MULTISCALE METHODS
IN
SCIENCE AND ENGINEERING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Guglielmo Scovazzi

August 2004

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Thomas J.R. Hughes
(Principal Adviser)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Peter M. Pinsky
(Co-Adviser)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Eric F. Darve
(Reader)

Approved for the University Committee on Graduate Studies.

Abstract

In recent years multiscale approaches have had a tremendous impact on the analysis of complex problems in Science and Engineering. The multiscale concept originally proposed by T.J.R. Hughes and co-workers is adopted and extended to new applications. The key idea of the multiscale paradigm is to decompose the solution into coarse scales (directly simulated) and fine scales (modeled or omitted in the numerical simulations). The effect of the fine scales on the coarse scales is exploited by means of approximate, local Green's function problems, which ultimately yield a subgrid-scale operator in the coarse-scale equations, in the form of a subgrid model and/or stabilization term.

This thesis is organized in three parts: Lagrangian hydrodynamics; multiscale Discontinuous Galerkin methods for advection-diffusion problems; and large eddy simulation of turbulence. More specifically, in Part I a new multiscale streamline-upwinded Petrov-Galerkin (SUPG) method is developed for Lagrangian hydrodynamic flow simulations. The multiscale interpretation of SUPG methods is also exploited to design a new artificial viscosity operator, in order to obtain a stable discontinuity capturing scheme. In Part II, the multiscale view is used to develop a new Discontinuous Galerkin finite element method for the advection-diffusion equation, in which the continuous and discontinuous representations of the solution are linked via local multiscale problems. Finally, in Part III, the multiscale analysis is applied to turbulent flow simulations, and a proposed new class of large eddy simulation models is tested with good results in transitional turbulent flows.

The work presented illustrates that the multiscale framework can provide a unifying view within which many apparently different phenomena can be analyzed.

Acknowledgements

The research documented in this dissertation has been supported by Sandia National Laboratories Contract No. A0340.0, and ONR Contract No. 00014-03-1-0263, gratefully acknowledged.

Many thanks to Professor Eric Darve and Professor Peter Pinsky, members of my Reading Committee, for insightful discussions and "procedural" help during my leave of absence from Stanford: their kindness is deeply appreciated.

I would like to thank Dr. Annalisa Buffa (CNR Pavia, Italy), whose cooperation on the work documented in Part II of the dissertation is much appreciated, especially in connection to aspects of numerical and functional analysis.

Special thanks to Dr. Pavel Bochev, Dr. Mark Christon, and Dr. John Shadid at Sandia National Laboratories, Albuquerque, for very insightful discussions and intense collaboration: I have had a wonderful experience working with them.

I would like to express my deepest appreciation to Professor Massimo Germano (Politecnico di Torino, Italy), for constant, kind ecouragement, as well as many enlightening conversations on turbulence and Physics.

I will be always indebted to Prof. Thomas J.R. Hughes for the knowledge and vision he instilled in me. Having him as an adviser and mentor has been a rare privilege.

Last and most important, I would like to express my deepest thanks to my family. It has helped me keep my balance with its unconditional support, in the good and bad days of my life. This thesis is dedicated to the memory of my grandmother Giuseppina, who helped my mother and father raise me since I was born and passed away on my first year at Stanford: I truly wish she could enjoy with me the achievement of many years of intense work.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past thirty years the role of multiscale approaches in the analysis of complex problems in Science and Engineering has grown in importance considerably.

The multiscale concept originally proposed by Hughes et al. [34] is adopted and extended to new applications. A decomposition of the solution of a system of equations into coarse scales (directly simulated) and fine scales (modeled or omitted in the numerical simulations) is the key idea of the multiscale paradigm. The effect of the fine scales on the coarse-scale dynamics is accounted for through local Green's function problems, ultimately responsible for the generation of a subgrid model and/or stabilization term in the equations for the coarse scales.

This thesis is organized in three parts: Lagrangian hydrodynamics; multiscale discontinuous Galerkin methods for advection-diffusion problems; and large eddy simulation of turbulence. A detailed introduction to each specific part follows.

The work presented illustrates that the multiscale framework can provide a unifying analysis and modeling tool for complex dynamical systems, so that each part of the present dissertation can be intended as a particular manifestation of the multiscale paradigm.

Part I: HYDRO-SUPG for Shock Hydrodynamics

In Part I of this dissertation thesis, a new, multiscale, Streamline Upwinded Petrov-Galerkin (SUPG) method for the simulation of Lagrangian hydrodynamic flows will be

presented. In the literature, algorithms for Lagrangian hydrodynamics are termed *hydrocodes*, which suggests the name HYDRO-SUPG for the proposed method. The typical field of application of such algorithms is time-accurate, transient analysis of mechanical systems undergoing high-rate, large deformations.

In *hydrodynamic flows*, fields of kinematic and thermodynamic variables are expressed in a Lagrangian reference frame, and characterize a medium with no shear strength, so that only pressure (normal) stresses can be applied to the material. In this regard, any mechanical system for which the pressure stresses are dominant by orders of magnitude with respect to shear stresses can be considered a hydrodynamic flow. In particular, the terminology "hydrodynamic" does not refer to an implied incompressibility constraint, and most of the hydrocode simulations are compressible flow computations. In more recent years, the definition of Lagrangian hydrodynamics has been weakened to include media with shear strength and Arbitrary Eulerian Lagrangian formulations in some instances, but throughout the following chapters, the discussion will be limited to the strict definition.

State of the art hydrocodes are based upon a continuous, piecewise-linear, node-centered approximation of the kinematic variables (displacements and velocities), while they hinge upon a discontinuous, piecewise-constant, cell-centered discretization for the thermodynamic variables (such as density, internal energy, pressure, etc.). Such choice of function spaces is clearly motivated by the observation that a combination of piecewise linear kinematic variables and staggered, piecewise constant thermodynamic variables leads to overall second-order accuracy in space. Furthermore, the piecewise-constant interpolation of thermodynamic quantities entails conservation properties typical of finite volume formulations, and finally, second-order temporal accuracy is obtained by staggering in time the velocities with respect to all other variables. Because this formulation is intrinsically unstable in the presence of shock discontinuities, a stabilizing artificial viscosity was first introduced by Von Neumann and Richtmyer [66], proportional to the square of the mesh spacing and the velocity gradients. The original idea of Von Neumann and Richtmyer was to smooth the solution in the presence of shocks, allowing coarse meshes to properly capture the *entropy solution* (i.e., the solution satisfying conservation laws *and* the entropy condition), while maintaining *accuracy* (by scaling the viscosity

with the square of the mesh spacing). This very ingenious concept was the key factor in the broad success of hydrocode technology, together with ease of implementation in multidimensional computations. Hydrocodes are still very competitive with respect to the most advanced compressible flow solvers, since the latter present problematic issues in their generalization to multiple dimensions.

Hydrocodes were born during the years of the Manhattan Project at Los Alamos [66], but they soon found application in the most diverse fields, from *blast* computations to *crash* and *contact dynamics* simulations (widely used in the car and construction industry as well as in earthquake engineering), from *semiconductor devices design* to *Astrophysics*. In the following discussion, the focus will be on engineering applications.

Due to the limited availability of computational resources, the initial implementations of the late 40's and 50's, made extensive use of symmetries (planar, cylindrical, or spherical) in order to reduce a multidimensional problem to a one-dimensional problem. As computers grew more and more powerful, the process of extending hydrocodes to complex, fully multi-dimensional geometries, witnessed the rise of a number of problematic issues.

A first problem was due to the detection of so-called *acoustic instabilities* in the flows, particularly evident in multidimensional calculations, although already present in the one-dimensional case. Such instabilities appear typically in the form of a tail of node-to-node oscillations past shocks. Kuropatenko [47] showed that a Godunov-type Riemann solver for Lagrangian hydrodynamics entails the generation of an artificial viscosity term scaling as the Von Neumann-Richtmyer viscosity in the limit for strong shocks and as the product of the local speed of sound and the mesh spacing in the limit for weak shocks: we will refer to the latter as the "Kuropatenko scaling". The modification of the original idea of Von Neumann-Richtmyer using the Kuropatenko scaling produces a correction only first-order accurate, but since it is applied only to the shock region, the overall accuracy is good.

Further developments in the field of artificial viscosities have led to the incorporation of *Total Variation Diminishing* (TVD), and *Total Variation Bounded* (TVB) approaches into the original viscosities. In these approaches, the strategy is to use a low- (first-) order viscosity which ensures robustness under all practical conditions and build a higher-order

viscosity by applying *limiting* to the Von Neumann-Richtmyer scaling of the viscosity.

A second class of issues is related to manifestations of the Ladizenskaya-Brezzi-Babŭska (LBB) condition, in term of so-called *hourglass instabilities* and *element locking*.

Hourglass modes are node-to-node oscillations that affect simulations on quadrilaterals and hexahedra, due to zero-energy modes that cannot be "sensed" by the degrees of freedom of the elements whenever low order quadratures are applied. In production hydrocodes, these modes are filtered out or *ad hoc* hourglass control is applied.

Locking is a phenomenon arising whenever the material is subject to an incompressibility constraint, either due to a limit behavior (as in plasticity) or the intrinsic constitutive laws of certain materials (e.g., rubber). If triangular or tetrahedral elements are used, then the displacements permitted by the degrees-of-freedom of the element may not be sufficient to ensure the correct kinematics. As a consequence, the element *locks*, that is, does not deform according to the physical displacement field but allows only rigid translation (which can reduce to no motion if one edge of the element is on a boundary where zero displacement conditions are enforced). Therefore, the locking phenomenon is usually associated with the intrinsically higher stiffness that triangular/tetrahedral elements have with respect to quadrilateral/hexahedral elements of equal order.

Occurrence of locking and intense acoustic instabilities for triangular elements are the main reasons why most of production hydrocodes are based on quadrilateral and hexahedral elements. However, when the computations involve very complex geometries, the tool of automatic grid generation on tetrahedral meshes can represent a significant advantage in the overall analysis and design process.

In the present dissertation, a completely new approach is proposed, based on the multiscale paradigm of Hughes et al. [34], and its potential is demonstrated with numerical tests.

Returning for a moment to the issue of acoustic instabilities, a multiscale analysis as in [34] – not reported here due to lengthy, although not conceptually complex, computations – reveals that the locally-linearized sub-grid scale problem for the Lagrangian hydrodynamic systems is governed by a wave propagation Green's function kernel.

In practice, the multiscale paradigm is based upon a splitting of the trial- and test-function spaces into subspaces of coarse (i.e., numerically resolved) scales and fine (i.e.,

sub-grid) scales. The subspaces are assumed linearly independent, so that the original weak formulation of the hydrodynamic equations can be decomposed into a coarse (resolved) scale problem and a sub-grid (unresolved) scale problem. For nonlinear systems, the multiscale analysis involves a local (i.e., element-by-element) linearization of the subgrid-scale problem. Subsequent to linearization, a formal solution for the subgrid-scale component is found in terms of the residual of the coarse-scale solution. This technique can be thought of as a local linearization followed by the application of an infinite-dimensional Shur's complement approach.

The resolvent operator for the sub-grid problem is obviously an integral operator, expressed in terms of a kernel which is the small scale Green's function operator. In the particular case of Lagrangian hydrodynamics such kernel is given by the Green's function for the wave equation. Now, SUPG stabilization terms in the coarse scale equations can be seen as terms arising from an approximation of the sub-grid resolvent operator.

Therefore, not only the multiscale framework reveals that the Kuropatenko analysis is correct and that instability of acoustic nature can arise in hydrodynamic flows, but also provides a way to avoid such instabilities, since all variants of SUPG stabilization terms can be interpreted as approximations to the resolvent operator for the sub-grid problem.

The great advantage of an SUPG stabilization is that the it can be proved to provide optimal error estimates, so that stability does not compromise accuracy.

Additionally, it is well-known from the extensive literature on the subject that SUPG stabilized methods *do not suffer* from hourglassing or locking (for this second issue, refer to the work of Miniatty and co-workers [45, 50–52]), so that the potential for new generation hydrocodes on unstructured tetrahedral or hexahedral meshes appears clear.

A simple SUPG method is however not sufficient in the stabilization of strong shocks: for this purpose, a new multiscale viscosity will be presented. The key feature that distinguishes the proposed viscosity from all the viscosities previously adopted in the hydrocode literature, is the fact that it is residual-based: this will ensure the viscosity not to perturb the accuracy of the method in the case of smooth flows.

The rest of the exposition is organized as follows: in chapter 2 the basic equations of Lagrangian hydrodynamics are introduced, in both integral and differential form. A

review of previous work on artificial viscosities is outlined in chapter 3. The variational formulation is established in chapter 4, while chapter 5 is devoted to the design of the stabilization terms. The analysis of a novel discontinuity capturing operator, based on multiscale concepts, is presented in chapter 6. Chapter 7 summarizes a predictor/multi-corrector algorithm used for time integration, and, finally, numerical results are presented in chapter 8.

Part II: A Multiscale Link between CG/DG Worlds

The discontinuous Galerkin (DG) method was developed for problems of neutron transport approximately 30 years ago by Reed and Hill [61]. Early works of note include the paper of Lesaint and Raviart [48] on the same subject, and the paper of Johnson, Navert and Pitkaranta [44] who, in the context of advection dominated processes, synthesized it with stabilized methods and performed a complete mathematical analysis. The interest in DG developed very slowly but has accelerated significantly in recent years. The compendium of papers in [11] provides valuable summaries of the current state-of-the-art and introductions to the literature.

The DG method is felt to have advantages of robustness over the classical continuous Galerkin (CG) method, especially for first-order differential operators associated with hyperbolic equations, and better conditioning of resultant linear equation systems leading to improved iterative performance. There is also an opportunity to link DG with the numerical fluxes (i.e., solutions of the one-dimensional Riemann problem) used in finite volume methods and develop higher-order accurate procedures for wave-propagation. These attributes have led to numerous applications in fluids where the CG method has often proved inadequate. There has also been recent interest in applying DG to elliptic problems so that advective-diffusive phenomena can be modeled. For a review of work in this area, see Arnold *et al.* [1]. Recent studies include Brezzi et al. [4] , Dawson[6], and Hughes, Masud and Wan [36]. There has been very little work in structural mechanics so far but interest is beginning to grow. See for example, Engel et al.[13] and Brezzi et al. [5].

Despite the increased interest in DG methods, there are shortcomings that limit their

practical utility. Foremost among these is the size of the DG equation system for interpolations linear and higher. By virtue of the fact the trial functions are completely discontinuous, there is no sharing of degrees-of-freedom at element boundaries. Consequently, the size of the solution space "explodes". For example, assuming about seven linear tetrahedral elements per node, the DG system involves approximately 28 times the number of unknowns of the corresponding CG system (see Hughes et al. [33]). Storage and solution cost are, obviously, adversely affected by this issue, which seems the main reason for the small commercial impact the DG method has had so far. In addition, it has been observed that the robustness of DG is somewhat exaggerated. Simple, one-dimensional examples of pure advection and pure diffusion were shown to give rise to spurious oscillations in Hughes et al. [33].

The proposed new approach begins with a typical continuous finite element space. By releasing all continuity requirements, we associate to it a completely discontinuous space. Then we apply standard, global DG formulations to the discontinuous space. The unique feature of our formulation is the use of local, element-wise problems, employing the same DG method, to define the discontinuous field in terms of the degrees-of-freedom of the continuous field. The local problems employ weakly imposed boundary conditions and the solutions are still discontinuous but they are parameterized by the degrees-of-freedom of the much smaller continuous space. The global problem has the equation size and structure of a CG method but it is indeed a DG method. The local problems serve to project the solution into a reduced dimensional manifold that expresses the partial differential structure of the problem considered. This aspect is seen to be related to methods used in wave propagation problems, relying on numerical fluxes inspired by local Riemann solutions but here the local problems are solved numerically using simple basis functions. The interesting result is that the new method gives more accurate and robust results than the global DG method, in particular, eliminating the deficiencies noted by Hughes et al. [33] previously, and, at the same time, the storage and computational effort are significantly reduced. As may be obvious from the description, the method has a multiscale interpretation.

The new method is demonstrated on simple test cases of advection-diffusion. Convergence studies for this case are also presented. However, the ideas are quite general

and may be applied to arbitrary partial differential equation systems. Chapter 9 is devoted to the introduction of the advection-diffusion problem, to prepare the ground for the global DG formulation presented in chapter 10. Three variants of the discretization of the Laplace operator will be analyzed: the *symmetric*, *neutral*, and *skew-symmetric* form. The numerical results of chapter 12 will show that the symmetric form leads to superior results in both the diffusive and convective limit.

Part III: Multiscale Methods in Turbulence

Variational multiscale concepts for Large Eddy Simulation (LES) were first introduced in [28]. The basic idea was to use variational projections in place of the traditional filtered equations approach and to focus modeling on fine-scale equations rather than coarse-scale equations. Avoidance of filters eliminates many difficulties associated with the traditional approach, namely, inhomogeneous non-commutative filters necessary for wall-bounded flows, use of complex filtered quantities in compressible flows, problem, etc.

In addition, modeling confined to the fine-scale equations retains numerical consistency in the coarse-scale equations and thus permits full rate-of-convergence of the underlying numerical method in contrast with the usual approach which limits convergence rate due to artificial viscosity effects in the fully resolved scales ($O(h^{4/3})$ in the case of Smagorinsky-type models).

Initial versions of the variational multiscale method focused on dividing resolved scales into coarse and fine designations and including eddy viscosities, inspired by traditional models, only in the fine scale equations and acting *only* on the fine scales. This version was studied in [29, 30, 54] and found to work very well on homogeneous isotropic flows and fully-developed equilibrium and non-equilibrium turbulent channel flows. Static eddy viscosity models were employed in these studies but superior results were subsequently obtained through the use of dynamic models, as reported in [26] and [31]. Good numerical results were obtained with this approach by other of investigators [12, 43, 59, 60, 68]. Particular mention should be made of the work of Farhat and Koobus [15], and Koobus and Farhat [46], who have implemented this procedure in an unstructured mesh, finite volume, compressible flow code, and applied it very successfully to a number of complex industrial flows.

This initial version of the variational multiscale concept appears to have already demonstrated its viability and practical utility and is already arguably superior to traditional LES turbulence modeling approaches. Nevertheless, there is still significant room for improvement. The use of traditional eddy viscosities to represent fine-scale dissipation under all conditions may be an inefficient mechanism in some instances. Employing an eddy viscosity in the resolved fine scales to represent turbulent dissipation introduces a consistency error which results in the resolved fine scales being "sacrificed" to retain full consistency in the coarse scales. This procedure is felt to be "inefficient" because approximately 7/8 of the resolved scales are typically ascribed to the fine scales.

Another shortcoming noted for the initial version of the variational multiscale method is too small an energy transfer to unresolved modes when the discretization is very coarse (see, e.g., [31]). This phenomenon is also occasionally noted for some traditional models, such as the dynamic Smagorinsky model on coarse meshes [31] but seems to be somewhat more pronounced for the multiscale version of the dynamic model.

The objectives of recent multiscale work have been to capture all scales consistency with a technique that does not rely on an eddy viscosity. The following chapters will describe a new variational multiscale formulation which makes considerable progress toward these goals. The basic ingredients of the new technique are the multiscale decomposition of the solution and an asymptotic expansion applied to the local subgrid-scale problems. Initiatory numerical results demonstrate that, under certain conditions, the technique holds the promise of much more accurate and efficient LES procedures, in which the role of eddy viscosities will be reduced.

The proposed method bridges the gap between turbulence modeling and Green's function techniques typical of SUPG stabilization, as will be clear in the following discussion. These newer variational multiscale ideas, and the older variants, were implemented in a finite volume code that has enjoyed widespread use in turbulence simulations (see [55]). Following along the lines of the DNS investigations of Durbin and Jacobs [40, 41], bypass transition of a boundary layer is examined from the point of view of the variational multiscale and classical LES.

The organization of the exposition is as follows. Chapter 13 presents the general concepts of the multiscale approach, with emphasis on the Navier-Stokes equations. Chapter 14 represents an instructive example on how the concept is applied to the Burgers equation, a simplified model that can highlight how the dynamics of the small scales is characterized by the asymptotic expansion concept. Finally, chapter 15 is devoted to the application of the new concept to large eddy simulation studies of bypass transition to turbulence.

Part I

HYDRO-SUPG: A Multiscale Stabilized Framework for Lagrangian Hydrodynamics

Chapter 2

Equations for Lagrangian Hydrodynamics

In this chapter, conservation laws in Lagrangian coordinates are derived. Although going through the detailed derivation may seem pedantic to the expert researcher, this step is considered necessary. In fact, standard hydrocode formulations use simplifications of the basic equations, which hinge upon the assumption of piecewise-constant fields in space/time for the thermodynamic variables. Due to the fact that in the newly proposed formulation the thermodynamic variables are approximated by continuous piecewise-linear functions, the usual simplifications no longer apply. Unless the correct conservative form of the equations is adopted, one cannot expect conservation properties to hold true.

2.1 General conservation laws

Let us define the transformation φ from a reference configuration $\boldsymbol{X}$ (in our case the initial configuration state) to another configuration state $\boldsymbol{x}(\boldsymbol{X}, t)$, the so-called *current configuration*, (in our case the Eulerian coordinate frame),

$$\varphi : V \ (open) \ \in \mathbb{R}^{n_\mathrm{d}} \ \rightarrow \ \mathbb{R}^{n_\mathrm{d}} \tag{2.1}$$

$$\boldsymbol{X} \ \mapsto \ \boldsymbol{x} = \varphi(\boldsymbol{X}, t) \tag{2.2}$$

where n_{d} is the number of spatial dimensions. The *deformation gradient* is defined as $\boldsymbol{F} = \mathbf{Grad}\ \varphi$, that is $F_{iA} = \dfrac{\partial x_i}{\partial X_A}, \quad i, A \in \{1, \ldots, n_{\mathrm{d}}\}$, and $J = \det \boldsymbol{F}$ is the *Jacobian determinant* of the transformation. (The Einstein summation convention for repeated indices is used throughout.)

Starting in a very abstract framework, we can write a general master balance law:

$$\frac{d}{dt}\left(\int_{\mathcal{V}} \rho\ \mathcal{A}\ d\mathcal{V}\right) = \int_{\mathcal{V}} \rho\ \mathcal{B}\ d\mathcal{V} + \int_{\mathcal{S}} \mathcal{C} \cdot \boldsymbol{n}\ d\mathcal{S} \tag{2.3}$$

where ρ is the density, and $\mathcal{S} = \partial\mathcal{V}$ is the boundary of the current configuration domain $\mathcal{V}$. Note also that $\mathcal{V}(\boldsymbol{X}, t) = \varphi(V)$, $\mathcal{S}(\boldsymbol{X}, t) = \varphi(S)$. $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ will be defined later, on a case-by-case basis: in particular, $\mathcal{A}$ is the specific (i.e. per unit mass) conserved quantity, $\mathcal{B}$ is its specific rate of production/destruction, and $\mathcal{C}$ is the flux across the boundary $\mathcal{S}$. Specifically, $\mathcal{A}$ and $\mathcal{B}$ can be either scalar or vector quantities, and, correspondingly, $\mathcal{C}$ will be either a vector or tensor quantity.

It will prove useful to introduce the following formulas from classical continuum mechanics:

$$\int_{\mathcal{V}} \ldots\ d\mathcal{V} = \int_{V} \ldots\ J dV \tag{2.4}$$

$$\begin{aligned} \int_{\mathcal{S}} \ldots\ \boldsymbol{n}\ d\mathcal{S} &= \int_{S} \ldots\ J\boldsymbol{F}^{-T}\boldsymbol{N} dS \\ &= \int_{S} \ldots\ \mathbf{cof}\boldsymbol{F}\ \boldsymbol{N} dS \qquad \text{(Piola Transformation)} \end{aligned} \tag{2.5}$$

where V is the volume occupied by some portion of material in the reference configuration, $\boldsymbol{n}$ is the outward normal to the surface $\mathcal{S}$ in the current (transformed) configuration, while $\boldsymbol{N}$ is the outward normal to the surface S of the body in the reference configuration. In index notation, (2.5) reads

$$n_i d\mathcal{S} = J\frac{\partial X_A}{\partial x_i} N_A dS \tag{2.6}$$

Let us apply the master balance law (2.3) with $\mathcal{A} = 1$, $\mathcal{B} = 0$, $\mathcal{C} = \mathbf{0}$. Using (2.4), we readily obtain the mass conservation law:

$$\frac{d}{dt}\left(\int_{\mathcal{V}} \rho d\mathcal{V}\right) = \frac{d}{dt}\left(\int_{V} \rho J dV\right) = 0 \tag{2.7}$$

Equation (2.7) holds for any volume V, and, in Lagrangian coordinates, the volume V, associated to a mass in the initial (reference) configuration, is not a function of time. Therefore:

$$\rho\ J = \rho_0(\boldsymbol{X}) \tag{2.8}$$

Note that $\rho_0(\boldsymbol{X})$ represents the reference configuration density, but in a Lagrangian setting, it is also the density distribution of the initial state of the system, that is

$$\rho(\boldsymbol{X}, t = 0) = \rho_0(\boldsymbol{X}) \tag{2.9}$$

The momentum equation can be obtained similarly by setting:

$\mathcal{A} = \boldsymbol{v}$, the velocity,

$\mathcal{B} = \boldsymbol{g}$, the internal or body forces,

$\mathcal{C} = \boldsymbol{\sigma}$, the tensor representing the surface momentum fluxes, or surface stresses.

$$\frac{d}{dt}\left(\int_{\mathcal{V}} \rho\ \boldsymbol{v}\ d\mathcal{V}\right) = \int_{\mathcal{V}} \rho\ \boldsymbol{g}\ d\mathcal{V} + \int_{\mathcal{S}} \boldsymbol{\sigma n}\ d\mathcal{S} \tag{2.10}$$

By transforming (2.10) to the original configuration and using (2.8), it is easily seen that

$$\begin{aligned} \frac{d}{dt}\left(\int_{V} \rho_0\ \boldsymbol{v}\ dV\right) &= \int_{V} \rho_0\ \boldsymbol{g}\ dV + \int_{S} \boldsymbol{\sigma F}^{-T}\boldsymbol{N} J\ dS \\ &= \int_{V} \rho_0\ \boldsymbol{g}\ dV + \int_{S} \boldsymbol{P N}\ dS \end{aligned} \tag{2.11}$$

where $\boldsymbol{P} = J\boldsymbol{\sigma F}^{-T} = \boldsymbol{\sigma}\ \mathbf{cof}\boldsymbol{F}$ is the first Piola stress tensor. Again, (2.11) must hold for any volume V, and, in addition,

$$\frac{d}{dt}\left(\int_{V(\mathbf{X})} \rho_0(\boldsymbol{X})\ \boldsymbol{v}\ dV\right) = \int_{V(\mathbf{X})} \rho_0(\boldsymbol{X})\ \dot{\boldsymbol{v}}\ dV$$

Applying the Gauss divergence theorem in the reference configuration yields

$$\rho_0 \ \dot{\boldsymbol{v}} \quad = \quad \rho_0 \ \boldsymbol{g} + \mathbf{Div}\boldsymbol{P} \tag{2.12}$$

or, in index notation,

$$\rho_0 \ \dot{v}_i \quad = \quad \rho_0 \ g_i + \frac{\partial P_{iA}}{\partial X_A} \tag{2.13}$$

The energy equation is obtained by setting:

$\mathcal{A} = E = e + \frac{v^2}{2}$, the specific total energy (the sum of the internal energy e and the kinetic energy $\frac{v^2}{2} = \frac{v_i v_i}{2}$).

