016/j.jcp.2012.08.028.

[135] Hongqiang Zhu, Yue Cheng, and Jianxian Qiu. A Comparison of the Performance of Limiters for Runge-Kutta Discontinuous Galerkin Methods. *Advances in Applied Mathematics and Mechanics*, 5, June 2013. doi:10.4208/aamm.2012.m22.

[136] Jun Zhu and Jianxian Qiu. Hermite WENO Schemes and Their Application as Limiters for Runge-Kutta Discontinuous Galerkin Method, III: Unstructured Meshes. *Journal of Scientific Computing*, 39(2):293–321, May 2009. ISSN 0885-7474, 1573-7691. doi:10.1007/s10915-009-9271-7.

[137] Jun Zhu, Xinghui Zhong, Chi-Wang Shu, and Jianxian Qiu. Runge-Kutta Discontinuous Galerkin Method with a Simple and Compact Hermite WENO Limiter. *Communications in Computational Physics*, 19(4):944–969, April 2016. ISSN 1815-2406, 1991-7120. doi:10.4208/cicp.070215.200715a.