$\mathcal{B} = \boldsymbol{g} \cdot \boldsymbol{v} + r$, the sum of $\boldsymbol{g} \cdot \boldsymbol{v}$, the specific power due to body forces, and the specific rate of internal energy production r,

$\mathcal{C} = \boldsymbol{v} \cdot \boldsymbol{\sigma} + \boldsymbol{q}$, the sum of $\boldsymbol{v} \cdot \boldsymbol{\sigma}$, the power per unit area due to forces acting on the surface $\mathcal{S}$, and the heat fluxes $\boldsymbol{q}$ transmitted across $\mathcal{S}$.

$$\frac{d}{dt}\left(\int_{\mathcal{V}} \rho \ E \ d\mathcal{V}\right) \quad = \quad \int_{\mathcal{V}} \rho \ (\boldsymbol{v} \cdot \boldsymbol{g} + r) \ d\mathcal{V} + \int_{\mathcal{S}} (\boldsymbol{v} \ \boldsymbol{\sigma} + \boldsymbol{q}) \cdot \boldsymbol{n} \ d\mathcal{S} \tag{2.14}$$

Analogously to the case of (2.11), for the reference configuration, (2.14) reads

$$\frac{d}{dt}\left(\int_{V} \rho_0 \ E \ dV\right) \quad = \quad \int_{V} \rho_0 \ (\boldsymbol{v} \cdot \boldsymbol{g} + r) \ dV + \int_{S} (\boldsymbol{v}\boldsymbol{P} + \boldsymbol{Q}) \cdot \boldsymbol{N} \ dS \tag{2.15}$$

where $\boldsymbol{Q} = J\boldsymbol{q}\boldsymbol{F}^{-T} = \boldsymbol{q} \ \mathbf{cof}\boldsymbol{F}$ is the Piola-transformed heat flux. Application of the divergence theorem in the reference configuration and evaluation of the dependencies on $\boldsymbol{X}$ and t of the various quantities yield:

$$\rho_0 \ \dot{E} \quad = \quad \rho_0 \ \boldsymbol{g} \cdot \boldsymbol{v} + \rho_0 \ r + \mathbf{Div}(\boldsymbol{v} \cdot \boldsymbol{P} + \boldsymbol{Q}) \tag{2.16}$$

or

$$\rho_0 \ \dot{E} \quad = \quad \rho_0 \ (\dot{e} + v_i \dot{v}_i) = \rho_0 \ g_i v_i + \rho_0 \ r + \frac{\partial}{\partial X_A}(v_i P_{iA} + Q_A) \tag{2.17}$$

So far, the equations have been formulated under the only assumption of the Lagrangian reference frame. Constitutive laws need to be specified: In the next section, attention will be focused on the hydrodynamic systems.

2.2 Hydrodynamics principles

The hydrodynamic flow assumption implies a number of simplifications in the general formulation given by (2.8), (2.12), and (2.16). For hydrodynamic flows, shear stresses are neglected and the following constitutive relation is used for the stress tensor $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma} \;=\; -p\boldsymbol{I}_{n_{\mathrm{d}}\times n_{\mathrm{d}}} \tag{2.18}$$

or, in index notation,

$$\sigma_{ij} \;=\; -p\;\delta_{ij} \tag{2.19}$$

with δ_{ij}, the Kronecker tensor. The constitutive law for the pressure, or equation of state (EOS), can be cast in the so-called *Mie-Grüneisen* form, namely:

$$p \;=\; f_1(\rho;\rho_r,e_r) \;+\; f_2(\rho;\rho_r,e_r)\;e \tag{2.20}$$

where ρ_r and e_r are fixed reference thermodynamic states. Equation (2.20) can be succinctly expressed as

$$p \;=\; f_1(\rho) \;+\; f_2(\rho)\;e \tag{2.21}$$

If $f_1 = 0$ and $f_2 = (\gamma - 1)\;\rho$, the equation of state for ideal gases is recovered:

$$p \;=\; (\gamma-1)\;\rho\;e \tag{2.22}$$

In the following derivations, the general form (2.21) will be used, since it incorporates into a unique framework constitutive laws of very different nature, such as hydrodynamic fluids, ideal gases, co-volume gases, elastic-plastic solids with no strength (situation that can be achieved when bulk stresses in the material are larger than shear stresses by orders of magnitude), high explosives, etc.

It will prove very useful, for the derivations in the next sections, to recast (2.21) as follows

$$e = g_1(\rho) + g_2(\rho)\, p \tag{2.23}$$
$$g_1 = -\frac{f_1}{f_2} \tag{2.24}$$
$$g_2 = \frac{1}{f_2} \tag{2.25}$$

and also introduce the derivatives

$$g_1' = \frac{\partial g_1}{\partial \rho} = -\frac{f_1' f_2 - f_1 f_2'}{f_2^2} = -\frac{f_1'}{f_2} + \frac{f_1 f_2'}{f_2^2} \tag{2.26}$$
$$g_2' = \frac{\partial g_2}{\partial \rho} = -\frac{f_2'}{f_2^2} \tag{2.27}$$

For ideal gases, (2.24)–(2.27) read

$$g_1 = 0 \tag{2.28}$$
$$g_2 = \frac{1}{(\gamma-1)\rho} \tag{2.29}$$
$$g_1' = 0 \tag{2.30}$$
$$g_2' = -\frac{1}{(\gamma-1)\rho^2} \tag{2.31}$$

In order to close the equations (which contain terms in $\boldsymbol{F}$ and J), it is necessary to have information regarding the deformation undergone by the original configuration, through the following quantities:

$$\boldsymbol{u} \equiv \boldsymbol{x} - \boldsymbol{X} = \varphi(X,t) - \boldsymbol{X} \qquad \text{(displacement field definition)} \tag{2.32}$$
$$\boldsymbol{F} \equiv \frac{\partial \varphi}{\partial X} = 1 + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} \tag{2.33}$$
$$J \equiv \det(\boldsymbol{F}) \tag{2.34}$$

with the underlaying assumption that $J > 0$, that is, volume inversions are not allowed. The classical displacements/velocities relation

$$\dot{\boldsymbol{u}} = \boldsymbol{v} \tag{2.35}$$

closes the kinematics of the problem. We will also assume, for the sake of simplicity, the absence of body forces, heat transfer, and internal sources of energy, so that the complete set of equations – written in conservative form – amounts to:

$$\begin{aligned} \dot{\boldsymbol{u}} &= \boldsymbol{v} && (2.36)\\ \rho_0\, \dot{\boldsymbol{v}} &= \mathbf{Div}\boldsymbol{P} && (2.37)\\ \rho_0\, \dot{E} &= \mathbf{Div}(\boldsymbol{v}\cdot\boldsymbol{P}) && (2.38)\\ \rho\, J &= \rho_0 && (2.39) \end{aligned}$$

It is important to note a fact (see for example Ciarlet[10] for a proof):

$$\begin{aligned} \mathbf{Div}\boldsymbol{P} &= \mathbf{Div}(\boldsymbol{\sigma}\,\mathbf{cof}\boldsymbol{F})\\ &= (\mathbf{Grad}\,\boldsymbol{\sigma})\,\mathbf{cof}\boldsymbol{F} + \boldsymbol{\sigma}(\mathbf{Div}\,\mathbf{cof}\boldsymbol{F})\\ &= J\mathbf{div}\,\boldsymbol{\sigma} && (2.40) \end{aligned}$$

where **div** is the divergence in the deformed configuration reference system. The so-called Piola identity

$$\mathbf{Div}\,\mathbf{cof}\boldsymbol{F} \equiv \mathbf{0} \qquad (2.41)$$

was used in the derivation of (2.40). This result will be very useful in order to simplify the hydrodynamics equations in multiple dimensions.

2.3 The one-dimensional hydrodynamics equations

The one-dimensional hydrodynamic flow assumption allows further simplification of equations (2.36)–(2.39): In the one dimensional case, $\boldsymbol{X}$ is actually a scalar, so that:

$$\begin{aligned} \boldsymbol{X} &\equiv X_1 = X && (2.42)\\ \boldsymbol{x} &\equiv x_1 = x = \varphi(X,t) && (2.43)\\ \boldsymbol{u} &\equiv \boldsymbol{x} - \boldsymbol{X} = \varphi(X,t) - X = u && (2.44) \end{aligned}$$

$$\boldsymbol{F} \equiv F_1 = F = \frac{\partial \varphi}{\partial X} = 1 + \frac{\partial u}{\partial X} \tag{2.45}$$

$$J \equiv \det(\boldsymbol{F}) = \frac{\partial \varphi}{\partial X} \equiv F \tag{2.46}$$

In particular, (2.46) leads to the following simplification for the Piola stress tensor:

$$\boldsymbol{P} = J\boldsymbol{\sigma}\boldsymbol{F}^{-T} = J\sigma J^{-1} = -p \tag{2.47}$$

where J^{-1} is always well defined, since the mapping φ is always invertible.

The complete set of equations is given by

$$\dot{u} = v \tag{2.48}$$

$$\rho_0 \, \dot{v} = -\frac{\partial p}{\partial X} \tag{2.49}$$

$$\rho_0 \, \dot{E} = -\frac{\partial}{\partial X}(vp) \tag{2.50}$$

$$\rho \, J = \rho_0 \tag{2.51}$$

Equations (2.48-2.50) can be recast into vector form as

$$\dot{\boldsymbol{U}} + \boldsymbol{F}_{1,1} + \boldsymbol{Z} = \boldsymbol{0} \tag{2.52}$$

where $\boldsymbol{F}_{1,1} = \dfrac{\partial \boldsymbol{F}_1}{\partial X_1}$, and

$$\boldsymbol{U} = \begin{bmatrix} u \\ \rho_0 v \\ \rho_0 E \end{bmatrix} \qquad \boldsymbol{F}_1 = \begin{bmatrix} 0 \\ p \\ pv \end{bmatrix} \qquad \boldsymbol{Z} = \begin{bmatrix} -v \\ 0 \\ 0 \end{bmatrix} \tag{2.53}$$

It will be very useful for future derivations, to rearrange equations (2.48-2.50) in quasi-linear form, as follows:

$$\boldsymbol{A}_0 \dot{\boldsymbol{Y}} + \boldsymbol{A}_1 \boldsymbol{Y}_{,1} + \boldsymbol{C}\boldsymbol{Y} = \boldsymbol{0} \tag{2.54}$$

where

$$\boldsymbol{A}_0 = \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{Y}} \qquad \boldsymbol{A}_1 = \frac{\partial \boldsymbol{F}_1}{\partial \boldsymbol{Y}} \tag{2.55}$$

are the Jacobian matrices for the temporal and spatial fluxes, respectively. We still have to specify which variables are going to be the entries of $\boldsymbol{Y}$. It becomes clear by just performing a few attempts, that if continuous piecewise-linear functions for the kinematic and at least one of the thermodynamic variables are to be used, then the pressure is the natural candidate in the definition of Y.

In fact, alternative options, such as *conservation variables*, for which the total energy E is chosen as the last entry in $\boldsymbol{Y}$, cause the problem of taking derivatives of J with respect to X when computing the matrix $\boldsymbol{A}_1$. The first derivative of J with respect to X involves the second derivative of the displacement u with respect to X, which would be defined as a Dirac distribution over element edges if piecewise linear interpolation is used.

The following definition of $\boldsymbol{Y}$ will be then used, corresponding to some sort of *primitive variables formulation*:

$$\boldsymbol{Y} = \begin{bmatrix} u \\ v \\ p \end{bmatrix} \tag{2.56}$$

The following algebraic manipulations apply to $\dot{E}$:

$$\begin{aligned} \rho_0 \dot{E} &= \rho_0(\dot{e} + v\dot{v}) \\ &= \rho_0 \left((g_1' + g_2' \, p)\dot{\rho} + g_2 \dot{p} \right) + \rho_0 v\dot{v} \qquad (\text{ using (2.23), (2.26), (2.27) }) \\ &= -(g_1' + g_2' \, p)\frac{\rho_0^2}{J^2}\dot{J} + \rho_0 \, g_2 \dot{p} + \rho_0 v\dot{v} \qquad (\text{ using (2.8) }) \\ &= \rho_0 \, g_2 \dot{p} - (g_1' + g_2' \, p)\frac{\rho_0^2}{J^2}\frac{\partial v}{\partial X} + \rho_0 v\dot{v} \end{aligned} \tag{2.57}$$

where we have used the fact that $\dot{J} = \frac{\partial \dot{u}}{\partial X} = \frac{\partial v}{\partial X}$.

The second term on the last line of (2.57) does not contain a temporal derivative, and is therefore incorporated in the definition of the Jacobian $\boldsymbol{A}_1$. Rearrangement of the terms according to the structure of (2.54), yields:

$$\boldsymbol{A}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \rho_0 & 0 \\ 0 & \rho_0 v & \rho_0 \, g_2\left(\frac{\rho_0}{J}\right) \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & \Psi\left(\frac{\rho_0}{J}, p\right) & v \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.58}$$

where, using (2.8),

$$\Psi\Big(\frac{\rho_0}{J},p\Big) = p \;-\; \Big(\frac{\rho_0}{J}\Big)^2 \Big(\; g_1'\Big(\frac{\rho_0}{J}\Big) + g_2'\Big(\frac{\rho_0}{J}\Big)\;\; p \;\Big) \tag{2.59}$$

In the case of an ideal gas, using (2.8), (2.23), (2.30), (2.31), equation (2.57) leads to

$$\rho_0\dot{E} \;\;=\;\; \frac{J}{\gamma-1}\,\dot{p} + \frac{p}{\gamma-1}\frac{\partial v}{\partial X} + \rho_0 v\dot{v} \tag{2.60}$$

so that (2.58) become

$$\boldsymbol{A}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \rho_0 & 0 \\ 0 & \rho_0 v & \frac{J}{\gamma-1} \end{bmatrix} \quad , \quad \boldsymbol{A}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & \frac{\gamma}{\gamma-1}p & v \end{bmatrix} \tag{2.61}$$

2.4 The two-dimensional hydrodynamics equations

The extension to the multidimensional case is now straightforward, having already practiced, in the one-dimensional case, the typical manipulations involved. Because the computations are not very onerous in the case of two-dimensional flow, it would be good practice to start from there in the effort to generalize the current approach to higher dimensions.

The reader will soon realize that the whole primitive variables approach hinges on the fact that the relation between total energy E and pressure p is never much more complicated than the result of the algebraic manipulations in (2.57). This is truly the key to the multidimensional generalization, leading to important simplifications in the form of the stabilization terms, as the reader will see in Section 5.

The following notation is adopted for the two-dimensional case:

$$\boldsymbol{X} \;\;\equiv\;\; \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{2.62}$$

$$\boldsymbol{x} \;\;\equiv\;\; \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \varphi_1(X_1,X_2,t) \\ \varphi_2(X_1,X_2,t) \end{bmatrix} \tag{2.63}$$

$$\boldsymbol{u} \equiv \boldsymbol{x} - \boldsymbol{X} = \begin{bmatrix} \varphi_1(X_1, X_2, t) \\ \varphi_2(X_1, X_2, t) \end{bmatrix} - \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{2.64}$$

$$\boldsymbol{F} \equiv \frac{\partial \varphi}{\partial \boldsymbol{X}} = \boldsymbol{I}_{2\times 2} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = \begin{bmatrix} 1 + u_{1,1} & u_{2,1} \\ u_{1,2} & 1 + u_{2,2} \end{bmatrix} \tag{2.65}$$

$$J \equiv \det(\boldsymbol{F}) = (1 + u_{1,1})(1 + u_{2,2}) - u_{2,1}u_{1,2} \tag{2.66}$$

where the notation

$$u_{i,A} = \frac{\partial u_i}{\partial X_A} \tag{2.67}$$

has been used in (2.65) and (2.66), to provide a compact notation. In particular, (2.18) and (2.65) lead to the following expression for the Piola stress tensor:

$$\boldsymbol{P} = J\boldsymbol{\sigma}\boldsymbol{F}^{-T} = -p \begin{bmatrix} 1 + u_{2,2} & -u_{2,1} \\ -u_{1,2} & 1 + u_{1,1} \end{bmatrix} \tag{2.68}$$

In turn, (2.35) becomes now:

$$\begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{2.69}$$

As usual, it is possible to express the system in vector form:

$$\dot{\boldsymbol{U}} + \boldsymbol{F}_{1,1} + \boldsymbol{F}_{2,2} + \boldsymbol{Z} = \boldsymbol{0} \tag{2.70}$$

with, $\boldsymbol{F}_{i,A} = \dfrac{\partial \boldsymbol{F}_i}{\partial X_A}$, and

$$\boldsymbol{U} = \begin{bmatrix} u_1 \\ u_2 \\ \rho_0 v_1 \\ \rho_0 v_2 \\ \rho_0 E \end{bmatrix} \qquad \boldsymbol{Z} = \begin{bmatrix} -v_1 \\ -v_2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.71}$$

$$\boldsymbol{F}_1 = \begin{bmatrix} 0 \\ 0 \\ p(1+u_{2,2}) \\ -pu_{1,2} \\ pv_1(1+u_{2,2}) - pv_2u_{1,2} \end{bmatrix} \qquad \boldsymbol{F}_2 = \begin{bmatrix} 0 \\ 0 \\ -pu_{2,1} \\ p(1+u_{1,1}) \\ -pv_1u_{2,1} + pv_2(1+u_{1,1}) \end{bmatrix} \tag{2.72}$$

The quasilinear form of (2.70) reads:

$$\boldsymbol{A}_0\dot{\boldsymbol{Y}} + \boldsymbol{A}_1\boldsymbol{Y}_{,1} + \boldsymbol{A}_2\boldsymbol{Y}_{,2} + \boldsymbol{C}\boldsymbol{Y} = \boldsymbol{0} \tag{2.73}$$

with

$$\boldsymbol{A}_0 = \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{Y}} \qquad \boldsymbol{A}_1 = \frac{\partial \boldsymbol{F}_1}{\partial \boldsymbol{Y}} \qquad \boldsymbol{A}_2 = \frac{\partial \boldsymbol{F}_2}{\partial \boldsymbol{Y}} \tag{2.74}$$

As for the one-dimensional case, the choice of primitive variables prompts algebraic manipulations similar to (2.57):

$$\begin{aligned} \rho_0\dot{E} &= \rho_0(\dot{e} + v_1\dot{v}_1 + v_2\dot{v}_2) \\ &= \rho_0\,(\,(g_1' + g_2'\,p)\dot{\rho} \;+\; g_2\dot{p}\,) + \rho_0 v_1\dot{v}_1 + \rho_0 v_2\dot{v}_2 \\ &= \rho_0\; g_2\; \dot{p} - (g_1' + g_2'\,p)\frac{\rho_0^2}{J^2}\dot{J} + \rho_0 v_1\dot{v}_1 + \rho_0 v_2\dot{v}_2 \end{aligned} \tag{2.75}$$

with

$$\begin{aligned} \dot{J} &= \left.\frac{\partial}{\partial t}\right|_{\boldsymbol{X}} ((1+u_{1,1})(1+u_{2,2}) - u_{2,1}u_{1,2}) \\ &= (1+u_{2,2})v_{1,1} - u_{1,2}v_{2,1} \qquad \text{(term contributing to } \boldsymbol{A}_1\text{)} \\ &\quad -u_{2,1}v_{1,2} + (1+u_{1,1})v_{2,2} \qquad \text{(term contributing to } \boldsymbol{A}_2\text{)} \end{aligned} \tag{2.76}$$

It is important now to notice a very important cancellation in the second-derivative terms relative to the momentum and energy equations in (2.73), due to (2.40), as anticipated in section 2.2.

In fact we can clearly see that:

$$\begin{aligned}
\boldsymbol{F}_{1,1}+\boldsymbol{F}_{2,2} &= \begin{bmatrix} 0 \\ 0 \\ p_{,1}(1+u_{2,2})+pu_{2,21}-p_{,2}u_{2,1}-pu_{2,12} \\ -p_{,1}u_{1,2}-pu_{1,21}+p_{,2}(1+u_{1,1})+pu_{1,12} \\ (\boldsymbol{F}_{1,1}+\boldsymbol{F}_{2,2})_5 \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ 0 \\ p_{,1}(1+u_{2,2})-p_{,2}u_{2,1} \\ -p_{,1}u_{1,2}+p_{,2}(1+u_{1,1}) \\ (\boldsymbol{F}_{1,1}+\boldsymbol{F}_{2,2})_5 \end{bmatrix}
\end{aligned} \tag{2.77}$$

due to commutativity of second mixed derivatives, with

$$\begin{aligned}
(\boldsymbol{F}_{1,1}+\boldsymbol{F}_{2,2})_5 &= p_{,1}v_1(1+u_{2,2})+pv_{1,1}(1+u_{2,2})+pv_1u_{2,21} \\
&\quad -p_{,1}v_2u_{1,2}-pv_{2,1}u_{1,2}-pv_2u_{1,21} \\
&\quad -p_{,2}v_1u_{2,1}-pv_{1,2}u_{2,1}-pv_1u_{2,12} \\
&\quad +p_{,2}v_2(1+u_{1,1})+pv_{2,2}(1+u_{1,1})+pv_2u_{1,12} \\
&= (v_1(1+u_{2,2})-v_2u_{1,2})p_{,1} \\
&\quad +(-v_1u_{2,1}+v_2(1+u_{1,1}))p_{,2} \\
&\quad +(1+u_{2,2})pv_{1,1}-u_{1,2}pv_{2,1} \\
&\quad -u_{2,1}pv_{1,2}+(1+u_{1,1})pv_{2,2}
\end{aligned} \tag{2.78}$$

Arrangement of the terms according to the structure of (2.73), yields:

$$\boldsymbol{A}_0=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \rho_0 & 0 & 0 \\ 0 & 0 & 0 & \rho_0 & 0 \\ 0 & 0 & \rho_0 v_1 & \rho_0 v_2 & \rho_0 g_2\left(\frac{\rho_0}{J},p\right) \end{bmatrix},\quad \boldsymbol{C}=\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.79}$$

$$\boldsymbol{A}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+u_{2,2} \\ 0 & 0 & 0 & 0 & -u_{1,2} \\ 0 & 0 & (1+u_{2,2})\Psi\big(\frac{\rho_0}{J},p\big) & -\,u_{1,2}\Psi\big(\frac{\rho_0}{J},p\big) & v_1(1+u_{2,2}) - v_2 u_{1,2} \end{bmatrix} \tag{2.80}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -u_{2,1} \\ 0 & 0 & 0 & 0 & 1+u_{1,1} \\ 0 & 0 & -u_{2,1}\Psi\big(\frac{\rho_0}{J},p\big) & (1+u_{1,1})\Psi\big(\frac{\rho_0}{J},p\big) & -v_1 u_{2,1} + v_2(1+u_{1,1}) \end{bmatrix} \tag{2.81}$$

with $\Psi\big(\frac{\rho_0}{J},p\big)$ defined in (2.59).

For an ideal gas, $\Psi\big(\frac{\rho_0}{J},p\big) = \frac{\gamma}{\gamma-1}\,p$, so that:

$$\boldsymbol{A}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \rho_0 & 0 & 0 \\ 0 & 0 & 0 & \rho_0 & 0 \\ 0 & 0 & \rho_0 v_1 & \rho_0 v_2 & \frac{J}{\gamma-1} \end{bmatrix} \tag{2.82}$$

$$\boldsymbol{A}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+u_{2,2} \\ 0 & 0 & 0 & 0 & -u_{1,2} \\ 0 & 0 & \frac{\gamma}{\gamma-1}\,p\,(1+u_{2,2}) & -\frac{\gamma}{\gamma-1}\,p\,u_{1,2} & v_1(1+u_{2,2}) - v_2 u_{1,2} \end{bmatrix} \tag{2.83}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -u_{2,1} \\ 0 & 0 & 0 & 0 & 1+u_{1,1} \\ 0 & 0 & -\frac{\gamma}{\gamma-1}\, p\, u_{2,1} & \frac{\gamma}{\gamma-1}\, p\, (1+u_{1,1}) & -v_1 u_{2,1} + v_2(1+u_{1,1}) \end{bmatrix} \tag{2.84}$$

Chapter 3

A Review on Past and Present Hydrocode Technology

As a first step in the development of a new framework for Lagrangian hydrodynamic computations, it is important to review previous hydrocode technology. The basic numerical architecture with which the vast majority of hydrocodes are implemented will be presented, together with a historical review on the design of artificial viscosities. In order to convey the clearest possible picture, the one-dimensional equations have been chosen as a model problem.

3.1 A one-dimensional example

The model problem to be studied is readily expressed by the following equations:

$$\dot{u} = v \tag{3.1}$$

$$\rho_0 \, \dot{v} = -\frac{\partial (p+q)}{\partial X} \tag{3.2}$$

$$\rho_0 \, \dot{E} = -\frac{\partial}{\partial X}(v \, (p+q)) \tag{3.3}$$

where q is an appropriate artificial viscosity, yet to be defined, representing – in terms of multiscale analysis – the effect of the sub-grid (mesh-unresolved) component of the solution onto the (mesh-) resolved component of the solution.

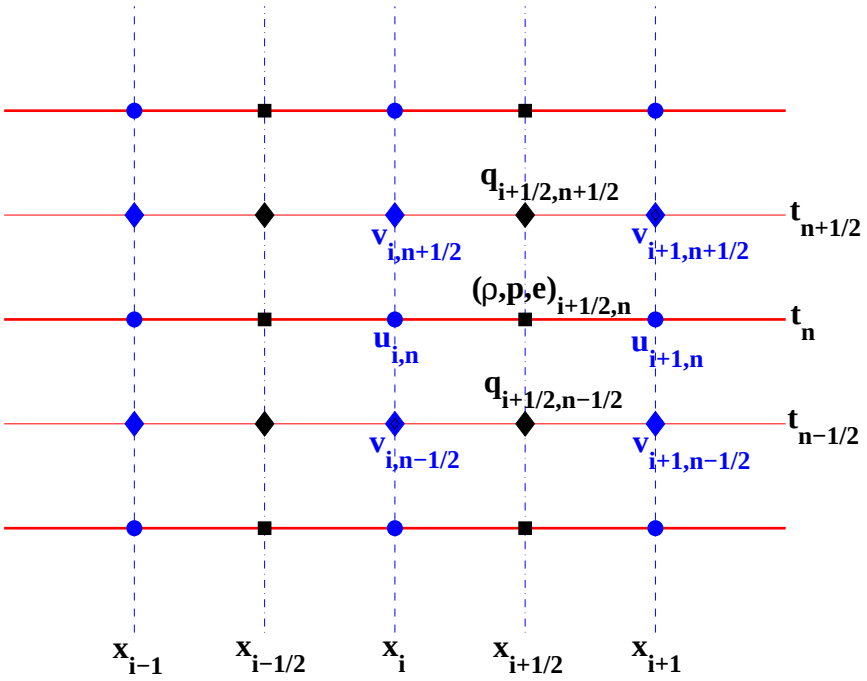

Figure 3.1: Sketch of the typical hydrocode numerical discretization in space and time for the one dimensional case on a uniform mesh. Blue circles indicate location of the displacements degrees-of-freedom; black squares are relative to densities, pressure, energy and any other thermodynamic variables (staggered in space); blue diamonds refer to velocities (staggered in time); black diamonds indicate artificial viscosities.

3.2 Hydrocode numerical architecture

In a typical hydrocode implementation, the equations are solved adopting a continuous, piecewise-linear (node-centered) representation for velocities and displacements, and a discontinuous, piecewise constant (cell-centered) representation for all the thermodynamic variables. The time integration is realized by means of a mid-point rule for the displacements, leading to staggering of the velocities in time with respect to all other variables (see Fig. 3.1 for a sketch of the numerical discretization in space/time).

This extremely ingenious algorithm is very simple from the implementation point of view and, consequently, very fast. On the other hand, the presented choice of function spaces poses a flexibility problem, since the mentioned staggered algorithm is very difficult to generalize to tetrahedral, unstructured meshes, of crucial importance in engineering applications, and, in fact, the use of hydrocodes is currently limited almost exclusively to quadrilateral/hexahedral meshes. When shocks occur in the flow, the presented hydrocode formulation becomes unstable, so that an artificial viscous flux q was

designed by Von Neumann and Richtmyer to stabilize the computations, while preserving conservation and accuracy. The viscous nature of the stabilizing term is also responsible for selecting the solution satisfying the entropy condition among all possible solutions of the conservation laws. The discussion of the artificial viscosity concept is postponed to section 3.3.1, while in the following discussion, emphasis will be given to the details of the numerical implementation.

3.2.1 Variational formulation

The domain to be considered is the one-dimensional interval $V = [X_l, X_r] = \cup_e V_e$ (V_e are the volumes of the cell/elements). For the sake of simplicity, only Dirichlet boundary conditions will be assumed, so that the variational form reads:

Find $u_h \in \mathcal{W}^h, v_h \in \mathcal{W}_g^h, e_h \in \mathcal{U}^h$, *such that, for all* $\varphi_h, \phi_h \in \mathcal{W}_0^h$, *and* $\psi_h \in \mathcal{U}^h$*:*

$$\sum_{e=1}^{n_{\rm el}} \int_{V_e} \varphi_h \ \dot{u}_h - \sum_{e=1}^{n_{\rm el}} \int_{V_e} \varphi_h v_h = 0 \tag{3.4}$$

$$\sum_{e=1}^{n_{\rm el}} \int_{V_e} \phi_h \ \rho_0 \dot{v}_h + \{\phi_h \ (p_h + q_h)\}|_{X_l}^{X_r} - \sum_{e=1}^{n_{\rm el}} \int_{V_e} \frac{\partial \phi_h}{\partial X} \ (p_h + q_h) = 0 \tag{3.5}$$

$$\sum_{e=1}^{n_{\rm el}} \int_{V_e} \psi_h \ \left\{ \rho_0 \dot{e}_h + (p_h + q_h) \ \frac{\partial v}{\partial X} \right\} = 0 \tag{3.6}$$

where:

$$\mathcal{U}^h = \left\{ w \in L^2(V) : w \in \mathcal{P}_0(V^e) \right\} \tag{3.7}$$

$$\mathcal{W}^h = \left\{ w \in C^0(V) \cap H^1(V) : w \in \mathcal{P}_1(V^e) \right\} \tag{3.8}$$

$$\mathcal{W}_g^h = \left\{ w \in C^0(V) \cap H^1(V) : w \in \mathcal{P}_1(V^e), w(X_l) = g_l, w(X_r) = g_r \right\} \tag{3.9}$$

$$\mathcal{W}_0^h = \left\{ w \in C^0(V) \cap H^1(V) : w \in \mathcal{P}_1(V^e), w(X_l) = 0, w(X_r) = 0 \right\} \tag{3.10}$$

$\mathcal{P}_m(V^e)$ is the space of polynomials of degree less or equal to m on the element (or cell) V^e, $\mathcal{W}^h$ is an affine set of continuous piecewise polynomials, $\mathcal{W}_g^h$ is subset of functions in $\mathcal{W}^h$ satisfying some prescribed Dirichlet boundary conditions, and $\mathcal{W}_0^h$ is the corresponding vector space with homogeneous Dirichlet boundary conditions. $\mathcal{U}^h$ is the space of discontinuous functions constant over each element/cell.

Introducing a basis for the function spaces previously defined, and applying the replacement $a_h = \dot{v}_h$ yields

$$(u_h, v_h, a_h) = \sum_{i=1}^{n_{\rm np}} (u, v, a)_i \ \phi_h^i, \qquad \phi_h^i \in \mathcal{W}^h, \quad \text{(pcw. linears)} \tag{3.11}$$

$$(\rho_0, p_h, e_h, q_h) = \sum_{i=1}^{n_{\rm el}} (\rho_0, p, e, q)_{i+1/2} \ \psi_h^i, \qquad \psi_h^i \in \mathcal{U}^h, \quad \text{(constants)} \tag{3.12}$$

Notice that the discretized displacement equation is an ordinary differential equation in the degrees-of-freedom for the displacements. The artificial viscous flux q is discretized as q_h, analogously to a pressure term. In classical hydrocode implementations, the following derivation

$$0 = \dot{\rho}_0 = \frac{\partial}{\partial t}(\rho \ J) \quad \Rightarrow \quad \dot{\rho} = -\frac{\rho}{J}\dot{J} \tag{3.13}$$

$$\frac{\partial}{\partial t}\left(\frac{1}{\rho}\right) = -\frac{1}{\rho^2}\dot{\rho} = \frac{1}{\rho \ J}\dot{J} \tag{3.14}$$

$$= \frac{1}{\rho_0}\dot{J} \qquad \text{(multi-dimensional flow)} \tag{3.15}$$

$$= \frac{1}{\rho_0}\frac{\partial v}{\partial X} \qquad \text{(one-dimensional flow)} \tag{3.16}$$

it used to simplify (3.6) to

$$\sum_{e=1}^{n_{\rm el}} \int_{V_e} \psi_h \ \rho_0 \left\{ \dot{e}_h + (p_h + q_h) \ \frac{\partial}{\partial t}\left(\frac{1}{\rho}\right) \right\} = 0 \tag{3.17}$$

3.2.2 Central-differences time integrator

A mid-point rule is usually applied to the displacement and momentum equations, which are *staggered* with respect to one another. For the energy equation, at least in one-dimensional computations, (3.17) is integrated in time by just expressing the time derivatives in terms of discrete differences. Since all the quantities in the energy equation are second-order accurate in time, as a consequence, also the updated value of the energy enjoys second-order accuracy.

The final result is an *explicit* algorithm summarized in the following sequence of steps (the reader may refer to the sketch in Fig. 3.1):

Momentum I. Assemble the acceleration vector,

$$a_i^n = (M^{L^{-1}})_{ii}(F_i^{ext} - F_i^{int}) \tag{3.18}$$

with

$$F_i^{int} = -\left(\mathop{\mathbf{A}}_{e=1}^{n_{\text{el}}} \int_{V_e} \frac{\partial \phi_h}{\partial X} \, (p_h + q_h) \right)_i \tag{3.19}$$

$$F_i^{ext} = -\Big((p_h(X_r) + q_h(X_r))\chi_{X_r} - (p_h(X_l) + q_h(X_l))\chi_{X_l} \Big)_i \tag{3.20}$$

$$(M^{L^{-1}})_{ii} = -\left(\mathop{\mathbf{A}}_{e=1}^{n_{\text{el}}} \int_{V_e} \rho_0 N_a \delta_{ab} \right)_i \tag{3.21}$$

where N_a is the local shape function, δ_{ab} is the Kronecker tensor, and $\mathbf{A}$ is the assembly operator. χ_{X_l} and χ_{X_r} are the characteristic functions on the left and right boundary faces, respectively (i.e., nodes, in the one-dimensional case): The local contributions to the external force vector are assembled only on the boundary elements.

Momentum II. Integrate in time the velocities (mid-point at the half time-step)

$$v_i^{n+1/2} = v_i^{n-1/2} + \frac{a_i^n}{2}(\Delta t^n + \Delta t^{n-1}) \tag{3.22}$$

Displacements. Integrate in time the displacements/current configuration (mid-point rule)

$$u_i^{n+1} = u_i^n + v_i^{n+1/2}\Delta t^n \qquad \text{or} \qquad x_i^{n+1} = x_i^n + v_i^{n+1/2}\Delta t^n \tag{3.23}$$

Density. Update the density according to

$$\left(\frac{1}{\rho_{i+1/2}^{n+1}} \right) = \left(\frac{1}{\rho_{i+1/2}^{n}} \right) \frac{x_{i+1}^{n+1} - x_i^{n+1}}{x_{i+1}^n - x_i^n} \tag{3.24}$$

Energy. Notice that there is not even assembly to perform, given the finite-volume type approximation for the thermodynamic variables involved:

$$e_{i+1/2}^{n+1} = e_{i+1/2}^{n} - \frac{1}{2}\left(p_{i+1/2}^{n} + p_{i+1/2}^{n+1} + q_{i+1/2}^{n+1/2} + q_{i+1/2}^{n-1/2}\right)\left(\frac{1}{\rho_{i+1/2}^{n+1}} - \frac{1}{\rho_{i+1/2}^{n}}\right) \tag{3.25}$$

Applying the equation of state (2.21) to (3.25), yields:

$$e_{i+1/2}^{n+1} = \frac{e_{i+1/2}^{n} - \frac{1}{2}\left(p_{i+1/2}^{n} + f_1(\rho_{i+1/2}^{n+1}) + q_{i+1/2}^{n+1/2} + q_{i+1/2}^{n-1/2}\right)\left(\frac{1}{\rho_{i+1/2}^{n+1}} - \frac{1}{\rho_{i+1/2}^{n}}\right)}{1 + \frac{f_2(\rho_{i+1/2}^{n+1})}{2}\left(\frac{1}{\rho_{i+1/2}^{n+1}} - \frac{1}{\rho_{i}^{n}}\right)} \tag{3.26}$$

Pressure. The equation of state (2.21) can be used again for the update of the pressure, according to

$$p_{i+1/2}^{n+1} = f_1(\rho_{i+1/2}^{n+1}) + f_2(\rho_{i+1/2}^{n+1})\, e_{i+1/2}^{n+1} \tag{3.27}$$

The formulation proposed is a prototype of the typical approaches followed in production hydrocodes. A few variants are possible, especially regarding time integration strategies for velocities/displacements and the energy update.

The presented implementation was adopted to generate benchmark results for the numerical experiments discussed in section 8.

3.3 Artificial viscosities

The formulation outlined in the previous section is *not* stable in the presence of shocks, unless the artificial viscosity operator q is activated. The early concept of artificial viscosity dates back to the years of the Manhattan Project at Los Alamos and it is due to Von Neumann and Richtmyer [66]. The following sections will describe the main aspects of the original concept and its latest developments.

3.3.1 The original idea of Von Neumann and Richtmyer

The original design by Von Neumann and Richtmyer [66] for a nonlinear artificial viscosity reads as follows:

$$\begin{aligned} q \approx q_{VN-R} &= \begin{cases} -\rho\,(c_{VN-R}\,h)^2\,|\partial_x v|\,\partial_x v, & \text{for } \partial_x v < 0 \\ 0, & \text{otherwise} \end{cases} \\ &= -\rho\,(c_{VN-R}\,h)^2\,|\partial_x v|\,\partial_x v\;\chi\{\partial_x v < 0\} \end{aligned} \tag{3.28}$$

where all the gradients are with respect to the current configuration, h is a measure of the current configuration mesh size and χ is the indicator function. The artificial viscosity is applied only in the case of compressions ($\partial_x v < 0$), and is set to zero in the case of expansions, for which the solution is smooth. More simply, transforming the gradients back to the reference configuration,

$$q \approx q_{VN-R} = -\frac{\rho_0}{J}\,(c_{VN-R}\,\Delta X)^2 \left|\frac{\partial v}{\partial X}\right| \frac{\partial v}{\partial X}\;\chi\left\{\frac{\partial v}{\partial X} < 0\right\} \tag{3.29}$$

For a detailed explanation of the method by Von Neumann and Richtmyer the reader can refer to their original paper [66], in which a thorough analysis is presented, from both the physical and numerical perspective. A summary of the main issues addressed in the original paper is presented in the following list of remarks.

Remarks

1. The viscosity q_{VN-R} is activated (i.e., non-negligible or non-zero) when high (negative) velocity gradients are present, as in the case of shocks.

2. q_{VN-R} is not activated by contact discontinuities (CD), that is, interfaces with an abrupt change of entropy.

3. q_{VN-R} smoothes the solution (all the variables and not only the velocity) when the gradients are too large to be resolved by the numerical discretization.

4. q_{VN-R} scales quadratically with respect to the mesh size, so it becomes negligible as the mesh is refined, and it is consistent with an algorithm second-order accurate in space.

5. q_{VN-R} is consistent with respect to conservation laws, because it vanishes away from shocks. Therefore the Rankine-Hugoniot relations are satisfied if the volume used to compute the conserved quantities budgets extends far enough away from the shock location.

The *simplicity* of the idea of Von Neumann and Richtmyer and its *consistency* with respect to conservation laws and second-order schemes was the key to its success.

3.3.2 Further improvements

After the original formulation, a number of researchers, mainly in the United States and the former Soviet Union, expanded the capabilities of artificial viscosity models. One main point to be made, is that at the time of the Manhattan Project, the computational power was very limited, and the interest of the researchers was to evaluate feasibility of certain design concepts before developing small- or full-scale experimental programs. Therefore, at least in the early years, the nonlinear viscosities were applied only to one-dimensional problems, or problems in two/three dimensions that could be reduced by cylindrical or spherical symmetry to a one-dimensional system of equations.

Later on, the success of artificial viscosities spurred the researchers in the attempt to compute fully three-dimensional problems, and a number of issues arose, hidden behind the "one-dimensionality" of the simulations undertaken until then. It was observed (see Wilkins [67] for a comprehensive review on artificial viscosity models and account of references) that, in some instances, so-called "linear acoustic instabilities" were arising both in one- and multi-dimensional computations, for which additional stabilization was required.

A couple of decades after the original paper of Von Neumann and Richtmyer, a modification was successfully proposed by Kuropatenko [47], and became an integral part of the artificial viscosity techniques implemented to date. The rationale behind the modification was the fact that the Rankine-Hugoniot conditions lead to an expression relating the pressure on each side of the shock as follows:

$$P_1 = P_0 + \frac{\gamma+1}{4}\rho_0(\Delta U)^2 + \rho_0|\Delta U|\sqrt{\left(\frac{\gamma+1}{4}\right)^2(\Delta U)^2 + c_s^2} \tag{3.30}$$

It can be shown that (3.30) is the typical equation to be solved when seeking the solution to a local (cell) Riemann problem in Godunov-type methods. Furthermore, by means of asymptotic analysis, Kuropatenko [47] argued that the artificial viscosity behaves like $\frac{\gamma+1}{2}\rho\ |\nabla \overline{\boldsymbol{v}}|\Delta x^2$ in the limit of strong shocks and like $\rho c_s \Delta x$ in the limit of weak shocks.

The initial concept proposed by Kuropatenko was further modified to achieve more straightforward implementations, and the typical form currently used is:

$$q_K = q_{VN-R} + q_{LIN} \tag{3.31}$$

with

$$q_{LIN} = -\rho\ c_{LIN}\ h\ c_s\ \ |\partial_x v|\ \ \chi\{\partial_x v < 0\} \tag{3.32}$$

where c_s is the local sound speed.

3.3.3 A remark on "acoustic instabilities"

The acoustic instabilities mentioned in the previous section, in the opinion of the author, are actually "an old friend". In fact it can be shown by means of a multiscale analysis that the linearized sub-grid scale Green's function problem for the SUPG stabilization of the hydrodynamic equations is given by a wave-equation kernel. Therefore, it seems that what Kuropatenko [47] was actually trying to obtain from his analysis, was some sort of SUPG stabilization for multidimensional hyperbolic systems.

The correspondence is not perfectly stringent, since the choice of interpolation spaces for the pressures are different for stabilized Galerkin methods and hydrocodes implementations, but the occurrence of instabilities for weak shocks conditions seems to indicate that also the hydrocode formulations need some sort of SUPG stabilization.

In chapter 5, the full SUPG approach is pursued and detailed, using equal-order, continuous, linear interpolation for both kinematic and thermodynamic variables.

3.3.4 The state of the art on artificial viscosities

At the moment, further developments on the artificial viscosity concept have been proposed, such as the application of flux limiters leading to total variation diminishing/total

variation bounded (TVD/TVB) artificial viscosities (see, e.g., the work by Christiansen [9], or the paper by Benson [3] for a thorough review of recent advances in hydrocode technology). While these ideas provide an improvement in some instances (sharper shock profiles), the present state of research indicates that the artificial viscosities cannot completely prevent node-to-node oscillations.

3.3.5 Additional issues in multiple dimensions

Usually hydrocodes are developed on structured or unstructured quadrilateral or hexahedral grids. There are very few examples with triangular or tetrahedral meshes, mainly because tetrahedral elements are intrinsically stiffer than hexahedral ones, and suffer less from *hourglass* instability, although this aspect is not regarded as an advantage in the hydrocode community, due to the fear of locking occurrence in the incompressibility limit (of great importance since many materials are intrinsically incompressible or assume such behavior for large plastic deformations).

SUPG stabilization acts correcting such behavior in triangular-type meshes, and it is well-documented that SUPG-stabilized finite elements do not suffer from hourglass instability either. For this reason, SUPG stabilization, may represent a framework with significant potential for new generation hydrocodes.

Chapter 4

Variational Formulations

The present chapter introduces the variational formulation that will be the basis for the novel SUPG framework outlined in chapter 5. The choice of the variational formulation and the time-marching method are defining the performance of the discretization. In our case, special care was devoted to the choice of the time integration, optimizing it for SUPG stabilization purposes. The following sections detail the derivations for a first- and second-order time accurate algorithms.

4.1 A note on time integrators and stabilization

It is well known in the research community of stabilized finite element methods, that the stabilization design is crucial for strongly unsteady flows. In our particular case, the presence/formation of strong shocks and the typical unsteadiness of any Lagrangian hydrodynamics simulation pose an important problem on the overall performance of the method.

It is also well known that space-time formulations are very well suited to devise good stabilization terms for unsteady, convective-dominated systems, since the temporal axis is almost treated as an additional spatial axis. Although it is possible to devise good stabilized methods based upon a semi-discrete formulation, the space-time framework eases significantly the task.

Previous implementations of Euler and Navier-Stokes equations using an Eulerian

reference frame – as in [24], [25], [63], and [64] – have resorted to *continuous–in–space/discontinuous–in–time* formulations, which are second-order accurate in space, and first- or third-order accurate in time. It can be argued that both approaches do not offer the best compromise between accuracy and computational complexity:

- First-order schemes are not accurate, unless convergence to a steady state condition is the goal of the simulation, in which the steady state solution enjoys second-order spatial accuracy.

- Third-order discontinuous Galerkin (DG)-in-time algorithms are, in a sense, excessively accurate, since the global error in the simulation will be dominated by the spatial error (second-order in the mesh spacing). For implicit time integration, advantage can be taken by increasing the time-step to larger values with respect to second-order algorithms.

  However, third order methods require twice as much storage and between two and four times more computational effort than first-order methods, due to the fact that they invoke a discontinuous piecewise-linear discretization in time, requiring the update of two full solution vectors per time-step.

  Possible variants as in Shakib, Hughes and Johan [64] reduce the onerous problem of the time-step solve by splitting the original problem into two separate updates, each as expensive as a first-order solve. This second approach, although less expensive than the standard one, is more expensive than a scheme that only invokes a single solution vector solve.

In the following sections, a second-order formulation will be presented and detailed, as it represents a very good compromise between accuracy and efficiency. The proposed formulation will be developed in parallel with the classical first-order approach.

4.2 Two time-integration algorithms

Two different time-integrators will be presented momentarily. In order to clarify their structure, it is useful to refer to the simple example of the (generally nonlinear) ordinary differential equation (ODE):

$$\dot{y} = f(y(t)) \tag{4.1}$$

$$y(0) = y_0 \tag{4.2}$$

The space-time formulation for (4.1)–(4.2) reads:

$$w(t_{n+1}^-)y(t_{n+1}^-) - w(t_n^+)y(t_n^-) - \int_{t_n}^{t_{n+1}} \left(w_{,s}(s)y(s) + w(s)f(y(s)) \right) \, ds = 0 \tag{4.3}$$

The numerical scheme will be entirely defined by the function spaces used to represent the solution y and the test function w.

4.2.1 A first-order Discontinuous-Galerkin time integrator

The first order algorithm proposed is the most widely know discontinuous-in-time algorithm for ODE integration. The reader should refer to Shakib, Hughes and Johan [64] and references therein for a complete analysis.

The function spaces are fixed so that both test and trial functions are discontinuous, piecewise-constant, along time-intervals, so that equation (4.3) yields:

$$w(t_{n+1}^-)y(t_{n+1}^-) - w(t_n^+)y(t_n^-) - \int_{t_n}^{t_{n+1}} w(s)f(y(s)) \, ds = 0 \tag{4.4}$$

where $w(s) = \chi\{]t_n, t_{n+1}]\}$ and $y(s) = y(t_{n+1}^-)\chi\{]t_n, t_{n+1}]\}$, implying

$$y(t_{n+1}^-) - y(t_n^-) - \Delta t_n f(y(t_{n+1}^-) = 0 \tag{4.5}$$

In the case of linear systems, that is if $f(y) = ay$, (4.5) reduces to the well-known backward Euler method:

$$y_{n+1} - y_n = a\Delta t_n y_{n+1} \tag{4.6}$$

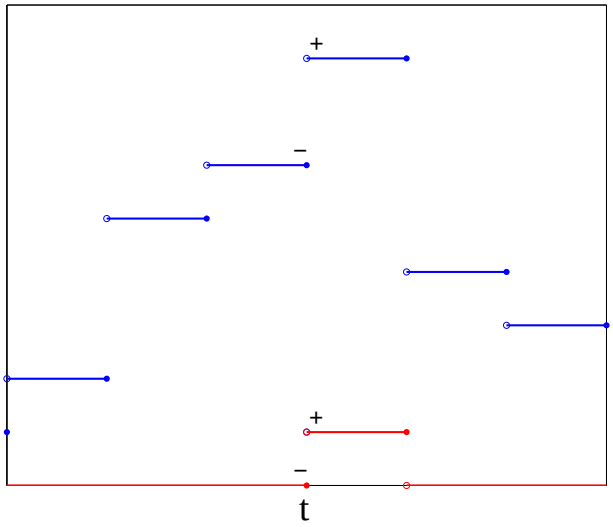

Figure 4.1: First-order algorithm: Sketch of the typical solution (blue) and test function (red).

where the super index "−" has been dropped for convenience. Figure 4.1 shows the solution and typical test function for the first-order time integrator.

4.2.2 A second-order Petrov-Galerkin time integrator

A higher order time integrator can be obtained via a space-time algorithm based on continuous, piecewise-linear trial functions in both space and time.

The method dates back to Aziz and Monk [2], Hulme [38], Jamet [42]. Recent references, which also contain a detailed review of the various contributions, applications to parabolic and second-order hyperbolic problems, and global error analysis, are Estep and French [14], French [17, 18], French and Jensen [19], and French and Paterson [20].

If the function spaces are fixed so that the test functions are discontinuous, piecewise-constant, and the trial functions are continuous piecewise-linear, (4.3) reduces to:

$$w(t_{n+1}^-)y(t_{n+1}) - w(t_n^+)y(t_n) - \int_{t_n}^{t_{n+1}} w(s)f(y(s))\,ds = 0 \tag{4.7}$$

The current formulation can be proved to be second-order accurate, and requires only one update (i.e., $y(t_{n+1})$). Additionally, it has some nice energy preserving properties when applied to Hamiltonian systems, although this does not represent a crucial feature in the present case. For linear systems ($f(y) = ay$), equation (4.7) reduces to the trapezoidal

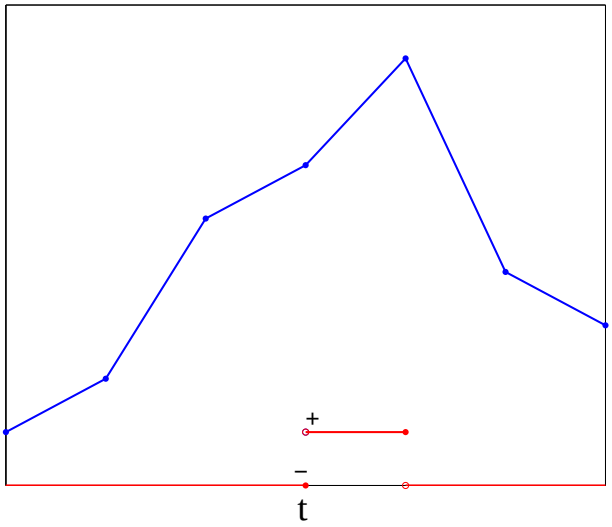

Figure 4.2: Second-order algorithm: Sketch of the typical solution (blue) and test function (red).

time-integration rule for the nodal degrees-of-freedom $y_k = y(t_k)$

$$y_{n+1} - y_n = \frac{a\Delta t_n}{2}\,(y_{n+1} + y_n) \tag{4.8}$$

and to the well-known Crank-Nicolson scheme, in the case of a general linear partial differential equations (PDE). Figure 4.2 shows the solution and typical test function for the first-order time integrator.

4.3 Space-time variational formulations

Given a partition $0 < t_1 < t_2 < \ldots < t_N = T$ of the time interval $I =]0, T[$, let $I_n =]t_n, t_{n+1}[$, so that $]0, T[= \bigcup_{n=0}^{N-1} I_n$. The space-time domain $Q = V \times I$ can be divided into time slabs

$$Q_n = V \times I_n \tag{4.9}$$

with boundary $P_n = S \times I_n$. A sketch of the general discretization in space-time is presented in Figure 4.3. It can be seen easily that, in general, the elements can assume fairly complex shapes in space-time. In what follows however, the numerical implementations will only make use of discretizations *prismatic* in time, for which the domain V is further partitioned into space-subdomains V^e (elements in space) fixed with respect

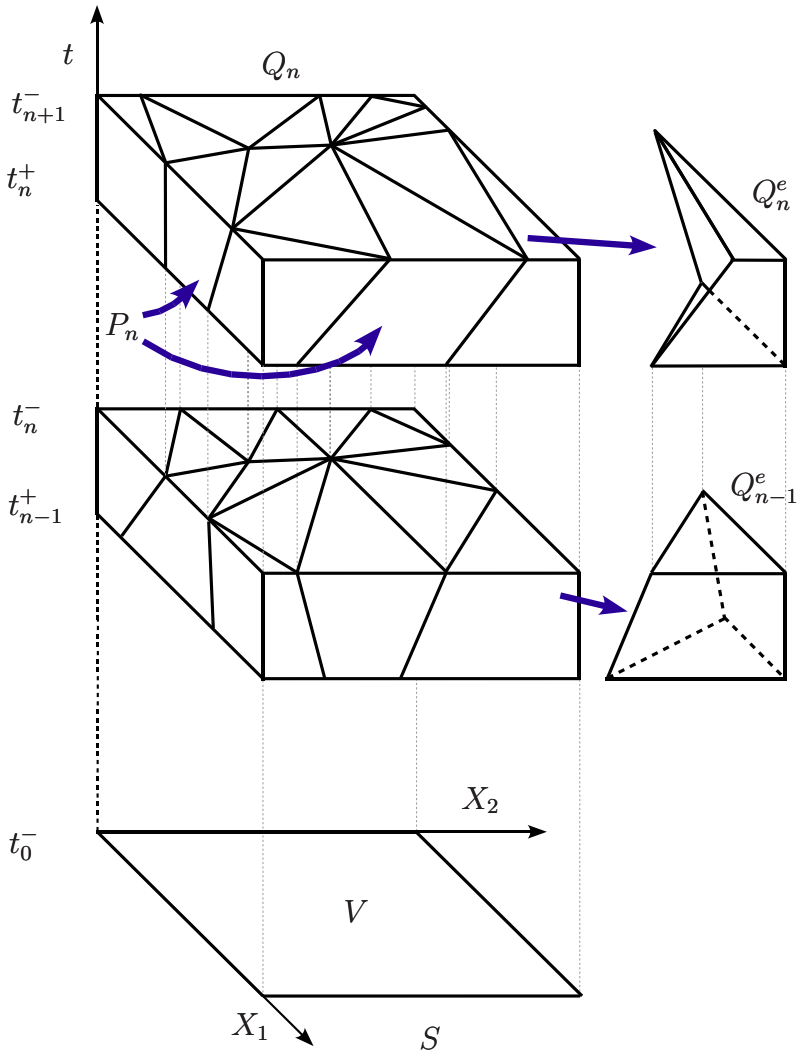

Figure 4.3: General finite element discretization in space-time.

to time. Thus $V = \bigcup_{e=1}^{n_{\rm el}} V^e$, and, consequently, a typical space-time element is given by the prism

$$Q_n^e = V^e \times I_n \tag{4.10}$$

It will be also assumed that the space-time boundary is partitioned as $P_n = P_n^g \cup P_n^h$, $P_n^g \cap P_n^h = \emptyset$ (i.e., P_n^h is divided into a Dirichlet boundary P_n^g and a Neumann boundary P_n^h). Using the notation $\boldsymbol{V}(\boldsymbol{X}, t_n^\pm) = \lim_{t \to t_n^\pm} \boldsymbol{V}(\boldsymbol{X}, t)$, the classical space-time variational formulation is defined as follows:

Find $\boldsymbol{Y}^h \in \mathcal{S}_n^h$, *such that* $\forall \boldsymbol{W}^h \in \mathcal{V}_n^h$

$$\mathcal{B}(\boldsymbol{W}^h, \boldsymbol{Y}^h) + \mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h) + \mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) \;=\; \mathcal{F}(\boldsymbol{W}^h) \tag{4.11}$$

with

$$\begin{aligned}
\mathcal{B}(\boldsymbol{W}^h, \boldsymbol{Y}^h) \;=\; & \int_V \boldsymbol{W}^h(\boldsymbol{X}, t_{n+1}^-) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-)) dV \\
& - \int_V \boldsymbol{W}^h(\boldsymbol{X}, t_n^+) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_n^-)) dV \\
& + \int_{Q_n} \left(-\boldsymbol{W}^h_{,t} \cdot \boldsymbol{U}(\boldsymbol{Y}^h) - \boldsymbol{W}^h_{,i} \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h) + \boldsymbol{W}^h \cdot \boldsymbol{Z}(\boldsymbol{Y}^h) \right) \; dQ \\
& + \int_{P_n^g} \boldsymbol{W}^h \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h) N_i \; dP
\end{aligned} \tag{4.12}$$

$$\mathcal{F}(\boldsymbol{W}^h) \;=\; - \int_{P_n^h} \boldsymbol{W}^h \cdot \boldsymbol{H} \; dP \tag{4.13}$$

where $\boldsymbol{H}$ represent the Neumann flux, and the form $\mathcal{B}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$, the stabilization term $\mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ and the Discontinuity Capturing operator $\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ are linear in the first argument, and nonlinear in the second. The choice of the spaces $\mathcal{S}_n^h$ and $\mathcal{V}_n^h$ will completely define the time integrator, causing (4.13) to further simplify.

4.3.1 A note on global conservation

It is important to realize that the formulation is globally conservative. In fact, suppose to set equal to unity one of the entries of the test vector $\boldsymbol{W}^h$, corresponding to the momentum or energy equations, and let the remaining entries be zero. Namely, $\boldsymbol{W}_i^h = 1$, for some $i \in \{n_\mathrm{d} + 1, \ldots, 2n_\mathrm{d} + 1\}$, and $\boldsymbol{W}_j^h = 0$ if $j \neq i$. This choice is possible because the test functions are linear in space and constant in time. Then, (4.11) reduces to:

$$\begin{aligned}
\int_V \boldsymbol{U}_i(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-)) \; dV \;=\; & \int_V \boldsymbol{U}_i(\boldsymbol{Y}^h(\boldsymbol{X}, t_n^-)) \; dV \\
& - \int_{P_n^g} \boldsymbol{W}^h \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h) N_i \; dP - \int_{P_n^h} \boldsymbol{W}^h \cdot \boldsymbol{H} \; dP
\end{aligned} \tag{4.14}$$

which is a statement of conservation from time t_n^- to time t_{n+1}^- for the $\boldsymbol{U}_i$ entry of the conservation variables vector $\boldsymbol{U}$, provided the boundary integrals vanish. In order to realize

how (4.14) is obtained, notice that, by definition (see (5.9) and (6.1)), $\mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ and $\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ vanish if $\boldsymbol{W}^h$ is a constant, and so happens for $\boldsymbol{W}^h_{,t}$ and $\boldsymbol{W}^h_{,i}$. Finally, $\boldsymbol{Z}(\boldsymbol{Y}^h)$, has zero entries for the momentum and energy equation, since it is not a true source term, but, rather, a kinematic term containing the components of the velocity field for the displacement equations.

4.3.2 Euler-Lagrange equations

The formulation is best understood through the Euler-Lagrange equations, obtained by integration by parts, once sufficient regularity of the solution has been assumed:

$$\begin{aligned}
&\int_{Q_n} \boldsymbol{W}^h \cdot \left\{ \boldsymbol{U}_{,t}(\boldsymbol{Y}^h) + \boldsymbol{F}_{i,i}(\boldsymbol{Y}^h) + \boldsymbol{Z}(\boldsymbol{Y}^h) \right\} \, dQ \\
+ &\int_V \boldsymbol{W}^h(\boldsymbol{X}, t_n^+) \cdot \left\{ \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_n^+)) - \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_n^-)) \right\} \, dV \\
&- \int_{P_n^h} \boldsymbol{W}^h \cdot \left\{ \boldsymbol{F}_i(\boldsymbol{Y}^h) N_i - \boldsymbol{H} \right\} dP \\
&+ \mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h) + \mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) \;=\; 0
\end{aligned} \tag{4.15}$$

As it can be easily observed, the integral on the first line is testing the system of PDEs inside the space-time domain, the integral on the second line enforces weak continuity of the solution across slabs, yielding a Time-Discontinuous Galerkin method, *upwinded* in time, in order to satisfy the *causality* principle of temporal evolution. The space-time surface integral on the third line tests the Neumann boundary conditions (Dirichlet conditions will be embedded in the definition of the trial functions), while the last two terms are yet to be defined.

4.3.3 First-order formulation

In terms of function spaces, the first-order algorithm can be obtained by assuming that both the trial space $\mathcal{S}_n^h$ and test function space $\mathcal{V}_n^h$ are given by functions that are continuous, piecewise-linear in space and discontinuous, piecewise-constant in time (see Fig. 4.4 for details in the one-dimensional case). Of course the trial/test functions differ because

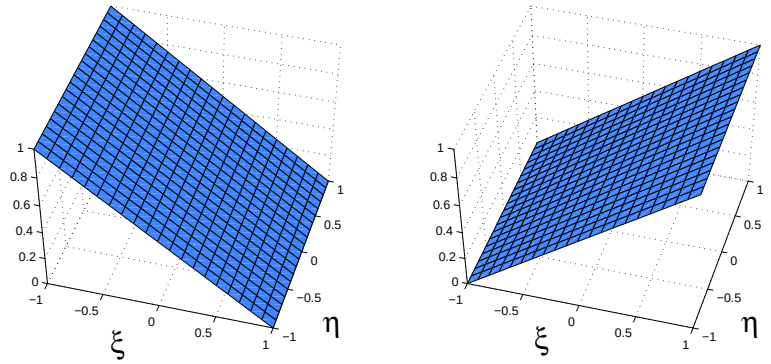

Figure 4.4: Local test functions for the first/second-order time integrator, in the one-dimensional case. The functions shown also represent the local trial function space for the first-order time integrator. ξ is the local space coordinate, while η is the local time coordinate.

of the boundary conditions.

$$\mathcal{S}_n^h = \left\{ \boldsymbol{V}^h : \boldsymbol{V}^h \in (C^0(Q))^m, \boldsymbol{V}^h\big|_{Q_n^e} \in (\mathcal{P}_1(Q_n^e))^m, \boldsymbol{V}^h = \boldsymbol{g}(t) \text{ on } P_n^g \right\} \tag{4.16}$$

$$\mathcal{V}_n^h = \left\{ \boldsymbol{W}^h : \boldsymbol{W}^h\big|_V \in (C^0(V))^m, \right. \\ \left. \boldsymbol{W}^h\big|_{Q_n^e} \in (\mathcal{P}_1(V^e) \times \mathcal{P}_0(I_n))^m, \boldsymbol{V}^h = \boldsymbol{0} \text{ on } P_n^g \right\} \tag{4.17}$$

with $m = 2n_{\mathrm{d}} + 1, \quad n_{\mathrm{d}} \in \{1, 2, 3\}$. $\boldsymbol{g}$ is the vector of Dirichlet boundary conditions, and P_n^g is the portion of P_n where Dirichlet boundary conditions are imposed. $\mathcal{P}_k(Q_n^e)$ and $\mathcal{P}_k(V^e)$ are the spaces of polynomials of degree less or equal to k, on the domains Q_n^e and V^e, respectively. Therefore (4.13) yields:

$$\begin{aligned} 0 = & \int_V \left(\boldsymbol{W}^h(\boldsymbol{X}, t_{n+1}^-) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-)) - \boldsymbol{W}^h(\boldsymbol{X}, t_n^+) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_n^-)) \right) \, dV \\ & - \Delta t_n \int_V \boldsymbol{W}_{,i}^h(\boldsymbol{X}, t_{n+1}^-) \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-)) \, dV \\ & + \Delta t_n \int_V \boldsymbol{W}^h(\boldsymbol{X}, t_{n+1}^-) \cdot \boldsymbol{Z}(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-)) \, dV \\ & + \Delta t_n \left(\int_{S_n^g} \boldsymbol{W}^h \cdot \boldsymbol{F}_i \left(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1}^-) \right) N_i \, dS + \int_{S_n^h} \boldsymbol{W}^h \cdot \boldsymbol{H} \, dS \right) \\ & + \mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h) + \mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) \end{aligned} \tag{4.18}$$

where the Stabilization term $\mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ and the Discontinuity Capturing operator $\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h)$ are both linear in the first argument, and nonlinear in the second.

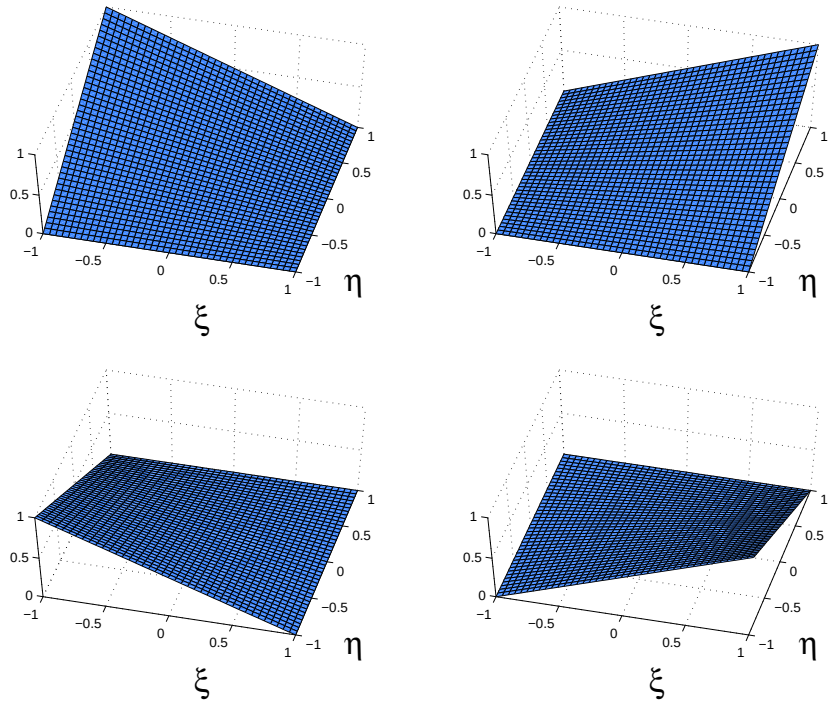

Figure 4.5: Local trial functions for the second-order time integrator, in the one-dimensional case. ξ is the local space coordinate, while η is the local time coordinate.

4.3.4 Second-order formulation

In terms of function spaces, we will assume that the trial function space $\mathcal{S}_n^h$ is given by the piecewise-linear, continuous functions on $P = S \times]0, T[$, while the test function space $\mathcal{V}_n^h$ will be given by functions that are continuous piecewise-linear in space and discontinuous, piecewise-constant in time (see Fig. 4.5 for details in the one-dimensional case). Therefore:

$$\mathcal{S}_n^h = \left\{ \boldsymbol{V}^h : \boldsymbol{V}^h \in (C^0(Q))^m, \boldsymbol{V}^h\big|_{Q_n^e} \in (\mathcal{P}_1(Q_n^e))^m, \boldsymbol{V}^h = \boldsymbol{g}(t) \text{ on } P_n^g \right\} \tag{4.19}$$

$$\begin{aligned} \mathcal{V}_n^h = \big\{ & \boldsymbol{W}^h : \boldsymbol{W}^h\big|_{V^e} \in (C^0(V))^m, \\ & \boldsymbol{W}^h\big|_{Q_n^e} \in (\mathcal{P}_1(V^e) \times \mathcal{P}_0(I_n))^m, \boldsymbol{V}^h = \boldsymbol{0} \text{ on } P_n^g \big\} \end{aligned} \tag{4.20}$$

with $m = 2n_{\mathrm{d}} + 1, \quad n_{\mathrm{d}} \in \{1, 2, 3\}$.

Equation (4.13) reduces to

$$
\begin{aligned}
0 \;=\; & \int_V \boldsymbol{W}^h(\boldsymbol{X}, t_{n+1}^-) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_{n+1})) - \boldsymbol{W}^h(\boldsymbol{X}, t_n^+) \cdot \boldsymbol{U}(\boldsymbol{Y}^h(\boldsymbol{X}, t_n)) \;\; dV \\
& + \int_{Q_n} \left(-\boldsymbol{W}^h_{,i} \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h) + \boldsymbol{W}^h \cdot \boldsymbol{Z}(\boldsymbol{Y}^h) \right) \;\; dQ \\
& + \int_{P_n^g} \boldsymbol{W}^h \cdot \boldsymbol{F}_i(\boldsymbol{Y}^h) N_i dP + \int_{P_n^h} \boldsymbol{W}^h \cdot \boldsymbol{H} \; dP \\
& + \mathcal{SUPG}(\boldsymbol{W}^h, \boldsymbol{Y}^h) + \mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h)
\end{aligned} \tag{4.21}
$$

where this time, since $\boldsymbol{Y}^h$ is continuous in space and time, there is no need to distinguish between positive and negative limit values for a certain instant in time.

Chapter 5

SUPG Stabilization

This chapter is devoted to the implementation of the SUPG stabilization. Stabilization is based upon a local linearized analysis, and provides a way to circumvent the intrinsic instability of a Galerkin formulation for linear-in-space trial/test functions. The stabilization of hyperbolic system of conservation laws is very well established nowadays and one can refer to the paper by Shakib, Hughes and Johan [64] for an exhaustive review.

It is also documented in the literature that SUPG stabilization prevents the occurrence of *hourglass* and *locking* phenomena, providing a stable framework for further development of discontinuity capturing operators.

The sketch of a multiscale analysis will also be presented in order to introduce a few important observations about the nature of the so-called acoustic instabilities. It was felt that the details of the calculations, performed according to the framework in [34], were not crucial for the thorough understanding of the ideas proposed, and that instead, a simple outline of the main steps involved would have been more beneficial.

5.1 A multiscale perspective on instabilities

Let us start from the variational form of the Lagrangian hydrodynamics equations (4.11), in which we assume, at the abstract level, that it is possible to have full knowledge about the exact solution $\boldsymbol{Y} \in \mathcal{S}$, where $\mathcal{S}$ is the abstract counterpart of the finite dimensional space $\mathcal{S}_n^h$ defining the numerical solution $\boldsymbol{Y}^h$ (introduced in chapter 4).

We will also define $\boldsymbol{W} \in \mathcal{V}$, where again, $\mathcal{V}$ is the abstract counterpart of $\mathcal{V}_n^h$. Hence, (4.11) reduces to:

$$\mathcal{B}(\boldsymbol{W}, \boldsymbol{Y}) \;=\; \mathcal{F}(\boldsymbol{W}) \tag{5.1}$$

Notice that the SUPG and DC operators have not been included, since they will be later defined as functions of the residual, vanishing everywhere if $\boldsymbol{Y}$ is exact. In particular, the following derivations will show how the SUPG operator is derived. For the sake of simplicity, we will assume that no body forces or Neumann conditions are applied, so that $\mathcal{F}(\boldsymbol{W}) \equiv 0$, throughout.

Let us now decompose the solution into a *coarse-scale* component, or *mesh* solution $\boldsymbol{Y}^h \in \mathcal{S}_n^h$ (the component of $\boldsymbol{Y}$ resolved by the numerical mesh), and a *fine-scale* or *subgrid* component $\boldsymbol{Y}' \in \mathcal{S}'$, ($\mathcal{S}' = \mathcal{V}/\mathcal{V}_n^h$ is the complement of $\mathcal{V}_n^h$ to $\mathcal{V}$). Analogously, the test function $\boldsymbol{W}$ can be decomposed as $\boldsymbol{W} = \boldsymbol{W}^h + \boldsymbol{W}'$, $\boldsymbol{W}' \in \mathcal{V}_n' = \mathcal{V}/\mathcal{V}_n^h$. It is very reasonable to assume that $\mathcal{V}'$ and $\mathcal{V}_n^h$ are *linearly independent*, as well as $\mathcal{S}'$ and $\mathcal{S}_n^h$. This leads to a decomposition of the original problem into two subproblems, namely:

$$\mathcal{B}(\boldsymbol{W}^h, \boldsymbol{Y}^h + \boldsymbol{Y}') \;=\; 0 \qquad \text{(mesh-scale problem)} \tag{5.2}$$

$$\mathcal{B}(\boldsymbol{W}', \boldsymbol{Y}^h + \boldsymbol{Y}') \;=\; 0 \qquad \text{(subgrid-scale problem)} \tag{5.3}$$

The multiscale analysis approach as outlined in [34] would proceed by making a local linearization approximation for the subgrid-scale problem (5.3). On each element, the nonlinear operators are linearized about a local base solution (for example the average values of $\boldsymbol{Y}^h$ over each element). Linerization will allow to use the quasi-linear form of the Lagrangian hydrodynamics equations, once an advective operator $\mathcal{L}_{adv}$ and the residual $\boldsymbol{Res}$ are defined:

$$\mathcal{L}_{adv} \;=\; \boldsymbol{A}_0 \frac{\partial}{\partial t} + \boldsymbol{A}_i \frac{\partial}{\partial X_i} \tag{5.4}$$

$$\boldsymbol{Res} \;=\; \mathcal{L} = \boldsymbol{A}_0 \frac{\partial}{\partial t} + \boldsymbol{A}_i \frac{\partial}{\partial X_i} + \boldsymbol{C} \tag{5.5}$$

Here $1 \leq i \leq n_{\mathrm{d}}$, and the repeated index notation has been used. Equation (5.3) will then transform, after linearization, to:

$$(\boldsymbol{W}', \mathcal{L}\boldsymbol{Y}') \;\; = \;\; -(\boldsymbol{W}', \mathcal{L}\boldsymbol{Y}^h) = -(\boldsymbol{W}', \boldsymbol{Res}(\boldsymbol{Y}^h)) \tag{5.6}$$

where $(\cdot, \cdot)$ indicates the L^2 inner product over the space-time slab. Equation (5.6) has to be intended in a loose sense, since it does not include boundary terms, omitted for the sake of brevity. Formally, one can now invert (5.6) by means of an inverse integral operator $\mathcal{L}^{-1}$, involving a Green's function kernel $\boldsymbol{G}'$

$$\boldsymbol{Y}'|_{Q_n^e}(\boldsymbol{X}) \;\; = \;\; -\int_{Q_n^e} \boldsymbol{G}' \; \boldsymbol{Res}(\boldsymbol{Y}^h) \; dQ \tag{5.7}$$

Although this is only a formal step, it becomes apparent, after detailed derivations are carried out, that:

1. The Lagrangian hydrodynamic equations simplify, after linearization, to the wave equation. If the reader is not convinced, it is easy to verify the assertion in the case of compressible gas dynamics: it is well known that the linearization of the compressible Euler equations is given by the equations of acoustics, which are wave propagation equations. Hence, $\boldsymbol{G}'$ can be proved to be a typical wave propagation Green's function kernel.

2. The multiscale analysis confirms the argument of Kuropatenko regarding acoustic instabilities. In the limit of weak shocks, or, more appropriately, isentropic compressions, the solution is smooth enough to be amenable to local linearization, and the presented multiscale analysis applies.

3. Finally, the multiscale framework provides an alternative *cure* for the acoustic instabilities, based on Green's function concepts rather than simple scaling arguments, with potential for improved robustness with respect to Kuropatenko-type corrections for artificial viscosities. SUPG stabilization methods are based on such concepts.

5.2 General form of SUPG stabilization

In SUPG-type approaches, the following approximation of the local Green's function $\boldsymbol{G}'$ is adopted:

$$\begin{aligned} \boldsymbol{Y}'|_{Q_n^e}(\boldsymbol{X}) &= -\int_{Q_n^e} \boldsymbol{G}'(\boldsymbol{X},\tilde{\boldsymbol{X}})\ \boldsymbol{Res}(\boldsymbol{Y}^h(\tilde{\boldsymbol{X}}))\ dQ_{\tilde{X}} \\ &\approx -\int_{Q_n^e} \boldsymbol{\tau}(\boldsymbol{X})\ \delta(\tilde{\boldsymbol{X}}-\boldsymbol{X})\ \boldsymbol{Res}(\boldsymbol{Y}^h(\tilde{\boldsymbol{X}}))\ dQ_{\tilde{X}} \\ &= -\boldsymbol{\tau}(\boldsymbol{X})\ \boldsymbol{Res}(\boldsymbol{Y}^h(\boldsymbol{X})) \end{aligned} \tag{5.8}$$

where $\delta(\cdot)$ is the Dirac delta. In stabilized methods, the approximation for $\boldsymbol{Y}'|_{Q_n^e}$ is substituted back into the mesh-scale equation, to yield the following form of the stabilization term:

$$\mathcal{SUPG}(\boldsymbol{W}^h,\boldsymbol{Y}^h) = \sum_{e=1}^{(n_{\mathrm{el}})_n} \int_{Q_n^e} (\mathcal{L}_{adv}^* \boldsymbol{W}_h) \cdot \underbrace{\boldsymbol{\tau}\ \boldsymbol{Res}(\boldsymbol{Y}^h)}_{\approx \boldsymbol{Y}'|_{Q_n^e}} dQ \tag{5.9}$$

where $\mathcal{L}_{adv}^*$ is the adjoint of $\mathcal{L}_{adv}$, and $\boldsymbol{\tau}$ is a tensor.

In the present work, we use the definition of $\boldsymbol{\tau}$ as in the article by Shakib, Hughes and Johan [64], to which the reader can refer for the general framework and a complete list of references on the subject:

$$\boldsymbol{\tau} = \boldsymbol{A}_0^{-1}\left(\tilde{\boldsymbol{C}}^2 + \left(\frac{\partial \xi_0}{\partial t}\right)^2 \boldsymbol{I}_{(2n_{\mathrm{d}}+1)\times(2n_{\mathrm{d}}+1)} + \frac{\partial \xi_i}{\partial X_j}\frac{\partial \xi_i}{\partial X_k}\tilde{\boldsymbol{A}}_j\tilde{\boldsymbol{A}}_k\right)^{-1/2} \tag{5.10}$$

where $\tilde{\boldsymbol{A}}_j = \boldsymbol{A}_j\boldsymbol{A}_0^{-1}$, $\tilde{\boldsymbol{C}} = \boldsymbol{C}\boldsymbol{A}_0^{-1}$, ξ_0 is the element local time coordinate (relative to the parent domain in time), and ξ_i are the element local spatial coordinates. Note also that the SUPG stabilization, in the case of piecewise-constant in time test functions, is equivalent to the GLS and Multiscale (or adjoint) stabilization, since the time derivative of the test function vanishes.

5.3 Stabilization in the Lagrangian framework

In the framework of Lagrangian hydrodynamics it is possible to radically simplify the computations of the stabilization terms. It is easily seen by carrying over the calculation of stabilization terms by brute force (the one-dimensional example is instructive in this regard), that stabilization truly affects only the momentum and energy equations. The rationale is that the kinematic equations relating the rates of displacement to the velocities are actually ODEs in the degrees-of-freedom of the discrete solution, so that stabilization - peculiar to boundary value problems (BVPs) for PDEs - is not needed. If the reader is not convinced by this argument, it is sufficient to carry over the full derivation of stabilization terms, and to observe that, as a result of lengthy calculations, the *stabilized* kinematic equations equal their original version multiplied by a factor different from unity (1/2 in the one-dimensional case).

In a more practical approach, it is possible then to reformulate the structure of the matrix $\boldsymbol{\tau}$ as follows:

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{0}_{n_\mathrm{d} \times n_\mathrm{d}} & \mathbf{0}_{n_\mathrm{d} \times (n_\mathrm{d}+1)} \\ \mathbf{0}_{(n_\mathrm{d}+1) \times n_\mathrm{d}} & \hat{\boldsymbol{\tau}}_{(n_\mathrm{d}+1) \times (n_\mathrm{d}+1)} \end{bmatrix} \tag{5.11}$$

As it is easily realized, in the case of one, two and three dimensions in space, instead of computing (either numerically or algebraically before hand) a $\boldsymbol{\tau}$ tensor of size 3×3, 5×5, or 7×7, it is sufficient to compute just the $\hat{\boldsymbol{\tau}}$ tensor of size 2×2, 3×3, or 4×4, respectively.

5.4 One-dimensional gas dynamic case

The one-dimensional case for an ideal gas is an interesting application of the definition (5.10) of $\boldsymbol{\tau}$: We retain just the lower right 2×2 blocks of the matrices defined in (2.58), and we denote those blocks by a "hat", according to the definition of $\hat{\boldsymbol{\tau}}$:

$$\hat{\boldsymbol{A}}_0 = \begin{bmatrix} \rho_0 & 0 \\ \rho_0 v & \frac{J}{\gamma - 1} \end{bmatrix} \quad \hat{\boldsymbol{A}}_1 = \begin{bmatrix} 0 & 1 \\ \frac{\gamma}{\gamma - 1} p & v \end{bmatrix} \tag{5.12}$$

Note also that the lower right block of $\boldsymbol{C}$ is zero, so that we do not need to account for it. Now,

$$\frac{\partial \xi_i}{\partial X_j}\frac{\partial \xi_i}{\partial X_k}\hat{\boldsymbol{A}}_j\hat{\boldsymbol{A}}_k = \left(\frac{2}{\Delta X}\right)^2 \hat{\boldsymbol{A}}_1{}^2 \tag{5.13}$$

and the computation of $\hat{\boldsymbol{A}}_1{}^2$ yields:

$$\begin{aligned}
\tilde{\hat{\boldsymbol{A}}}_1 = \hat{\boldsymbol{A}}_1\hat{\boldsymbol{A}}_0{}^{-1} &= \begin{bmatrix} 0 & 1 \\ \frac{\gamma}{\gamma-1}p & v \end{bmatrix} \begin{bmatrix} \rho_0 & 0 \\ \rho_0 v & \frac{J}{\gamma-1} \end{bmatrix}^{-1} \\
&= \begin{bmatrix} 0 & 1 \\ \frac{\gamma}{\gamma-1}p & v \end{bmatrix} \begin{bmatrix} \frac{1}{\rho_0} & 0 \\ -\frac{\gamma-1}{J}v & \frac{\gamma-1}{J} \end{bmatrix} \\
&= \begin{bmatrix} -\frac{\gamma-1}{J}v & \frac{\gamma-1}{J} \\ \frac{\gamma}{\gamma-1}\frac{p}{\rho_0} - \frac{\gamma-1}{J}v^2 & \frac{\gamma-1}{J}v \end{bmatrix}
\end{aligned} \tag{5.14}$$

$$\tilde{\hat{\boldsymbol{A}}}_1^2 = \begin{bmatrix} \frac{\gamma p}{\rho_0 J} & 0 \\ 0 & \frac{\gamma p}{\rho_0 J} \end{bmatrix} = \left(\frac{c_s}{J}\right)^2 \boldsymbol{I}_{2\times 2} \tag{5.15}$$

with $c_s = \sqrt{\frac{\gamma p}{\rho}} = \sqrt{\frac{\gamma p J}{\rho_0}}$. It is important to realize that the form of the SUPG stabilization is dependent on the function spaces adopted, and in particular on the time-integration strategy.

5.4.1 First-order time integrator

Noticing that for constant-in-time discontinuous test/trial functions, $\frac{\partial \xi_0}{\partial t} \equiv 0$, one can easily see that (5.10) simplifies as:

$$\begin{aligned}
\hat{\boldsymbol{\tau}} &= \hat{\boldsymbol{A}}_0{}^{-1}\left(\left(\frac{2\ c_s}{J\ \Delta X}\right)^2 \boldsymbol{I}_{2\times 2}\right)^{-1/2} \\
&= \frac{J\ \Delta X}{2\ c_s}\hat{\boldsymbol{A}}_0{}^{-1} = \Delta t \frac{J\ \Delta X}{2\ c_s \Delta t}\hat{\boldsymbol{A}}_0{}^{-1} = \frac{\Delta t}{2\alpha}\hat{\boldsymbol{A}}_0{}^{-1} \\
&= \frac{\Delta t}{2\alpha}\begin{bmatrix} \frac{1}{\rho_0} & 0 \\ -\frac{\gamma-1}{J}v & \frac{\gamma-1}{J} \end{bmatrix}
\end{aligned} \tag{5.16}$$

where $\alpha = \frac{c_s \Delta t}{J \Delta X}$, is the mesh Courant number.

5.4.2 Second-order time integrator

For the second order case, $\frac{\partial \xi_0}{\partial t} = \frac{2}{\Delta t}$ and

$$\begin{aligned}
\hat{\boldsymbol{\tau}} &= \hat{\boldsymbol{A}}_0{}^{-1} \left(\left(\frac{2}{\Delta t} \right)^2 \boldsymbol{I}_{2\times 2} + \left(\frac{2\ c_s}{J\ \Delta X} \right)^2 \boldsymbol{I}_{2\times 2} \right)^{-1/2} \\
&= \frac{\Delta t/2}{\sqrt{1+\alpha^2}} \hat{\boldsymbol{A}}_0{}^{-1} \\
&= \frac{\Delta t}{2\sqrt{1+\alpha^2}} \begin{bmatrix} \frac{1}{\rho_0} & 0 \\ -\frac{\gamma-1}{J} v & \frac{\gamma-1}{J} \end{bmatrix}
\end{aligned} \tag{5.17}$$

An alternative definition of $\hat{\boldsymbol{\tau}}$ is:

$$\hat{\boldsymbol{\tau}} = \hat{\boldsymbol{A}}_0{}^{-1} \left(|\tilde{\hat{\boldsymbol{C}}}| + \frac{2}{\Delta t} \boldsymbol{I}_{2\times 2} + \frac{2\ c_s}{J\ \Delta X} |\tilde{\hat{\boldsymbol{A}}}_1| \right)^{-1} \tag{5.18}$$

where, in the present case, $\tilde{\hat{\boldsymbol{C}}} = 0$. Given the hyperbolicity of the system, $\hat{\boldsymbol{A}}_1$ has real eigenvalues with linearly independent eigenvectors, so that equation (5.18) can be computed by an eigenvalue expansion of $\hat{\boldsymbol{A}}_1$, as follows:

$$\tilde{\hat{\boldsymbol{A}}}_1 = \boldsymbol{Q}\ \boldsymbol{\Lambda}\ \boldsymbol{Q}^{-1} \tag{5.19}$$

where

$$\tilde{\hat{\boldsymbol{A}}}_1 = \boldsymbol{Q}\ \boldsymbol{\Lambda}\ \boldsymbol{Q}^{-1} \tag{5.20}$$

$$\boldsymbol{\Lambda} = \begin{bmatrix} -\sqrt{\frac{\gamma p}{\rho_0 J}} & 0 \\ 0 & \sqrt{\frac{\gamma p}{\rho_0 J}} \end{bmatrix} = \begin{bmatrix} -\frac{c_s}{J} & 0 \\ 0 & \frac{c_s}{J} \end{bmatrix} \tag{5.21}$$

$$\boldsymbol{Q} = \begin{bmatrix} 1 & 1 \\ v - \frac{\sqrt{\frac{\gamma p J}{\rho_0}}}{\gamma-1} & v + \frac{\sqrt{\frac{\gamma p J}{\rho_0}}}{\gamma-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ v - \frac{c_s}{\gamma-1} & v + \frac{c_s}{\gamma-1} \end{bmatrix} \tag{5.22}$$

It is then easy to check that $|\hat{\tilde{\boldsymbol{A}}}_1| = \boldsymbol{Q}\ |\boldsymbol{\Lambda}|\ \boldsymbol{Q}^{-1} = \boldsymbol{Q}\ c_s/J\boldsymbol{I}_{2\times 2}\ \boldsymbol{Q}^{-1} = c_s/J\boldsymbol{I}_{2\times 2}$, so that, finally,

$$\hat{\boldsymbol{\tau}} \;=\; \frac{\Delta t/2}{1+\alpha}\hat{\boldsymbol{A}}_0{}^{-1} = \frac{\Delta t}{2(1+\alpha)}\begin{bmatrix} \frac{1}{\rho_0} & 0 \\ -\frac{\gamma-1}{J}v & \frac{\gamma-1}{J} \end{bmatrix} \tag{5.23}$$

Definitions (5.17) and (5.23) have been tested and compared with one another, leading to very similar results. Definition (5.17) produce somewhat less stabilization and is used in section 8, where numerical experiments are presented.

5.5 Two-dimensional hydrodynamic case

In the multidimensional case, the computation of the stabilization parameter is more involved, due to the fact that evaluation of the negative square root of a $(n_{\mathrm{d}}+1)\times(n_{\mathrm{d}}+1)$ tensor is needed. However, in the case of two spatial dimensions, there is a very elegant derivation due to Franca [16] for a 3×3-matrix which addresses this issue. The algorithm is presented in appendix A, and for more details the reader can refer to the original paper [16]. Therefore, the two-dimensional tensor $\hat{\boldsymbol{\tau}}$ can be computed as follows:

1. Compute the matrix:

$$\boldsymbol{B} = \hat{\tilde{\boldsymbol{C}}}^{\,2} + \left(\frac{\partial \xi_0}{\partial t}\right)^2 \boldsymbol{I}_{3\times 3} + \sum_{i,j,k=1}^{2} \frac{\partial \xi_i}{\partial X_j}\frac{\partial \xi_i}{\partial X_k}\hat{\tilde{\boldsymbol{A}}}_j\hat{\tilde{\boldsymbol{A}}}_k \tag{5.24}$$

   where $\hat{\tilde{\boldsymbol{C}}} = \boldsymbol{0}$.

2. Use Franca's algorithm to compute $\boldsymbol{B}^{-1/2}$.

3. Compute:

$$\hat{\boldsymbol{\tau}} = \hat{\boldsymbol{A}}_0{}^{-1}\boldsymbol{B}^{-1/2} \tag{5.25}$$

For the more challenging three-dimensional case, analogous techniques can be used to compute $\hat{\boldsymbol{\tau}}$, some of which are presented by Shakib, Hughes and Johan [64].

Chapter 6

Multiscale Discontinuity Capturing

The design of the Discontinuity Capturing (DC) operator is essential to the overall performance of the numerical method. Shocks must be smoothed in a band of possibly no more than 3-4 elements across the discontinuities.

A vast literature is available nowadays on discontinuity capturing operators for finite element methods (see [25] and [64] for a comprehensive survey of the most successful techniques). Such operators usually take the form of a residual-based artificial viscosity, rapidly vanishing when the solution is fairly smooth, as for isentropic compressions/expansions.

However, when a number of preliminary tests was performed, it was found that artificial viscosities scaled as suggested in [25] or [64] were not sufficiently robust to preserve reasonable accuracy for the energy and densities in blast- and implosion-type computations. The strongly unsteady character of Lagrangian hydrodynamics simulations may be the main reason why the scalings of previous literature, mostly designed for steady or quasi-steady flows, could not perform well in the most demanding transient simulations. In addition, it was found that the time-step limitations in the case of such DC operators were particularly severe, making them inefficient for transient analysis applications.

It appeared clear that a new design for the DC operator – specifically tailored for hydrodynamic flows – was needed, and the following sections will serve this purpose.

6.1 A few design requirements

Following the work by Caramana, Shashkov and Whalen [8], Christiensen [9], Noh [53] and Wilkins [67], design requirements can be summarized as follows:

1. *Dissipativity*: the DC operator must be a dissipative operator (i.e., it must dissipate energy), possibly – but not necessarily – defined through a viscosity operator.

2. *Galilean Invariance*: the DC operator must be invariant under Galilean transformations of coordinates.

3. *Rotation invariance*: it is advisable, especially in problems involving convergent flows, for the DC operator to be invariant under rotations.

4. The model should not introduce artificial dissipation along a surface of constant phase (a surface along which the magnitude of the velocity is constant, with a possible change in direction). An example of a surface of constant phase is a shock front, either planar or curved, as in the case of cylindrical or spherical symmetry.

5. The DC operator must vanish for expansions, since the solution is smooth enough to be accurately computed by the underlying numerical discretization. It is also advisable for the DC operator to vanish in a zone of uniform compression, for which a second-order code can represent the exact solution. Finally, it is of interest that the transition from zones in which the DC operator is inactive, to zones in which it is different from zero, be continuous.

The presented goals for a good DC operator will be achieved using concepts and techniques familiar in stabilized finite element methods for compressible flows.

6.2 A novel discontinuity capturing operator

In Lagrangian hydrodynamics, the natural way (also in terms of implementation) of expressing the DC operators is to do it in current configuration. A slightly different approach will be followed here, and the DC operator will be initially defined in the reference configuration, and only subsequently rephrased in the current configuration.

$$\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) = \sum_{e=1}^{(n_{\rm el})_n} \int_{(Q_n^e)_X} \nu_{DC}\; G^{ij}\; \boldsymbol{W}_{,X_i} \cdot \hat{\boldsymbol{A}}_0 \boldsymbol{Y}_{,X_j}\; dQ_X \tag{6.1}$$

$$\begin{aligned} \nu_{DC} &= 4C_1 \left(\frac{\sqrt{\boldsymbol{v}' \cdot [g^{ij}]^{-1}\; \boldsymbol{v}'}}{2} \right)^{\beta_B} \left((\boldsymbol{S} : \boldsymbol{S})^{1/2} \right)^{1-\beta_B}\; \chi\left\{ (\boldsymbol{b} \otimes \boldsymbol{b}) : \boldsymbol{S} < 0 \right\} \\ &= 4C_1 \underbrace{\left(\frac{\sqrt{\boldsymbol{v}' \cdot [g^{ij}]^{-1}\; \boldsymbol{v}'}}{2(\boldsymbol{S} : \boldsymbol{S})^{1/2}} \right)^{\beta_B}}_{\text{residual based}} \underbrace{(\boldsymbol{S} : \boldsymbol{S})^{1/2}\; \chi\left\{ (\boldsymbol{b} \otimes \boldsymbol{b}) : \boldsymbol{S} < 0 \right\}}_{\text{V.Neumann-Richtmyer}} \end{aligned} \tag{6.2}$$

The following definitions apply:

$$G^{ij} = \left[\frac{\partial \xi_k}{\partial X_i} \frac{\partial \xi_k}{\partial X_j} \right]^{-1} \qquad g^{ij} = \left[\frac{\partial \xi_k}{\partial x_i} \frac{\partial \xi_k}{\partial x_j} \right]^{-1} \tag{6.3}$$

$$\left\{ \begin{array}{c} \boldsymbol{v}' \\ p' \end{array} \right\} = \boldsymbol{Y}' = -\hat{\boldsymbol{\tau}}\; \hat{\boldsymbol{Res}}(\boldsymbol{Y}^h) = - \left[\begin{array}{c} \hat{\boldsymbol{\tau}}^{(mom)}_{n_{\rm d} \times (n_{\rm d}+1)} \\ \hat{\boldsymbol{\tau}}^{(energy)}_{1 \times (n_{\rm d}+1)} \end{array} \right] \left\{ \begin{array}{c} \boldsymbol{Res}^{(mom)}_{n_{\rm d} \times 1}(\boldsymbol{Y}^h) \\ \boldsymbol{Res}^{(energy)}_{1 \times 1}(\boldsymbol{Y}^h) \end{array} \right\} \tag{6.4}$$

$$\boldsymbol{b} = \dot{\boldsymbol{v}} / \|\dot{\boldsymbol{v}}\|_{l^2} \tag{6.5}$$

$$\begin{aligned} \boldsymbol{S} &= \frac{1}{2} \left((\mathbf{grad}\; \boldsymbol{v})^T + \mathbf{grad}\; \boldsymbol{v} \right) \\ &= \frac{1}{2} \left(\boldsymbol{F}^{-T} (\mathbf{Grad}\; \boldsymbol{v})^T + \mathbf{Grad}\; \boldsymbol{v}\; \boldsymbol{F}^{-1} \right) \end{aligned} \tag{6.6}$$

$$(\mathbf{grad}\; \phi)_i = \frac{\partial \phi}{\partial x_i} \tag{6.7}$$

$\beta_B \in [0, 1]$ is a parameter, which *blends* the effect of the classical Von Neumann-Richtmyer scaling with a new scaling based on the subgrid-scale velocity $\boldsymbol{v}'$, G^{ij} is the metric tensor map from the reference configuration to the parent domain, and g^{ij} is the metric tensor map from the current configuration to the parent domain. $\hat{\boldsymbol{A}}_0$ is equal to the Jacobian $\boldsymbol{A}_0$, where only the block relative to the momentum and energy equations is retained, using the same notation on the previous section on SUPG stabilization. Therefore, the DC operator is not applied to the kinematic equations of displacements, as we should expect.

6.2.1 Rationale behind the multiscale viscosity

Rewriting (6.1) in the current configuration, and assuming that all variables are discretized with piece-wise linear interpolation yields:

$$\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) \;=\; \sum_{e=1}^{(n_{\mathrm{el}})_n} \int_{(Q_n^e)_x} \nu_{DC}\; g^{ij}\; \boldsymbol{W}_{,x_i} \cdot J^{-1} \hat{\boldsymbol{A}}_0 \boldsymbol{Y}_{,x_j}\; dQ_x \tag{6.8}$$

The index x in $(Q_n^e)_x$ and dQ_x indicates that the space/time domain and its measure are defined in the current configuration rather than the reference (original) configuration. It is possible now to proceed with the analysis of the various terms present in expressions (6.8) and (6.2):

- The term $1/J\boldsymbol{A}_0$ can be thought of as a Jacobian in the current configuration, so that the effect of the DC operator is to smooth jumps in the momentum and total energy.

- By introducing the parameter β_B, the proposed viscosity is blending the classical Von Neumann scaling with a multiscale, residual-based scaling. This allows recovery of higher-order accuracy when mild compressions are happening in smooth regions of the flow, while oscillations are appropriately damped for strong shocks.

- $\chi_{\boldsymbol{b}}$ is a "switch", a characteristic function, triggered by negative values of the deviatoric strain rate along the direction of $\boldsymbol{b}$. It is not advisable to just use the negative values of the velocity divergence, since this alternative definition can cause loss of accuracy when compressions are due to streamline divergence rather than shocks (see e.g. Wilkins [67] for a detailed discussion).

- $\boldsymbol{b}$ is the unit vector in the direction of the acceleration $\dot{\boldsymbol{v}}$. An approximation to $\boldsymbol{b}$ can be given by:

$$\boldsymbol{b} \approx \frac{\mathbf{grad}\; p}{\|\mathbf{grad}\; p\|_{l^2}} \tag{6.9}$$

  whenever the acceleration is too expensive to compute.

- The term $\boldsymbol{v}'$ can be thought of as an approximate "sub-grid velocity". Let us consider the momentum equations. Correspondingly, there will be a $n_{\mathrm{d}} \times (n_{\mathrm{d}} + 1)$

block of $\hat{\boldsymbol{\tau}}$ (the n_{d} upper rows of $\hat{\boldsymbol{\tau}}$), namely $\hat{\boldsymbol{\tau}}^{(mom)}$, such that:

$$\boldsymbol{v}' \;=\; -\hat{\boldsymbol{\tau}}^{(mom)}\ \hat{\boldsymbol{Res}}(\boldsymbol{Y}^h) \tag{6.10}$$

The fact that $\boldsymbol{v}'$ is actually a velocity can be argued from detailed dimensional analysis applied to the structure of $\hat{\boldsymbol{\tau}}^{(mom)}$ and $\hat{\boldsymbol{Res}}(\boldsymbol{Y}^h)$.

This is the key idea introduced with the new nonlinear DC operator. The term $\frac{\boldsymbol{v}'^2}{2}$, corresponds to a sub-grid kinetic energy: each component v_i' is formed according to the multiscale analysis as in Hughes *et al.* [34].

$$\begin{aligned}
\boldsymbol{v}'(\boldsymbol{X},t) \;&=\; -\sum_{e=1}^{n_{\mathrm{el}}}\int_{Q_n^e} \mathrm{g}'(\boldsymbol{X},\tilde{\boldsymbol{X}};t,\tilde{t})\ \hat{\boldsymbol{Res}}(\boldsymbol{Y}^h(\tilde{\boldsymbol{X}},\tilde{t}))\ d\tilde{Q} && (6.11)\\
&\approx\; -\sum_{e=1}^{n_{\mathrm{el}}}\int_{Q_n^e} \hat{\boldsymbol{\tau}}^{(mom)}\delta(\boldsymbol{X}-\tilde{\boldsymbol{X}};t-\tilde{t})\ \hat{\boldsymbol{Res}}(\boldsymbol{Y}^h(\tilde{\boldsymbol{X}},\tilde{t}))\ d\tilde{Q}\\
&\approx\; -\hat{\boldsymbol{\tau}}^{(mom)}\ \hat{\boldsymbol{Res}}(\boldsymbol{Y}^h(\boldsymbol{X},t)) && (6.12)
\end{aligned}$$

from which (6.10) is easily deduced, as an approximation to (6.11). According to [34], $\hat{\boldsymbol{\tau}}$ is considered as an approximation to the integral operator providing the solution to the sub-grid scale problem over the space/time element Q_n^e. The kernel of such integral operator is the element Green's function

$$\{\boldsymbol{G}'(\boldsymbol{X},\tilde{\boldsymbol{X}};t,\tilde{t})\}^{(mom)} \approx \hat{\boldsymbol{\tau}}^{(mom)}\delta(\boldsymbol{X}-\tilde{\boldsymbol{X}};t-\tilde{t}) \tag{6.13}$$

acting on the residual of the numerical discretization, approximated by means of a localization hypothesis.

One of the most important consequences of the proposed definition, is that whenever the residual is zero (consider for example the case of uniform compression for a second-order finite element formulation) so is the DC operator. In addition, residual-based mechanisms will be very small in the vicinity of well-resolved flow features, retaining higher-order accuracy thereby. At the same time, the largest residuals (usually related to the occurrence of discontinuities in the flow) produce the largest amounts of dissipation.

Let us now analyze if the proposed model verifies the design requirements specified in section 6.1:

1. *Dissipativity*: by definition, the DC operator is a dissipative operator.

2. *Galilean/Rotation Invariance*: the proposed DC operator is based upon $\boldsymbol{S}$ (well-known to be invariant under Galilean/Rotation maps) and residuals, which are just the Euler equations in Lagrangian form, invariant in the first place.

3. In order not to introduce artificial dissipation along a surface of constant phase, $\boldsymbol{b}$ is chosen to be a unit vector in the direction of the acceleration $\dot{\boldsymbol{v}}$, as suggested by Wilkins [67]. A cruder approximation of the acceleration would be the pressure gradient $\mathbf{grad}\ p$. Because of the presence of the term $\boldsymbol{b}\otimes\boldsymbol{b}$, if one of the coordinates is aligned in the direction of the acceleration, the whole DC operator activates only if the gradient along $\boldsymbol{b}$ is negative.

4. The DC operator does vanish for expansions, according to the specification of the switch $\chi_{\boldsymbol{b}}$. Also, it vanishes for uniform compression, because in the case of a continuous piecewise-linear basis, linear fields can be represented exactly, so that the residual vanishes and so does the viscosity. Formally, it is not possible to prove continuous transition from zones in which the viscosity is inactive, to zones of compression. However, it is worthwhile mentioning that the residual structure is such that if we consider a moving front and the band of elements needed to resolve it numerically, the residual is going to peak at the center of the band and decrease toward the boundaries. Outside the band the viscosity is forced to zero by the switch, but, in the vicinity of the boundary of the band, the residual $\hat{\boldsymbol{Res}}(\boldsymbol{Y})$ might already have decreased its value by orders of magnitude with a "partial" recovery of the smooth transition requirement.

6.2.2 One-dimensional case

In the one-dimensional case, it is easily seen that:

$$\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) = \sum_{e=1}^{(n_{\rm el})_n} \int_{Q_n^e} \nu_{DC} \, \frac{\Delta X^2}{4} \, \boldsymbol{W}_{,X} \cdot \hat{\boldsymbol{A}}_0 \boldsymbol{Y}_{,X} \, dQ \tag{6.14}$$

$$\nu_{DC} = \frac{4C_1}{J} \left(\frac{|v'|}{\Delta X} \right)^{\beta_B} |\partial_X v|^{1-\beta_B} \, \chi \{\partial_x v < 0\} \tag{6.15}$$

where ΔX is the mesh spacing in the original configuration, and there is no ambiguity in requiring $\frac{\partial v}{\partial X} < 0$ to switch on the viscosity, since $\frac{\partial v}{\partial x} = J \frac{\partial v}{\partial X}$ with $J > 0$ (in this case the negative rate of strain in the acceleration direction or negative divergence condition are exactly the same). In terms of the current configuration, one has:

$$\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) = \sum_{e=1}^{(n_{\rm el})_n} \int_{(Q_n^e)_x} \nu_{DC} \, \frac{\Delta x^2}{4} \, \boldsymbol{W}_{,x} \cdot \frac{1}{J} \hat{\boldsymbol{A}}_0 \boldsymbol{Y}_{,x} \, dQ_x \tag{6.16}$$

$$\nu_{DC} = 4C_1 \left(\frac{|v'|}{\Delta x} \right)^{\beta_B} |\partial_x v|^{1-\beta_B} \, \chi \{\partial_x v < 0\} \tag{6.17}$$

yielding:

$$\mathcal{DC}(\boldsymbol{W}^h, \boldsymbol{Y}^h) = C_1 \sum_{e=1}^{(n_{\rm el})_n} \int_{(Q_n^e)_x} \Big\{ (|v'|/\Delta x)^{\beta_B} \, |\partial_x v|^{1-\beta_B} \, \Delta x^2 \, \boldsymbol{W}_{,x} \cdot J^{-1} \hat{\boldsymbol{A}}_0 \boldsymbol{Y}_{,x} \, \chi \{\partial_x v < 0\} \Big\} \, dQ_x \tag{6.18}$$

Notice now that the term $J^{-1}\hat{\boldsymbol{A}}_0$ is exactly the Jacobian of the transformation from the set of variables $\boldsymbol{Y}$ to the conservation variables in the current (deformed) configuration. To better understand the nature of the proposed DC operator, suppose to set $\beta_B = 0$, so that the artificial viscosity becomes independent of the residual. The proposed viscosity reduces then to a *modified* Von Neumann-Richtmyer viscosity *à la* Noh [53]. The similarity is striking, although the form of the proposed operator is not exactly equal to the one developed by Noh. In particular, in [53] the rationale for the beneficial effects of the introduction of an artificial heat flux in the energy equation is not supported by analytical considerations, but mainly relies on empirical considerations.

Instead, from the analysis presented herein it appears clear that:

- The form of the proposed artificial viscosity is designed so that it smoothes the solution in the direction of the *generalized gradient*.
- As a result, the DC operator acts on the full vector of conservation variables, and, in the energy equation (see, e.g. (2.58) and (2.79)), is composed of an artificial work and an artificial heat flux term.

Chapter 7

Predictor/Multi-corrector Algorithm

In order for the proposed approach to be competitive with the state of the art hydrocodes, explicit time integration is required. An explicit time integrator can be defined very naturally by means of a Newton nonlinear iterative solver.

More generally, each iteration of the Newton solver can be considered as a pass of a predictor/multi-corrector algorithm, in which the initial guess (predictor) is given by the solution vector of the discretized Lagrangian hydrodynamics equations at the previous time step. Because in general the tangent matrix for the Newton algorithm is non-diagonal, the predictor/multi-corrector results in an implicit scheme.

However, if mass lumping is used, or, equivalently, if nodal quadratures in space are used in the assembly of the tangent matrix, an explicit algorithm is generated, in which the matrix solves of the implicit counterpart are substituted by vector divisions.

A detailed explanation of the theoretical framework for this class of time-integration algorithms is presented in Hughes [27] p.562, while the paper by Shakib, Hughes and Johan [64] is very useful for an example of the implementation in the context of compressible flow computations in Eulerian coordinates.

In the following sections, a unifying abstract framework will be presented and applied to the first- and second-order algorithms introduced in chapter 4.

7.1 The nonlinear algebraic system of equations

The first step in the numerical solution of the Lagrangian hydrodynamic equations is the assembly of an algebraic system of equations. Because the PDEs for hydrodynamic flows are clearly nonlinear, an appropriate linearization of the (nonlinear) algebraic system of equations arising from the discretization is needed. The next section is devoted to the symbolic forming of the nonlinear algebraic system, while its linearization will be undertaken subsequently. It is now necessary to differentiate between the first- and second-order case, since a detailed representation of the nonlinear algebraic system is needed.

7.1.1 First-order time integrator

Both the solution $\boldsymbol{Y}^h$ and test function $\boldsymbol{W}^h$ are assumed continuous in space but discontinuous and piecewise constant in time. Only time-slabs given by a single element in time will be considered, so that the solution and test vector functions, for each time-slab, can be expressed as:

$$\boldsymbol{Y}^h(\boldsymbol{X},t) = \sum_{A=1}^{n_{\rm np}} \boldsymbol{N}_A(\boldsymbol{X})\boldsymbol{y}_{A;(n+1)} \tag{7.1}$$

$$\boldsymbol{W}^h(\boldsymbol{X},t) = \sum_{A=1}^{n_{\rm np}} \boldsymbol{N}_A(\boldsymbol{X})\boldsymbol{w}_{A;(n+1)} \tag{7.2}$$

with $\boldsymbol{X} \in V, t \in I_n =]t_n, t_{n+1}[$. In general, $\boldsymbol{y}_{A;(n)}$ is the global nodal unknown at time t_n and node A in the global node ordering. In what follows, global ordering will always be denoted by upper-case letters.

Also, let $(\mathcal{L}^h_{adv})_A$ and $(\mathcal{L}^h)_B$ be the discrete equivalent of (5.4) and (5.5), that is, recalling $\boldsymbol{W}^h_{,t} = \boldsymbol{Y}^h_{,t} = \boldsymbol{0}$:

$$(\mathcal{L}^h_{adv})_A = \boldsymbol{A}_i \boldsymbol{N}_{A,i} \tag{7.3}$$

$$(\mathcal{L}^h)_B = \boldsymbol{A}_i \boldsymbol{N}_{B,i} + \boldsymbol{C}\boldsymbol{N}_B \tag{7.4}$$

Substituting (7.1) and (7.2) into (4.18), and performing the trivial integration in time, one obtains

$$\begin{aligned} 0 \;=\; \sum_{A=1}^{n_{\rm np}} \boldsymbol{w}_A \cdot \Bigg\{ & \int_V \boldsymbol{N}_A \, \boldsymbol{U}\Bigg(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)}\Bigg) - \boldsymbol{N}_A \boldsymbol{U}\Bigg(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n)}\Bigg) \; dV \\ & -\Delta t_n \int_V \; \boldsymbol{N}_{A,i} \boldsymbol{F}_i\Bigg(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)}\Bigg) \; dV \\ & +\Delta t_n \int_V \; \boldsymbol{N}_A \boldsymbol{C} \; \sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)} \; dV \\ & +\Delta t_n \int_S \boldsymbol{N}_A \cdot \boldsymbol{F}_i\Bigg(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)}\Bigg) N_i \; dS \\ & +\Delta t_n \sum_{e=1}^{n_{\rm el}} \int_{V^e} (\mathcal{L}^h_{adv})_A \cdot \boldsymbol{\tau} \sum_{B=1}^{n_{\rm np}} (\mathcal{L}^h)_B \boldsymbol{y}_{B;(n+1)} \; dV_e \\ & +\mathcal{DC}\left(\boldsymbol{N}_A, \sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)}\right)\Bigg\} \end{aligned} \tag{7.5}$$

7.1.2 Second-order time integrator

The solution $\boldsymbol{Y}^h$ is continuous in space-time while the test function $\boldsymbol{W}^h$ is continuous in space but discontinuous and piecewise constant in time. On a single time-slab, the solution and test vector functions can be expressed as:

$$\boldsymbol{Y}^h(\boldsymbol{X},t) \;=\; \sum_{A=1}^{n_{\rm np}} \boldsymbol{N}_A(\boldsymbol{X}) \left(\pi_n^+(t)\boldsymbol{y}_{A;(n+1)} + \pi_n^-(t)\boldsymbol{y}_{A;(n)}\right) \tag{7.6}$$

$$\boldsymbol{W}^h(\boldsymbol{X},t) \;=\; \sum_{A=1}^{n_{\rm np}} \boldsymbol{N}_A(\boldsymbol{X}) \boldsymbol{w}_{A;(n+1)} \tag{7.7}$$

with $\boldsymbol{X} \in V, t \in I_n$, and

$$\pi_n^+(t) \;=\; \frac{t - t_n}{\Delta t} \tag{7.8}$$

$$\pi_n^-(t) \;=\; \frac{t_{n+1} - t}{\Delta t} \tag{7.9}$$

Also, let

$$(\mathcal{L}^h_{adv})_A = \boldsymbol{A}_i \boldsymbol{N}_{A,i} \tag{7.10}$$

$$(\mathcal{L}^h_{\pm})_B = \boldsymbol{A}_0 \boldsymbol{N}_B \pi^{\pm}_{,t} + \boldsymbol{A}_i \boldsymbol{N}_{B,i} \pi^{\pm} + \boldsymbol{C} \boldsymbol{N}_B \pi^{\pm} \tag{7.11}$$

Substitution of (7.6) and (7.7) into (4.21) yields

$$\begin{aligned} 0 = \sum_{A=1}^{n_{\rm np}} \boldsymbol{w}_A \cdot \Bigg\{ & \int_V \boldsymbol{N}_A \, \boldsymbol{U}\left(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n+1)}\right) - \boldsymbol{N}_A \boldsymbol{U}\left(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \boldsymbol{y}_{B;(n)}\right) \, dV \\ & - \int_{Q_n} \boldsymbol{N}_{A,_i} \boldsymbol{F}_i \left(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \left(\pi^+_n \boldsymbol{y}_{B;(n+1)} + \pi^-_n \boldsymbol{y}_{B;(n)}\right)\right) \, dQ \\ & + \int_{Q_n} \boldsymbol{N}_A \boldsymbol{C} \sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \left(\pi^+_n \boldsymbol{y}_{B;(n+1)} + \pi^-_n \boldsymbol{y}_{B;(n)}\right) \, dQ \\ & + \int_{P_n} \boldsymbol{N}_A \cdot \boldsymbol{F}_i \left(\sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \left(\pi^+_n \boldsymbol{y}_{B;(n+1)} + \pi^-_n \boldsymbol{y}_{B;(n)}\right)\right) N_i \, dP \\ & + \sum_{e=1}^{n_{\rm el}} \int_{Q^e_n} (\mathcal{L}^h_{adv})_A \cdot \boldsymbol{\tau} \sum_{B=1}^{n_{\rm np}} \left((\mathcal{L}^h_+)_B \boldsymbol{y}_{B;(n+1)} + (\mathcal{L}^h_-)_B \boldsymbol{y}_{B;(n)}\right) \, dQ^e_n \\ & + \mathcal{DC}\left(\boldsymbol{N}_A, \sum_{B=1}^{n_{\rm np}} \boldsymbol{N}_B \left(\pi^+_n \boldsymbol{y}_{B;(n+1)} + \pi^-_n \boldsymbol{y}_{B;(n)}\right)\right) \Bigg\} \end{aligned} \tag{7.12}$$

7.2 Newton iterative solver

Since on the interval I_n the nodal values $\boldsymbol{y}_{B;(n)}$ are known from the previous time-step, the system can be therefore abstractly represented as:

$$\boldsymbol{w} \cdot \tilde{\boldsymbol{G}}(\boldsymbol{y}_{(n+1)}; \boldsymbol{y}_{(n)}) = \boldsymbol{0} \tag{7.13}$$

and since (7.13) has to old for any $\boldsymbol{w}$, we obtain:

$$\tilde{\boldsymbol{G}}(\boldsymbol{y}_{(n+1)}; \boldsymbol{y}_{(n)}) = \boldsymbol{0} \tag{7.14}$$

Here a unifying notation is used, since $\tilde{\boldsymbol{G}}$ may originate either from the first- or second-order algorithm. The Newton iteration reads:

$$\tilde{\boldsymbol{G}}(\boldsymbol{y}_{(n+1)};\boldsymbol{y}_{(n)}) \approx \tilde{\boldsymbol{G}}(\boldsymbol{y}^{(i)};\boldsymbol{y}_{(n)}) + \partial_{\boldsymbol{y}}\tilde{\boldsymbol{G}}(\boldsymbol{y}^{(i)};\boldsymbol{y}_{(n)}) \;\; \delta\boldsymbol{y} = \boldsymbol{0} \tag{7.15}$$

with $\delta\boldsymbol{y} = (\boldsymbol{y}^{(i+1)} - \boldsymbol{y}^{(i)})$. Here the upper index (i) refers to an iterate of the Newton algorithm, while the lower index (n) refers to the solution at time t_n^-. Starting from (7.15) we can also accommodate for boundary conditions of Dirichlet type, by removing the rows of $\tilde{\boldsymbol{G}}$ corresponding to nodes on the Dirichlet part of the boundary, and moving to the right hand side products of the respective columns of $\partial_{\boldsymbol{y}}\tilde{\boldsymbol{G}}(\boldsymbol{y}^{(i)};\boldsymbol{y}_{(n)})$ multiplied by the corresponding known nodal values. Equation (7.15) can be therefore rewritten as:

$$\boldsymbol{M}^{*(i)} \;\delta\boldsymbol{y} = -\boldsymbol{R}^{(i)} \tag{7.16}$$

where the more convenient notation

$$\boldsymbol{M}^{*(i)} = \partial_{\boldsymbol{y}}\boldsymbol{G}(\boldsymbol{y}^{(i)};\boldsymbol{y}_{(n)}) \tag{7.17}$$

$$\boldsymbol{R}^{(i)} = \boldsymbol{G}(\boldsymbol{y}^{(i)};\boldsymbol{y}_{(n)}) \tag{7.18}$$

has been used, and $\boldsymbol{G}$ and $\partial_{\boldsymbol{y}}\boldsymbol{G}$ are obtained from $\tilde{\boldsymbol{G}}$ and $\partial_{\boldsymbol{y}}\tilde{\boldsymbol{G}}$, respectively, after Dirichlet boundary conditions have been accounted for.

The values of the tensor $\boldsymbol{\tau}$ and the DC operator are computed using the solution at the previous iterate: therefore they are said to be "frozen" or "lagged" at the previous iterate. The lagging practice simplifies the otherwise tedious and inefficient implementation of the SUPG and DC contributions to the tangent matrix. The number of terms to be lagged is the minimum amount of terms that renders the discrete operators linear in the nodal unknowns. As the Newton iteration converges, the distance between two successive iterates tends to zero, and so do the slight discrepancies between the true and lagged discrete operators.

7.3 Assembly

First, the assembly of $\boldsymbol{G}$ and $\boldsymbol{R}$ will be described, secondly, possible options for explicit/implicit and block-explicit/implicit time integration will be discussed. The assembled system of equations can be represented according to:

$$\boldsymbol{R}^{(i)} = \mathop{\mathsf{A}}_{e=1}^{n_{\text{el}}} \boldsymbol{R}_a^{e;(i)} \tag{7.19}$$

$$\boldsymbol{M}^{*(i)} = \mathop{\mathsf{A}}_{e=1}^{n_{\text{el}}} \boldsymbol{M}_{ab}^{e;(i)} \tag{7.20}$$

where A is the finite element assembly operator, and $\boldsymbol{R}_a^{e;(i)}$, $\boldsymbol{M}_{ab}^{e;(i)}$ are the element (therefore local) contributions to the residual and tangent matrix, respectively. Once more, a distinction between the first- and second- order method is required. As a note on implementation, the two methods have been implemented by just changing the element definition at the local level, preserving the entire data structure for the solution and the interactive Newton solver. This task has been eased by the fact that both algorithms require the update of only a single solution vector.

7.3.1 First-order time integrator

For the first-order case, it is more convenient to perform the trivial integration in time (recall that the test/trial functions are constant-in-time) so that the local (i)-iterate at t_{n+1} and the local solution at t_n can be defined as:

$$\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}) = \sum_{a=1}^{n_{\text{en}}} \boldsymbol{N}_a^e(\boldsymbol{X}) \boldsymbol{y}_a^{(i)} \tag{7.21}$$

$$\boldsymbol{Y}_{(n)}^{e}(\boldsymbol{X}) = \sum_{a=1}^{n_{\text{en}}} \boldsymbol{N}_a^e(\boldsymbol{X}) \boldsymbol{y}_{(n);a} \tag{7.22}$$

where $\boldsymbol{N}_a^e$ is the local shape function in space. The vector $\boldsymbol{y}_{a;(n)}$ represents the value of the solution at time t_n, while the vector $\boldsymbol{y}_a^{(i)}$ is the value of the Newton iterate of the solution, both evaluated at the locally-indexed node a. It will be always assumed that the lower case indexes denote local ordering. n_{en} is the number of element nodes in space (e.g. $n_{\text{en}} = 3$ for triangles, $n_{\text{en}} = 4$ for tetrahedra).

With the previous assumptions, $\boldsymbol{R}_a^{e;(i)}$, $\boldsymbol{M}_{ab}^{e;(i)}$ are defined as follows:

$$\begin{aligned}
\boldsymbol{R}_a^{e;(i)} &= \int_{V^e} \boldsymbol{N}_a^e \, \boldsymbol{U}(\boldsymbol{Y}^{e;(i)}) - \boldsymbol{N}_a^e \, \boldsymbol{U}(\boldsymbol{Y}_{(n)}^e) \, dV \\
&+\Delta t_n \int_{V^e} \left(-\boldsymbol{N}_{a,_i}^e \boldsymbol{F}_i(\boldsymbol{Y}^{e;(i)}) + \boldsymbol{N}_a^e \boldsymbol{C} \, \boldsymbol{Y}^{e;(i)} \right) \, dV \\
&+\Delta t_n \int_{S^e} \boldsymbol{N}_a^e \cdot \boldsymbol{F}_i(\boldsymbol{Y}^{e;(i)}) \, N_i \, dS \\
&+\Delta t_n \int_{V^e} \boldsymbol{A}_j \boldsymbol{N}_{a,_j}^e \cdot \boldsymbol{\tau} \left(\boldsymbol{A}_k \boldsymbol{Y}_{,k}^{e;(i)} + \boldsymbol{C} \boldsymbol{Y}^{e;(i)} \right) \, dV \\
&+\Delta t_n \, \mathcal{DC}_s^e(\boldsymbol{N}_A^e, \boldsymbol{Y}^{e;(i)})
\end{aligned} \tag{7.23}$$

$$\begin{aligned}
\boldsymbol{M}_{ab}^{e;(i)} &= \int_{V^e} \boldsymbol{N}_a^e \boldsymbol{A}_0 \boldsymbol{N}_b^e \, dV \\
&+\Delta t_n \int_{V^e} -\boldsymbol{N}_{a,_i}^e \boldsymbol{A}_i \boldsymbol{N}_b^e + \boldsymbol{N}_a^e \boldsymbol{C} \, \boldsymbol{N}_b^e \, dV \\
&+\Delta t_n \int_{S^e} \boldsymbol{N}_a^e \boldsymbol{A}_i \boldsymbol{N}_b^e \, N_i \, dS \\
&+\Delta t_n \int_{V^e} \boldsymbol{A}_j \boldsymbol{N}_{a,_j}^e \cdot \boldsymbol{\tau} \left(\boldsymbol{A}_k \boldsymbol{N}_{b,_k}^e + \boldsymbol{C} \boldsymbol{N}_b^e \right) \, dV \\
&+\Delta t_n \frac{\partial \mathcal{DC}_s^e}{\partial \boldsymbol{y}}(\boldsymbol{N}_a^e, \boldsymbol{N}_b^e; \boldsymbol{Y}^{e;(i)})
\end{aligned} \tag{7.24}$$

where, for convenience, the trivial integration in time has been performed and

$$\mathcal{DC}^e(\boldsymbol{N}_A^e, \boldsymbol{Y}^{e;(i)}) = \Delta t_n \, \mathcal{DC}_s^e(\boldsymbol{N}_A^e, \boldsymbol{Y}^{e;(i)}) \tag{7.25}$$

$$\frac{\partial \mathcal{DC}^e}{\partial \boldsymbol{y}}(\boldsymbol{N}_a^e, \boldsymbol{N}_b^e; \boldsymbol{Y}^{e;(i)}) = \Delta t_n \frac{\partial \mathcal{DC}_s^e}{\partial \boldsymbol{y}}(\boldsymbol{N}_a^e, \boldsymbol{N}_b^e; \boldsymbol{Y}^{e;(i)}) \tag{7.26}$$

7.3.2 Second-order time integrator

The local solution can be expressed – with a full space-time characterization – as:

$$\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t) = \sum_{a=1}^{n_{\text{en}}} \boldsymbol{N}_a^e(\boldsymbol{X}) \left(\pi_n^{e;+}(t) \boldsymbol{y}_a^{(i)} + \pi_n^{e;-}(t) \boldsymbol{y}_{a;(n)} \right) \tag{7.27}$$

where $\boldsymbol{N}_a^e$, analogously to the implementation of the first-order time integrator, is the local shape function in space, $\pi_n^{e;\pm}$ are the local trial shape function in time. The two-level

splitting of the solution in time is very useful in the implementation phase: the space-time elements are in fact prismatic in time, so that the implementation complexity for the space-time element is only marginally larger than for its purely spatial counterpart, even in the three-dimensional case. With the previous assumptions,

$$\begin{aligned}
\boldsymbol{R}_a^{e;(i)} &= \int_{V^e} \boldsymbol{N}_a^e \, \boldsymbol{U}\Big(\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t_{n+1})\Big) - \boldsymbol{N}_a^e \, \boldsymbol{U}\Big(\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t_n)\Big) \; dV \\
&+ \int_{Q_n^e} -\boldsymbol{N}_{a,i}^e \boldsymbol{F}_i\Big(\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t)\Big) + \boldsymbol{N}_a^e \boldsymbol{C} \; \boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t) \; dQ \\
&+ \int_{P_n^e} \boldsymbol{N}_a^e \cdot \boldsymbol{F}_i\Big(\boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t)\Big) \, N_i \; dP \\
&+ \int_{Q_n^e} \boldsymbol{A}_j \boldsymbol{N}_{a,j}^e \cdot \boldsymbol{\tau} \left(\boldsymbol{A}_0 \boldsymbol{Y}_{,t}^{e;(i)}(\boldsymbol{X}, t) + \boldsymbol{A}_k \boldsymbol{Y}_{,k}^{e;(i)}(\boldsymbol{X}, t) + \boldsymbol{C} \boldsymbol{Y}^{e;(i)}(\boldsymbol{X}, t) \right) \; dQ \\
&+ \mathcal{DC}^e(\boldsymbol{N}_A^e, \boldsymbol{Y}^{e;(i)})
\end{aligned} \tag{7.28}$$

$$\begin{aligned}
\boldsymbol{M}_{ab}^{e;(i)} &= \int_{V^e} \boldsymbol{N}_a^e \boldsymbol{A}_0 \boldsymbol{N}_b^e \; dV \\
&+ \int_{Q_n^e} -\boldsymbol{N}_{a,i}^e \boldsymbol{A}_i \boldsymbol{N}_b^e + \boldsymbol{N}_a^e \boldsymbol{C} \; \boldsymbol{N}_b^e \; dQ \\
&+ \int_{P_n^e} \boldsymbol{N}_a^e \boldsymbol{A}_i \boldsymbol{N}_b^e \, N_i \; dP \\
&+ \int_{Q_n^e} \boldsymbol{A}_j \boldsymbol{N}_{a,j}^e \cdot \boldsymbol{\tau} \left(\boldsymbol{A}_0 \boldsymbol{N}_b^e \pi_{n,t}^{e;+} + \boldsymbol{A}_k \boldsymbol{N}_{b,k}^e \pi_n^{e;+} + \boldsymbol{C} \boldsymbol{N}_b^e \pi_n^{e;+} \right) \; dQ \\
&+ \frac{\partial \mathcal{DC}^e}{\partial \boldsymbol{y}}(\boldsymbol{N}_a^e, \boldsymbol{N}_b^e, \pi_n^{e;+}; \boldsymbol{Y}^{e;(i)})
\end{aligned} \tag{7.29}$$

7.4 Predictor/multi-corrector outline

The whole time-step scheme is summarized in Table 7.1. Depending on how the matrix $\boldsymbol{M}^{*(i)}$ is formed, the algorithm presented can result in an explicit or implicit scheme. Although the form of the tangent matrix $\boldsymbol{M}^{*(i)}$ may change, the expression for the discrete residual vector $\boldsymbol{R}^{(i)}$ stays the same. Therefore, only the direction of search of the new iterate would change. As long as $\boldsymbol{M}^{*(i)}$ is not steering the solution out of the basin of convergence of the iterative solver, the residual $\boldsymbol{R}^{(i)}$ will always be driven to zero as the iteration proceeds.

Retreive loop parameters: n_{step}, $i_{\max}$
Initialize: set $\boldsymbol{y}_{(0)}$
For $n = 0, \ldots, n_{\text{step}}$ (*Time-step loop begins*)
  Predictor: $\boldsymbol{y}^{(0)} = \boldsymbol{y}_{(n)}$
  Set Δt (respecting the CFL condition)
  For $i = 0, \ldots, i_{\max} - 1$ (*Multi-corrector loop begins*)
    Form $\boldsymbol{R}^{(i)}(\boldsymbol{y}^{(i)}; \boldsymbol{y}_{(n)})$
    Form $\boldsymbol{M}^{*(i)}(y^{(i)})$
    Solve $\boldsymbol{M}^{*(i)} \delta \boldsymbol{y}^{(i)} = \boldsymbol{R}^{(i)}(\boldsymbol{y}^{(i)}; \boldsymbol{y}_{(n)})$
    Corrector: $\boldsymbol{y}^{(i+1)} = \boldsymbol{y}^{(i)} + \delta \boldsymbol{y}^{(i)}$
  End *(Multi-corrector loop ends)*
  Time update: $\boldsymbol{y}_{(n+1)} = \boldsymbol{y}^{i_{\max}}$
End *(Time-step loop ends)*
Exit

Table 7.1: Outline of the predictor-multicorrector algorithm.

7.4.1 Implicit algorithm

If $\boldsymbol{M}^{*(i)}$ is assembled from the full $\boldsymbol{M}^{e;(i)}_{ab}$ (i.e. the full expressions (7.24) or (7.29) are used, respectively), then the resulting scheme is implicit. Usually this means more flexibility in the choice of the time-step, since the scheme is unconditionally stable. Of course, accuracy would require time-steps close to the ones prescribed by the CFL condition. By an appropriate reordering of the nodal unknowns, (7.16) can be recast as:

$$\begin{bmatrix} \boldsymbol{M}^{*(i)}_{\boldsymbol{uu}} & \boldsymbol{0} & \boldsymbol{M}^{*(i)}_{\boldsymbol{uv}} \\ \boldsymbol{0} & \boldsymbol{M}^{*(i)}_{\boldsymbol{pp}} & \boldsymbol{M}^{*(i)}_{\boldsymbol{pv}} \\ \boldsymbol{0} & \boldsymbol{M}^{*(i)}_{\boldsymbol{vp}} & \boldsymbol{M}^{*(i)}_{\boldsymbol{vv}} \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{u}^{(i)} \\ \delta \boldsymbol{p}^{(i)} \\ \delta \boldsymbol{v}^{(i)} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^{(i)}_{\boldsymbol{u}} \\ \boldsymbol{R}^{(i)}_{\boldsymbol{p}} \\ \boldsymbol{R}^{(i)}_{\boldsymbol{v}} \end{bmatrix} \tag{7.30}$$

In what follows, the super-index (i) will be dropped, without loss of generality. Let

$$\boldsymbol{M}^{*}_{\boldsymbol{\Phi\Phi}} = \begin{bmatrix} \boldsymbol{M}^{*}_{\boldsymbol{pp}} & \boldsymbol{M}^{*}_{\boldsymbol{pv}} \\ \boldsymbol{M}^{*}_{\boldsymbol{vp}} & \boldsymbol{M}^{*}_{\boldsymbol{vv}} \end{bmatrix} \quad , \quad \delta \boldsymbol{\Phi} = \begin{bmatrix} \delta \boldsymbol{p} \\ \delta \boldsymbol{v} \end{bmatrix} \quad , \quad \boldsymbol{R}_{\boldsymbol{\Phi}} = \begin{bmatrix} \boldsymbol{R}_{\boldsymbol{p}} \\ \boldsymbol{R}_{\boldsymbol{v}} \end{bmatrix} \tag{7.31}$$

If block Gauss elimination is applied, it is possible to solve for the displacements after velocities and pressure have been updated, according to the sequence:

$$\delta\boldsymbol{\Phi} = -\boldsymbol{M}^{*^{-1}}_{\boldsymbol{\Phi}\boldsymbol{\Phi}}\boldsymbol{R}_{\boldsymbol{\Phi}} \tag{7.32}$$

$$\delta\boldsymbol{u} = -\boldsymbol{M}^{*^{-1}}_{\boldsymbol{u}\boldsymbol{u}}(\boldsymbol{R}_{\boldsymbol{u}} + \boldsymbol{M}^{*}_{\boldsymbol{u}\boldsymbol{v}}\delta\boldsymbol{v}) \tag{7.33}$$

It will be clear from the derivations in Section 7.5, that the update of the displacements never involves a true system solve, even in the implicit case, since the displacement equations have ODE rather than PDE character.

7.4.2 Explicit algorithm

One of the major issues in hydrocode technology is fast time integration. An explicit algorithm is now presented, with the specific goal of high performance for production-type implementations. The explicit scheme is stable, provided a CFL-type condition is satisfied. More details on the abstract framework are presented in the book by Hughes [27], p. 562. The tangent matrix $\boldsymbol{M}^{*^{(i)}}$ is non-diagonal, but – as it will be clear shortly – it is possible to step through a block Gauss elimination involving only diagonal matrix inversions, which are trivial vector divides.

As a side note, the structure of the algorithm closely relates to the typical time-marching schemes for hydrocodes. The key features of the proposed explicit approach can be enumerated as follows:

1. The tangent matrix $\boldsymbol{M}^{*^{(i)}}$ is assembled from a local matrix $\boldsymbol{M}^{e;(i)}_{ab}$ involving just the first of the terms in (7.24) or (7.29), namely:

$$\boldsymbol{M}^{e;(i)}_{ab} = \int_{V^e} \boldsymbol{N}^e_a \boldsymbol{A}_0 \boldsymbol{N}^e_b \, dV \tag{7.34}$$

   and in particular, the displacements are updated according to (7.32)–(7.33).

2. Specifically, $\boldsymbol{M}^{*^{(i)}}$ is assembled using nodal quadrature in space (row-sum mass lumping), yielding a diagonal tangent matrix: $\boldsymbol{M}^{*^{(i)}} \leftarrow \boldsymbol{M}^{L^{(i)}}$.

3. On the other hand, Gauss quadratures have been used for the algebraic residual $\boldsymbol{R}^{(i)}$ (two-point quadratures have been the choice in the one-dimensional case, while in the multidimensional case simplifications are possible, by means of tailored single-point quadratures).

4. The system is solved by block elimination with a further development of the strategy outlined in Section 7.4.2: first the increments of the velocities are computed, followed by the displacement and (finally) pressure increments.

Issues related to the non-diagonal structure of $\boldsymbol{A}_0$ are obviated by this approach, since all the blocks to be inverted are diagonal. The system to be solved is (the index i has been dropped):

$$\begin{bmatrix} \boldsymbol{M}^L_{\boldsymbol{uu}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M}^L_{\boldsymbol{pp}} & \boldsymbol{M}^L_{\boldsymbol{pv}} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{M}^L_{\boldsymbol{vv}} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{u}^{(i)} \\ \delta\boldsymbol{p}^{(i)} \\ \delta\boldsymbol{v}^{(i)} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^{(i)}_{\boldsymbol{u}} \\ \boldsymbol{R}^{(i)}_{\boldsymbol{p}} \\ \boldsymbol{R}^{(i)}_{\boldsymbol{v}} \end{bmatrix} \tag{7.35}$$

Assembly of (7.34) renders $\boldsymbol{M}^L$ an upper triangular block matrix (with diagonal blocks). The non-diagonal blocks $\boldsymbol{M}^L_{\boldsymbol{uv}}$, $\boldsymbol{M}^L_{\boldsymbol{pv}}$ originate from the assembly of the off-diagonal entries in the matrix $\boldsymbol{A}_0$, while contributions involving the $\boldsymbol{A}_i$'s are absent, by definition of $\boldsymbol{M}^L$. Backward block-Gaussian elimination yields

$$\delta\boldsymbol{v} = -\boldsymbol{M}^{L^{-1}}_{\boldsymbol{vv}} \boldsymbol{R}_{\boldsymbol{v}} \tag{7.36}$$

$$\begin{aligned} \delta\boldsymbol{u} &= -\boldsymbol{M}^{L^{-1}}_{\boldsymbol{uu}} \boldsymbol{R}_{\boldsymbol{u}} \\ &= -\boldsymbol{M}^{L^{-1}}_{\boldsymbol{uu}} \boldsymbol{R}_{\boldsymbol{u}} \end{aligned} \tag{7.37}$$

$$\begin{aligned} \delta\boldsymbol{p} &= -\boldsymbol{M}^{L^{-1}}_{\boldsymbol{pp}} (\boldsymbol{R}_{\boldsymbol{p}} + \boldsymbol{M}^L_{\boldsymbol{pv}} \boldsymbol{M}^{L^{-1}}_{\boldsymbol{vv}} \boldsymbol{R}_{\boldsymbol{v}}) \\ &= -\boldsymbol{M}^{L^{-1}}_{\boldsymbol{pp}} (\boldsymbol{R}_{\boldsymbol{p}} + \boldsymbol{M}^L_{\boldsymbol{pv}} \delta\boldsymbol{v}) \end{aligned} \tag{7.38}$$

where all the matrices to be inverted, due to lumping, are now diagonal. Note that $\boldsymbol{M}^L_{\boldsymbol{pp}}$ is a function of $\boldsymbol{u}$ and $\boldsymbol{v}$, already updated before the pressure stage in the block elimination. It is therefore possible to update also the entries of $\boldsymbol{M}^L_{\boldsymbol{pp}}$, before (trivially) inverting it. This might result in an increase in the iteration rate of convergence.

7.5 Time-integration strategy for displacements

This section is entirely devoted to explain how the displacements have to be properly updated. Distinctions will be made between the implicit and explicit cases, and within each case, between the first- and second-order sub-cases.

In conclusion, it will be observed that the original $[(2n_{\rm d}+1)\times n_{\rm np}]\times[(2n_{\rm d}+1)\times n_{\rm np}]$ global system solve for the nodal values of $\boldsymbol{u}_A$, $\boldsymbol{v}_A$, and $\boldsymbol{p}_A$ $(1 \le A \le n_{\rm np})$ can be decomposed into a $[(n_{\rm d}+1)\times n_{\rm np}]\times[(n_{\rm d}+1)\times n_{\rm np}]$ system solve for the nodal unknowns $\boldsymbol{v}_A$ and $\boldsymbol{p}_A$ followed by a simple $[n_{\rm d}\times n_{\rm np}]\times[n_{\rm d}\times n_{\rm np}]$ ODE update for $\boldsymbol{u}_A$.

7.5.1 Implicit algorithm

Let us recall the general space-time discretization:

$$\begin{aligned} 0 \;=\; & \int_V \boldsymbol{w}^h(\boldsymbol{X},t^-_{n+1})\cdot\boldsymbol{u}^h(\boldsymbol{X},t^-_{n+1})\;dV - \int_V \boldsymbol{w}^h(\boldsymbol{X},t^+_{n})\cdot\boldsymbol{u}^h(\boldsymbol{X},t^-_{n})\;dV \\ & - \int_{Q_n} \boldsymbol{w}^h(\boldsymbol{X},t)\cdot\boldsymbol{v}^h(\boldsymbol{X},t)\;dQ \end{aligned} \tag{7.39}$$

First-order time integrator

Starting from (7.5), and extracting the displacements equations, one obtains:

$$\begin{aligned} 0 \;=\; & \sum_{A,B=1}^{n_{\rm np}} \boldsymbol{w}_A\cdot\Big\{\int_V \boldsymbol{N}_A\,\boldsymbol{N}_B\boldsymbol{u}_{B;(n+1)} - \boldsymbol{N}_A\boldsymbol{N}_B\boldsymbol{u}_{B;(n)}\;dV \\ & \qquad +\Delta t_n\int_V \boldsymbol{N}_A\boldsymbol{N}_B\boldsymbol{v}_{B;(n+1)}\;dV\Big\} \end{aligned} \tag{7.40}$$

and, in a more compact notation,

$$\boldsymbol{M}_s\boldsymbol{u}^{(i+1)} - \boldsymbol{M}_s\boldsymbol{u}_n = \Delta t_n\boldsymbol{M}_s\boldsymbol{v}^{(i+1)} \tag{7.41}$$

where $\boldsymbol{u}^{(i+1)}/\boldsymbol{v}^{(i+1)}$ and $\boldsymbol{u}_n/\boldsymbol{v}_n$ are used to denote the nodal displacements/velocities at the iterate i or time t_n, respectively, and$\boldsymbol{M}_s$ is the spatial mass matrix for continuous, piecewise linear functions, that is,

$$\boldsymbol{M}_s \;=\; \int_V \boldsymbol{N}_A\boldsymbol{N}_B\;dV \tag{7.42}$$

Note that $\boldsymbol{v}^{(i+1)}$ is a known quantity, since it has been computed from the momentum/energy block of the system. Differentiating (7.41), according to the Newton paradigm, leads to:

$$\boldsymbol{M}_s \left(\delta\boldsymbol{u}^{(i)} - \Delta t_n \delta\boldsymbol{v}^{(i)}\right) \;=\; -\boldsymbol{M}_s \left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \Delta t_n \boldsymbol{v}^{(i)}\right) \tag{7.43}$$

Since no mass lumping is applied on the right hand side (RHS), the mass matrices (identical for velocities and displacements due to the choice of function spaces) readily simplify. Furthermore, when the displacements are updated, the new velocity increments are already available, and can be considered as known quantities. Therefore, the update of the nodal displacements does not involve any matrix solve, and is given by the classical implicit Euler algorithm for ODEs:

$$\delta\boldsymbol{u}^{(i)} \;=\; \Delta t_n \delta\boldsymbol{v}^{(i)} - \left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \Delta t_n \boldsymbol{v}^{(i)}\right) \tag{7.44}$$

Second-order time integrator

For the second-order case, the procedure closely follows the previous derivation. Note however that, for the second-order case, the trial functions are not constant in time over single-element time slabs any more. A simplified version of (7.12) is obtained for the displacements:

$$\begin{aligned} 0 \;=\; \sum_{A,B=1}^{n_{\rm np}} \boldsymbol{w}_A \cdot \Big\{ \int_V \boldsymbol{N}_A \, \boldsymbol{N}_B \boldsymbol{u}_{B;(n+1)} - \boldsymbol{N}_A \boldsymbol{N}_B \boldsymbol{u}_{B;(n)} \; dV \\ + \int_{Q_n} \boldsymbol{N}_A \, \boldsymbol{N}_B \left(\pi_n^+ \boldsymbol{v}_{B;(n+1)} + \pi_n^- \boldsymbol{v}_{B;(n)}\right) \; dQ \Big\} \end{aligned} \tag{7.45}$$

or, by performing the integration in space and time, and using a simplified notation,

$$\boldsymbol{M}_s \boldsymbol{u}^{(i+1)} - \boldsymbol{M}_s \boldsymbol{u}_n = \frac{\Delta t_n}{2} \left(\boldsymbol{M}_s \boldsymbol{v}^{(i+1)} + \boldsymbol{M}_s \boldsymbol{v}_n\right) \tag{7.46}$$

or, in incremental form,

$$\boldsymbol{M}_s \left(\delta\boldsymbol{u}^{(i)} - \frac{\Delta t_n}{2} \delta\boldsymbol{v}^{(i)}\right) \;=\; -\boldsymbol{M}_s \left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \frac{\Delta t_n}{2} \left(\boldsymbol{v}^{(i)} + \boldsymbol{v}_n\right)\right) \tag{7.47}$$

Once again, $\boldsymbol{M}_s$ readily simplifies, and comparing with (4.8), it is easily recognized the trapezoidal rule:

$$\boldsymbol{u}^{(i+1)} = \boldsymbol{u}_n + \frac{\Delta t_n}{2}\left(\boldsymbol{v}^{(i+1)} + \boldsymbol{v}_n\right) \tag{7.48}$$

or

$$\delta\boldsymbol{u}^{(i)} = \frac{\Delta t_n}{2}\delta\boldsymbol{v}^{(i)} - \left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \frac{\Delta t_n}{2}\left(\boldsymbol{v}^{(i)} + \boldsymbol{v}_n\right)\right) \tag{7.49}$$

7.5.2 Explicit algorithm

For the explicit method, the matrices on the left hand side (LHS) of (7.43) and (7.47) are lumped, so that – strictly speaking – they cannot simplify, although there are still no matrix solves to perform. The results for first- and second-order time integrators are briefly stated below.

First order time integrator

$$\delta\boldsymbol{u}^{(i)} \;=\; -(\boldsymbol{M}_s^L)^{-1}\boldsymbol{M}_s\left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \Delta t_n\boldsymbol{v}^{(i)}\right) \tag{7.50}$$

Second order time integrator

$$\delta\boldsymbol{u}^{(i)} = -(\boldsymbol{M}_s^L)^{-1}\boldsymbol{M}_s\left(\boldsymbol{u}^{(i)} - \boldsymbol{u}_n - \frac{\Delta t_n}{2}\left(\boldsymbol{v}^{(i)} + \boldsymbol{v}_n\right)\right) \tag{7.51}$$

Chapter 8

Numerical Experiments and Conclusions

An efficient and accurate method for compressible flows must be tested on two very important issues. The first issue is robustness under shock conditions (discontinuities must be properly captured, without dangerous over/under-shoots), the second is accuracy in regions where the solution is smooth.

In order to assess the performance of the Hydro-SUPG formulation, tests from one-dimensional gas dynamics have been performed, ranging from acoustic wave propagation studies to blast and implosion computations. The performance of the proposed approach is compared with a classical one-dimensional hydrocode formulation as presented in the introductory section 3.

Detailed analysis of the numerical test for one-dimensional gas dynamics shows that the Hydro-SUPG approach delivers comparable or superior accuracy with respect to a prototypical hydrocode implementation. In terms of computational cost – evaluated in terms of cumulative number of iterations to target time – the Hydro-SUPG method results in a more expensive algorithm, although still very competitive with the classical hydrocode implementations. Since the proposed method – still in its initial exploratory stage – is far from having been optimized, its potential for Lagrangian Hydrodynamics appears clear. In addition, SUPG approaches for hyperbolic system of conservation laws are very well suited for computations on multi-dimensional unstructured tetrahedral meshes, a combination very little explored by classical hydrocode approaches to date.

8.1 One-dimensional gas dynamics

The present section is devoted to the analysis of numerical experiments in the case of one-dimensional gas dynamics. A number of general considerations on the details of the simulations is presented first, followed by comments on the results.

8.1.1 General remarks

A point of departure in the discussion is presenting the overall solution update strategy, which can be summarized as follows:

1. The primitive variables $\boldsymbol{u}$, $\boldsymbol{v}$, p are solved for in the numerical computation.
2. The Jacobian determinant J is then computed from the displacement field $\boldsymbol{u}$.
3. The density ρ is post-processed using the mass conservation (2.8).
4. The internal energy e is post-processed using the equation of state (constitutive law) (2.21).

Note that energy and density are not directly used in the computation. Any time they appear in the variational form, their expressions in terms of the initial density ρ_0, the Jacobian J, the pressure p and the specific heat ratio γ are used. In this sense, they do not need to be specifically computed. However, in the post-processing phase of the results of a simulation, it is important to the analyst to represent these variables accurately.

8.1.2 On the accurate computation of density and energy

The initial density ρ_0 has been assumed piecewise-constant over elements and discontinuous from element to element. Since u is linear, recalling (2.8) and (2.33)-(2.34), J is discontinuous piecewise-constant. Specifically, J involves derivatives of u, and is therefore an intrinsically less accurate quantity (see, e.g., the plot for J on the bottom right corner of Fig. 8.8). Obviously, the degradation in accuracy migrates to e and ρ, in the form of node-to-node oscillations with point-wise errors up to 5–10%.

One way to post-process ρ retaining accuracy, is to first compute its *cell-centered* representation as $\rho = \rho_0 \; J$, and finally redefine it as a nodal quantity by means of an

averaging over co-volumes. With this procedure, the results – especially in shock-tube tests – underwent a dramatic improvement: this technique will be referred to as the *co-volume average post-processing*.

Other strategies are possible, and for this reason it is important to know what worked less accurately, as the assumption that all quantities – including ρ_0, ρ, e – are continuous, piecewise-linear.

Special care has also been taken in the post-processing of the internal energy e. It has been found beneficial to apply a smoothing operator to the vector of nodal degrees-of-freedom for the pressure, before using it to compute the internal energy. Namely:

$$\{p_{avg}\}_i = \sum_{j=1}^{n_{\rm np}} G_{ij} p_j \tag{8.1}$$

where $G_{ij} = (\text{meas}(cov_i))^{-1} M_{ij}$, M_{ij} is the mass matrix operator, and $\text{meas}(cov_i)$ is the average measure of the co-volume centered at node i (coincident, for one-dimensional, uniform meshes, to the length of the mesh interval). This type of *pressure post-processing* has been used throughout all simulations presented herein.

One point to be made is that accuracy of the method must not be confused with representation issues of the reconstructed energy and density fields, which never enter directly the computation. Although there is a certain degree of arbitrariness in the choice of the post-processing, the very good results that will be observed in the forthcoming sections show that the proposed reconstruction technique is very accurate.

8.1.3 Proper setting of the density initial condition

Having the pressure piecewise-linear and the initial density ρ_0 piecewise-constant posed some issues on how to setup the initial conditions. The strategy that proved the best for the purpose of oscillation-free results was to interpolate the exact initial density distribution with a continuous piecewise-linear distribution $\rho_0^{(nodal)}$, and average it subsequently over the cells (i.e., the elements), in order to retrieve a piecewise-constant distribution $\rho_0^{(cell)}$. This seems the most natural choice, since, taken the one-dimensional case as an example, if we consider an element where an initial discontinuity is located, the pressure would vary linearly between the left and right values and there would be ambiguity in

setting the element centered $\rho_0^{(cell)}$ equal to either the corresponding left or right values of $\rho_0^{(nodal)}$. By cell-averaging $\rho_0^{(nodal)}$, a better match was obtained between pressure and density in the initial condition. This technique is straightforward to generalize to the multi-dimensional case, using cell-averaging in multiple dimensions.

We add, as a *proviso*, that if the initial conditions are discontinuous, it is not advisable to L^2-project them onto the continuous finite element basis (different is the case if a discontinuous basis is used), since Gibbs phenomena may arise in the vicinity of discontinuities.

A simpler and more accurate approach would be just to define the initial nodal values as the nodal interpolant of the exact initial condition.

8.1.4 Optimization of the DC parameters

A number of tests has been performed initially on the proposed class of nonlinear viscosities, in order to assess their overall performance and tune their parameters.

In particular, it was observed that a blending parameter $\beta_B = 1/4$ yielded the best results for the second-order variant of the Hydro-SUPG scheme, while the choice $\beta_B = 1/2$ was better suited for the first-order variant (not presented here). Test performed are not conclusive in this regard, but for all the following numerical examples, the following choices were adopted:

First-order time integrator: $C_1 = 3$, $\beta_B = 1/2$. Several test showed that constants in the range $C_1 \in [1, 6]$ where delivering good performance.

Second-order time integrator: $C_1 = 3$, $\beta_B = 1/4$. Good performance was observed for $C_1 \in [1, 6]$.

One point to be made is that the viscosities were more sensitive to β_B than C_1: This was expected, since the former is changing the scaling of the viscosity, while the latter allows fine-tuning adjustments.

Typically, the tuning requires just a few simulations, and is usually much easier than for the case of Von Neumann-Richtmyer viscosities, due to the fact that the proposed approach is residual-based, hence very localized. Away from shock discontinuities, the

residual is almost nil, so that unless the parameters used are very far from the optimal range, good results are obtained.

In the following discussion, results obtained with a prototypical one-dimensional hydrocode (as detailed in section 3) will also be presented. In that case, the artificial viscosity has the form given by (3.31), with $c_{VN_R} = 1.5$, and $c_{LIN} = .06$, a very popular choice, as documented by Benson [3].

8.1.5 Explicit versus implicit

For all test presented, the *explicit* version of the code has been used, being the most probable candidate for large-scale computations, since no algebraic system solve is required at each iteration.

The implicit and explicit versions were compared keeping the number of iterations fixed, and the implicit proved superior for all test cases. Hence, showing only the results for the explicit version represents a conservative approach. As a general remark, provided the nonlinear solver converges, both implicit and explicit versions yield the same limit solution as the number of iterations tends to infinity. Therefore, differences can be appreciated only if a few nonlinear iterations are performed per time-step.

In terms of future research, it seems also viable the opportunity to implement an element-by-element implicit/explicit algorithm (see [27] p. 559 and [35] for details), since the lack of CFL restrictions in *implicit sub-domains* can allow a larger global time-step.

8.1.6 Physical versus "artificial" CFL condition

This section is the true core of the implementation strategy, and it is of capital importance for obtaining reliable, robust, and accurate simulations.

In a nutshell, when using artificial viscosities to construct DC operators, the numerics cannot distinguish whether such viscosities are a manifestation of a physical phenomenon or just a "stabilizing artifact".

Therefore, together with the standard definition of the Courant-Friedrichs-Levy (CFL) condition for advection, a diffusion CFL condition must be applied. Failing to account for artificial diffusion leads to "computational disasters", as preliminary simulations showed,

especially in the early times of computations, when the artificial viscosities typically dominate the time constraints.

To fix the notation let us define an advective global Courant number CFL_{adv}:

$$CFL_{adv} = \max_{1\leq j\leq n_{\rm el}} \alpha_j \tag{8.2}$$

where α_j corresponds to the parameter α (defined in section 5.4.2) relative to the element j. Let us also introduce the diffusive Courant number CFL_q:

$$CFL_q = \max_{1\leq j\leq n_{\rm el}} \left(\frac{2\nu_{DC_j}\Delta t}{\Delta X_j^2} \right) \tag{8.3}$$

The stability limit in the explicit version of the proposed algorithm is given by:

$$CFL = \max(CFL_{adv}, CFL_q) \leq 1 \tag{8.4}$$

Typically, the threshold 1 is never used, for safety reasons, and the combined CFL parameter is taken smaller than unity ($CFL \in [0.5, 0.9]$ is a common range of values). The CFL condition (8.4) is better understood when recast in terms of the maximum allowed time step:

$$\Delta t_{CFL_{adv}} = \min_{1\leq j\leq n_{\rm el}} \left(CFL_{adv} \frac{\Delta X_j J_j}{c_j} \right) \tag{8.5}$$

$$\Delta t_{CFL_q} = \min_{1\leq j\leq n_{\rm el}} \left(CFL_q \frac{2\Delta X_j^2}{\nu_{DC_j}} \right) \tag{8.6}$$

Clearly, the combined CFL condition yields:

$$\Delta t_{CFL_{max}} = \min(\Delta t_{CFL_{adv}}, \Delta t_{CFL_{adv}}) \tag{8.7}$$

In all the simulations of Riemann-type problems, for which the CFL restrictions are severe, the history of $\Delta t_{CFL_{adv}}$, Δt_{CFL_q}, and $\Delta t_{CFL_{max}}$ will be presented.

To conclude this section, it is now important to note what should be the properties of an efficient numerical algorithm based on artificial viscosity DC operators. In practice, the aim is to have limitations on the time-step only due to the true physics of the

problem. Therefore one would hope to have $\Delta t_{CFL_{max}} = \Delta t_{CFL_{adv}}$. In fact, whenever the restriction on the time step comes from the artificial diffusion, one is forced to pay the price of smaller time-steps in order to have a stable scheme. It will be shown in the extensive numerical experiments that the proposed DC operator yields diffusive CFL condition that are less restrictive than the advective CFL condition, with the exception of very few time-steps (usually less than ten) in the early stages of the simulation. In this sense, the proposed DC operator is practically optimal, since it stabilizes strong shocks without causing additional restrictions on the time-step limits. It will also be observed that this is not usually the case for the standard Von Neumann-Richtmyer viscosity.

8.1.7 Time integration strategy

In order to quantify the nonlinear solver convergence, it was found satisfactory to use a tolerance ϵ_p acting on the relative correction of the pressure iterate. In the proposed implementation, the Newton iteration proceeds as long as:

$$\frac{\sum_{A=1}^{n_{\text{np}}} |\delta p_A|}{\sum_{A=1}^{n_{\text{np}}} |p_A|} > \epsilon_p = 3 \times 10^{-4} \tag{8.8}$$

The number of iterations was fixed in certain situations, and left free to vary in others. The history of the number of iterations is presented for all Riemann problems tested. Typically, whenever new features appear in the solution (either in terms of the computed or post-processed variables), the number of iterations increases locally in time and reduces subsequently, as the numerics resolves the occurring features. As a simple example, if a new *plateau* is formed in the density or energy, the code will take more iterations in the early instants of its formation, while the number of iterations will decrease as soon as the plateau is captured by 5-10 elements.

Overall, the number of iterations is therefore reasonable, with most of the tests averaging within 3-4 iterations. Of course, two-iterations runs were performed, with a slight degradation in accuracy for the less demanding tests, and a more pronounced degradation for the more demanding tests.

As a general rule, it was found that running larger time-steps and iterating up to 3-4 times was better than running smaller time-steps and iterating only twice, since the key

parameter, the *cumulative number of iterations*, relative to the final simulation time, was lower using the first strategy.

"Investing in the initial transient pays off ..."

Usually, the most demanding simulations require special care in the early stages of the time evolution. Therefore, the approach followed was to allow a larger number of iterations for the *first ten* time-step of most simulations, and reduce the threshold of the maximum number of iterations subsequently (typically to *only three* iterations).

As a result, unwanted oscillations arising in the first couple of time-steps were successfully cured. In the early time-steps the numerics adjusts to the combination of initial and boundary conditions, and after this phase is elapsed, the system settles into its evolving state, making increasingly easier the convergence of the corrector. Although it may not seem so evident, the improvement on the accuracy was consistent.

In addition, just for the first of the predictor/corrector passes (and for only the first ten time steps), a check on the intensity of the artificial viscosity was made: if the time step constraint, based upon the new estimate of the solution (and artificial viscosity), was more restrictive than the one initially provided for the predictor step, then the time step was reduced and the predictor/corrector iteration restarted. Obviously, the additional cost of repeating part of the calculations was involved, but since only the first ten time steps were affected by such strategy, the impact on the overall computational cost of the simulation was minimal.

8.2 Acoustic wave propagation problem

The first test presented is an acoustic problem in which an initial pulse propagates in a one-dimensional domain. This test assesses the ability of the numerical scheme to propagate without damping acoustic disturbances, and, in practice, only a truly second- or higher-order numerical scheme can deliver satisfactory performance.

First- and second-order algorithm solutions are shown in Figures 8.1 and 8.3, while Figure 8.2 shows the solution when a classical hydrocode formulation is applied as outlined in chapter 3.

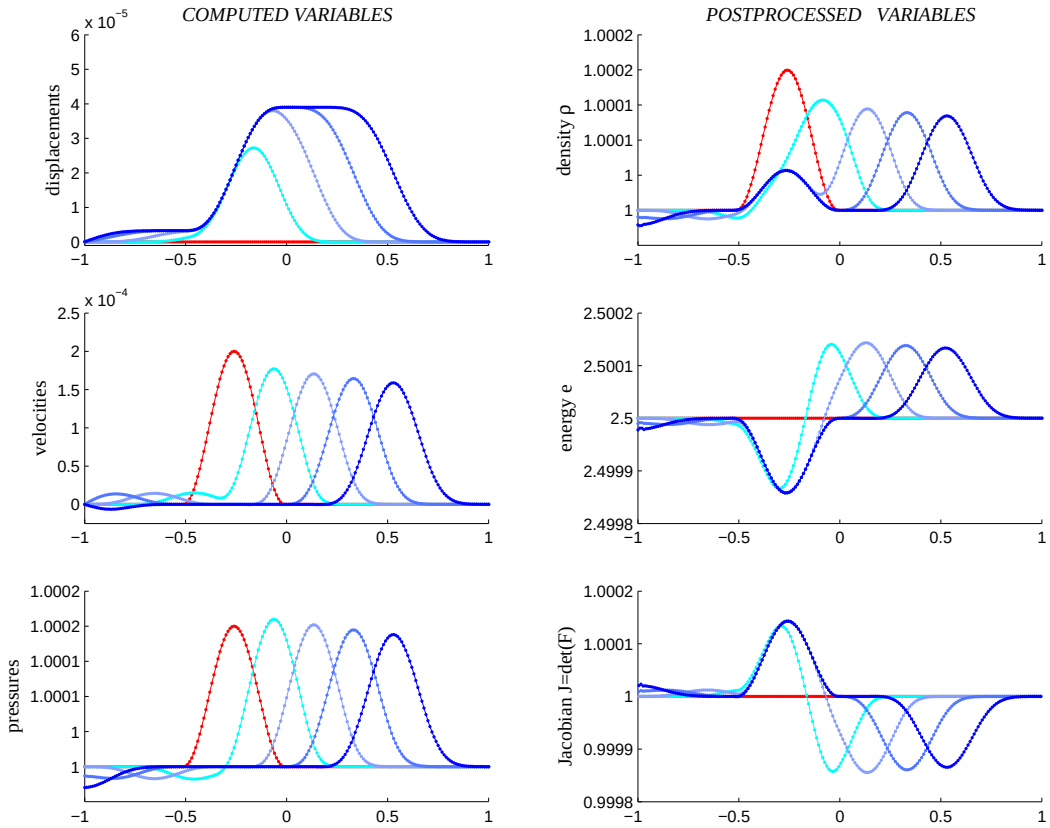

Figure 8.1: First-order scheme Hydro-SUPG solution for the acoustic wave propagation test case. The initial condition is in red. Notice the damping of the solution as time progresses, especially evident in the pressure and velocity plots for the right-moving wave.

The test, similarly to what is reported in Shakib, Hughes, and Johan [64], consists of the following initial conditions at time $t = 0$:

$$u_0 = 0 \tag{8.9}$$

$$v_0 = \tilde{c}\, S \tag{8.10}$$

$$\rho_0 = \tilde{\rho}\, (1 + S) \tag{8.11}$$

$$p_0 = \tilde{\rho}\, \tilde{c}^2\, (1 + S) \tag{8.12}$$

with $\tilde{\rho} = 1$, $\tilde{c} = 1$, and, defining λ as the wavelength,

$$S(X) = \begin{cases} (\tilde{\rho}/10^4)(1 - \cos((2\pi/\lambda)(X - X_{off})), & 0 \leq X - X_{off} \leq \lambda, \\ 0, & \text{otherwise}, \end{cases} \tag{8.13}$$

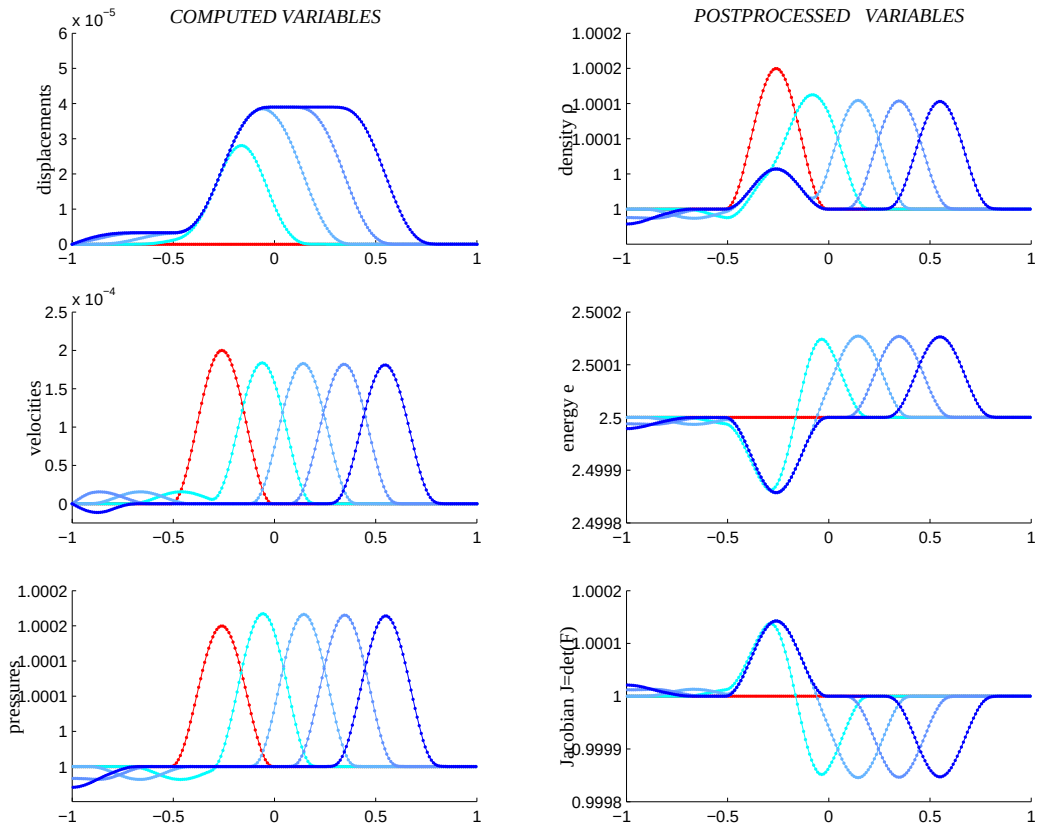

Figure 8.2: Hydrocode solution for the acoustic wave propagation test case. The initial condition is in red. The scheme seems to deliver good performance, although a closer look reveals that the disturbances are slightly damped by the artificial viscosity acting in compression (see Fig. 8.4), and for long times would eventually dissipate completely.

In practice the initial field is a cosine pulse superposed to a constant field, 10,000 times larger in magnitude. X_{off} is an offset point with respect to the origin of the coordinate system.

The domain is $V = [-1, 1]$, discretized with a uniform mesh of 200 elements, and the choices $\lambda = 1/2$ and $X_{off} = -1/2$ were made. Therefore the initial acoustic disturbance is initially located on the interval $[-1/2, 0]$, and is expected to propagate with wave speed c_0.

The first-order method and the hydrocode have been run at $CFL = 0.9$, since the simulation was unstable further approaching the stability threshold of $CFL = 1$. Surprisingly enough, it is possible with the second-order to run at $CFL = 1$. For the second-order Hydro-SUPG method, other simulations were performed for $CFL \in [0.5, 1]$

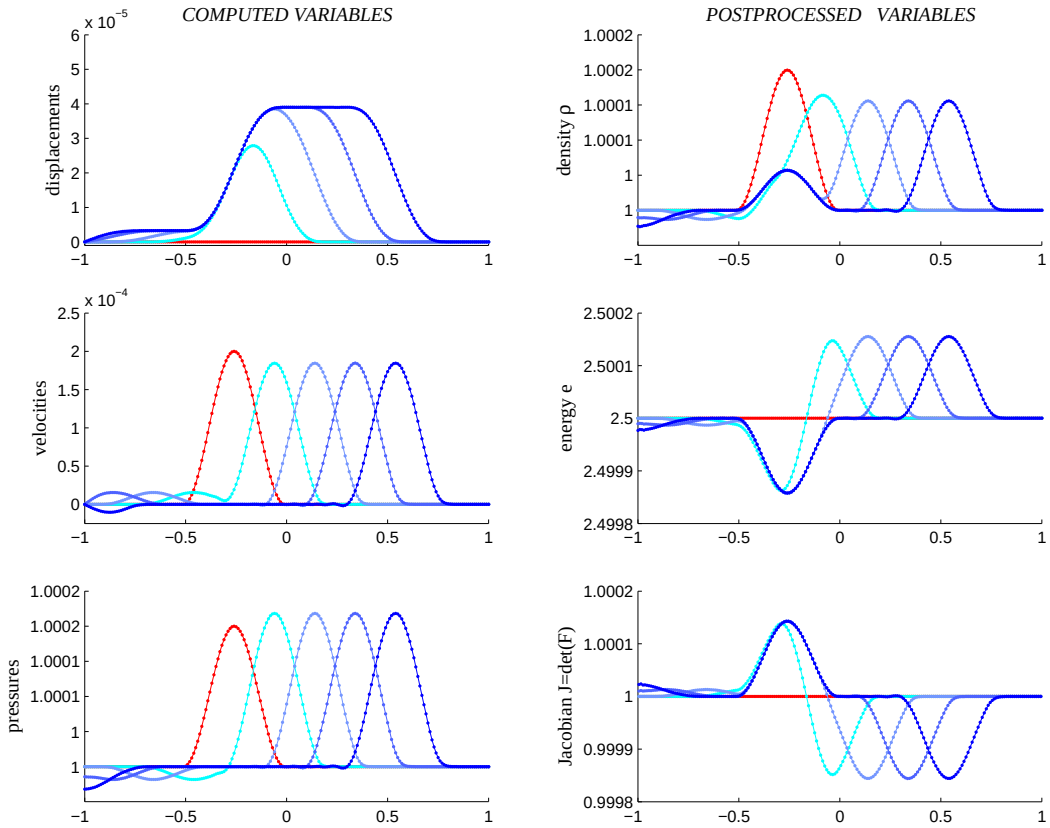

Figure 8.3: Second-order Hydro-SUPG solution for the acoustic wave propagation test case. The initial condition is in red. It is noticeable the extremely good performance of the scheme, since virtually no damping is produced and the disturbances relative to the three waves, once formed, maintain their respective magnitude, as time progresses in the simulation.

and no significant differences where observed with respect to the case $CFL = 1$, presented herein. For the given tolerance on the pressure, *two* iterations were needed for the first-order scheme and *three* iterations for the second-order scheme.

It is interesting to observe that three waves are actually generated by the initial condition (8.9)–(8.13):

1. A (forward) right-moving wave of larger amplitude.

2. A (reverse) left-moving wave, which eventually bounces off the left wall, due to the zero displacement Dirichlet boundary condition.

3. A standing entropy wave, which persists in the original position (see the density and energy plots).

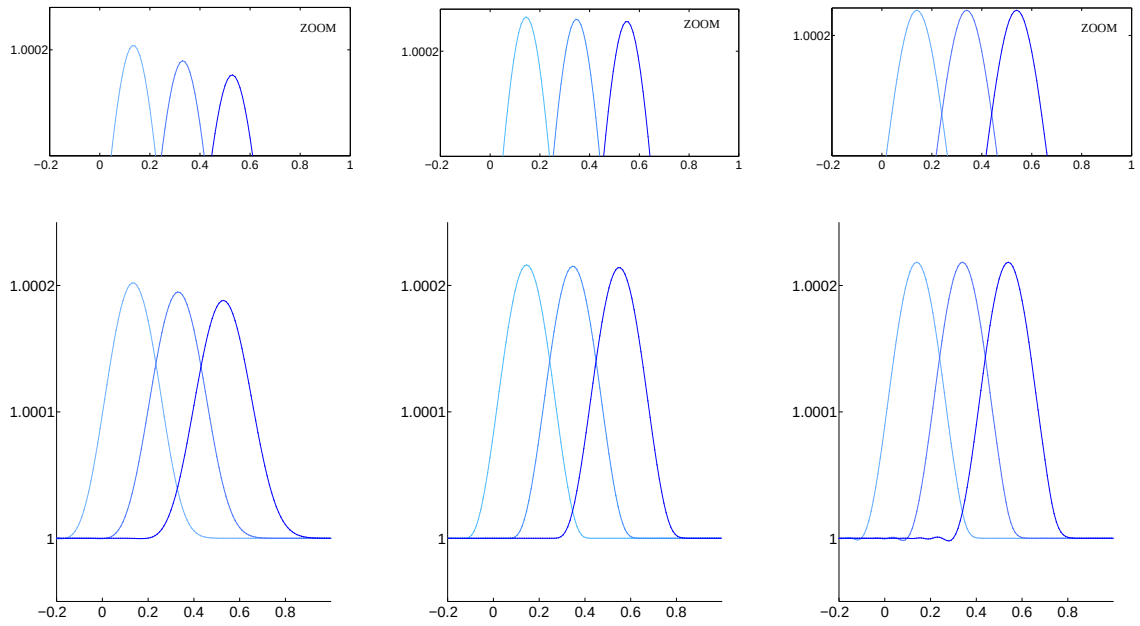

Figure 8.4: Pressure at $t = 0.338, 0.507, 0.676$. Comparison between the first-order Hydro-SUPG (left), the hydrocode solution (middle) and the second-order Hydro-SUPG (right) for the right moving acoustic wave. The three curves in each plot show the pressures corresponding to the last three sample times of Figures 8.1, 8.2 and 8.3. Severe damping is noticeable for the first-order solution. Although mild, damping is still present in the hydrocode solution (compare the upper plots, where a zoomed view is presented), while the second-order Hydro-SUPG solution is practically undamped, a very remarkable result.

Eventually, the waves will bounce off the boundaries at $X = -1$ and $X = +1$, where Dirchlet zero velocity boundary conditions are imposed.

In the case of Hydro-SUPG, the stabilization and DC operators were active at all times, in the sense that they were not turned off. However, due to the smoothness of the solution, the residual was negligible, and so were both the stabilization and discontinuity capturing terms. The solution is actually indistinguishable from the plain Galerkin solution (with stabilization/DC operators turned off).

Damping of the waves is severe for the first-order case, and milder for the hydrocode solutions (see Fig. 8.4, and compare the magnitude of the pressure peaks as time progresses). On the other hand, optimal performance is obtained with the second-order Hydro-SUPG scheme, for which damping is virtually absent.

It is important to stress once more that the key factor for the good performance of the second-order SUPG algorithm is the residual nature of the stabilization/viscosity.

Figure	$\rho_0^{(L)}$	$\rho_0^{(R)}$	$p_0^{(L)}$	$p_0^{(R)}$	$v_0^{(L)}$	$v_0^{(R)}$	γ
8.5 & 8.6	1.0	1.0	0.4	0.4	-2.0	2.0	1.4
8.7 & 8.8	1.0	0.125	1.0	0.1	0.0	0.0	1.4
8.9 & 8.10	1.0	1.0	1000	0.01	0.0	0.0	1.4
8.11 & 8.12	1.0	1.0	0.01	100	0.0	0.0	1.4
8.13 & 8.14	5.99924	5.99242	460.894	46.0950	19.5975	-6.19633	1.4
8.15 & 8.16	–	1.0	–	0.0	–	-1.0	5/3

Table 8.1: Initial condition for the Riemann problem suite from Toro [65], and the one-dimensional Noh [53] test (last row of the table).

Figure	$\rho_*^{(L)}$	$\rho_*^{(R)}$	p_*	v_*	M_* (*Shock Mach no.*)
8.5 & 8.6	0.02185	0.02185	0.00189	0.0000	No shocks
8.7 & 8.8	0.42632	0.26557	0.30313	0.92745	1.65563
8.9 & 8.10	0.57506	5.99924	460.894	19.5975	198.760
8.11 & 8.12	5.99242	0.57511	46.0950	-6.19633	62.8581
8.13 & 8.14	14.2823	31.0426	1691.64	8.68975	1.813523 / 5.621311
8.15 & 8.16	4.0	–	4/3	0.0	∞

Table 8.2: Intermediate state values for the Riemann problem suite from Toro [65], as well as the Noh [53] test.

8.3 A suite of Riemann problems

In the following sections a suite of Riemann problems from the book by Toro [65] is solved, with the purpose of comparing the performance of the Hydro-SUPG second-order implementation against the prototypical hydrocode implementation, under a very broad range of conditions. In addition, the planar version of the implosion test devised by Noh [53] will be presented.

Table 8.1 summarizes the initial conditions and Table 8.2 presents the intermediate (*star*-) state values – computed solving an exact Riemann problem – at the final time of the simulation.

Table 8.3 presents the parameters and the statistics relative to the tests performed. All tests were performed on a 100-element uniform mesh. As a general remark, it was observed that the hydrocode solutions were developing much stronger oscillations in the transient phase before the final time, compared to the corresponding Hydro-SUPG solutions. For the sake of brevity, the corresponding early stage results will be omitted.

Method	Test	Fig.	CFL	$N_{i-P/C}$	$N_{P/C}$	Avg. it.	N_t	I_T	T_{final}
H-CODE	123	8.5	.9	–	–	–	7	–	0.15
H-SUPG2nd	123	8.6	.9	7	3	4.8571	7	34	0.15
H-CODE	Sod	8.7	.9	–	–	–	41	–	0.25
H-SUPG2nd	Sod	8.8	.9	7	3	3.4750	40	139	0.25
H-CODE	LWC	8.9	.9	–	–	–	50	–	0.012
H-SUPG2nd	LWC	8.10	.9	7	3	3.9231	39	153	0.012
H-CODE	RWC	8.11	.9	–	–	–	46	–	0.035
H-SUPG2nd	RWC	8.12	.9	7	3	4.2086	35	141	0.035
H-CODE	2SH	8.13	.9	–	–	–	107	–	0.035
H-SUPG2nd	2SH	8.14	.9	7	3	3.3626	91	306	0.035
H-CODE	NOH	8.15	.9	–	–	–	119	–	0.6
H-SUPG2nd	NOH	8.16	.9	4	3	3.0893	112	346	0.6

Table 8.3: Summary of the tests. All simulations were performed on a uniform grid of 100 elements. H-CODE and H-SUPG2nd, stand for hydrocode and second-order Hydro-SUPG, respectively. N_t is the number of time-steps to reach the target time T_{final}. $N_{i-P/C}$ is the maximum number of predictor/corrector passes (including repetition of the predictor pass due to CFL constraints) allowed in the first 10 time-steps. $N_{P/C}$ is the maximum number of predictor/corrector passes allowed after the first 10 time-steps. I_T is the cumulative number of iteration to target, used as measure of the overall cost to reach the time T_{final} in the simulation. All the simulations were performed using the explicit variant and at the maximum Courant number (CFL) allowing stable computations.

8.3.1 123 problem

The so-called 123 problem assesses whether or not a numerical scheme delivers higher-order performance under severe expansion conditions.

The true challenge for this test is to obtain an accurate solution for the energy. Notice that the initial condition for the Hydro-SUPG simulation is such that the initial states present an off-set of half a cell to the right, resulting in different evaluation points for the solution as it can be seen comparing the plots of the energy e and Jacobian J.

It can be easily checked in [49] and [65] that substantially none of the typical Eulerian state-of-the-art methods considered higher-order, based on MUSCL, ENO, WENO, WAFT, CLAW, or more advanced approaches passes satisfactorily the 123 test case.

On the other other hand, Lagrangian-type methods are extremely accurate for this type of problems as it can be seen in Figures 8.5 and 8.6, where the computed solutions are practically always on top of the exact solution.

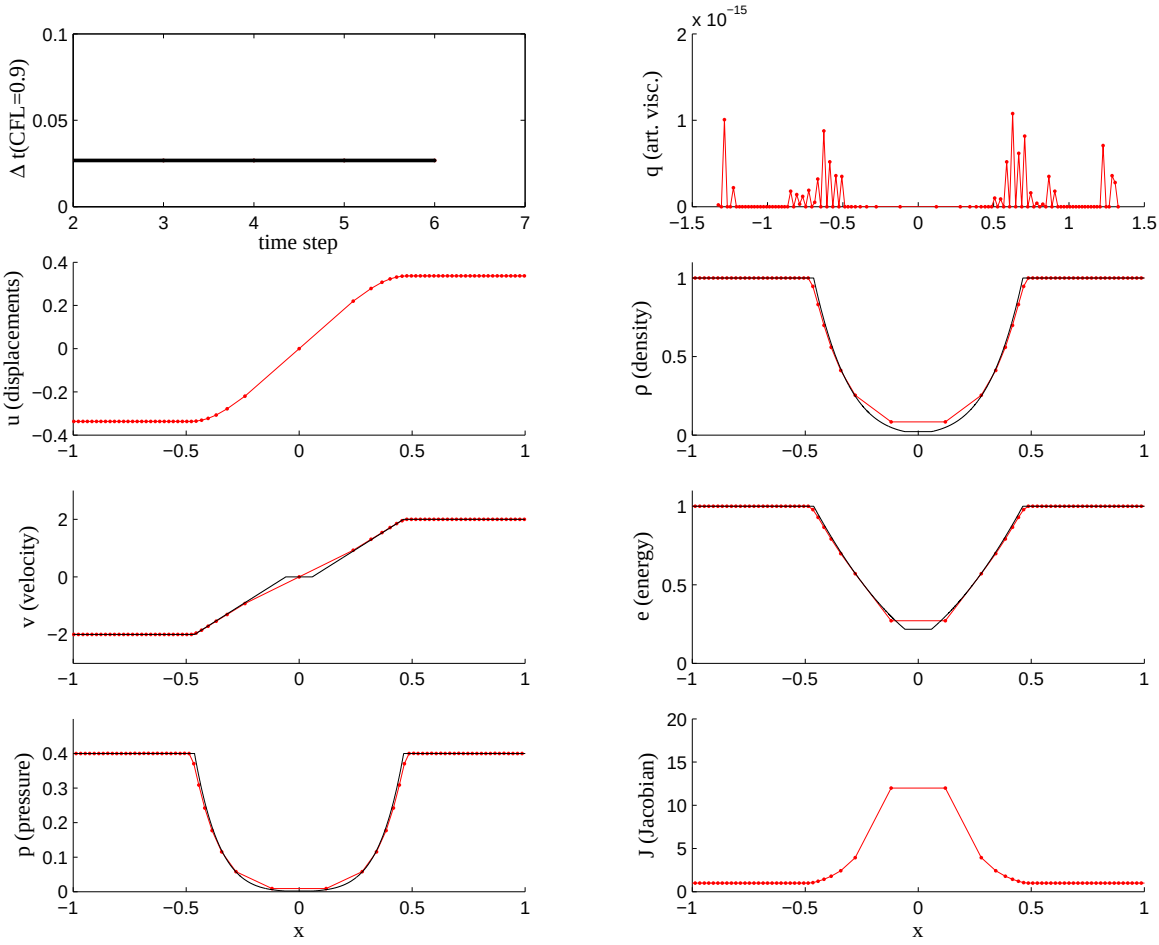

Figure 8.5: Hydrocode solution for the 123 problem. Exact solution is black, computed is red (continuous with dots). Notice the accuracy with which the expansion hull is captured, due also to the fact that the artificial viscosity (top-right corner) is virtually zero. On the top-left corner, the history of $\Delta t_{CFL_{adv}}$ (red), Δt_{CFL_q} (blue), and $\Delta t_{CFL_{max}}$ (black) is presented: The CFL condition is dominated by the advective constraint.

For the Hydro-SUPG version, when the threshold ϵ_p for the nonlinear solver was reduced, the results were almost indistinguishable from the exact solution, at the price of an increase in the number of iterations.

8.3.2 Sod's problem

The Sod's test is a classical for one-dimensional gas dynamics. Both hydrocode and SUPG algorithm were run at $CFL = 0.9$.

Notice however that the hydrocode tends to be affected by the time-step constraint

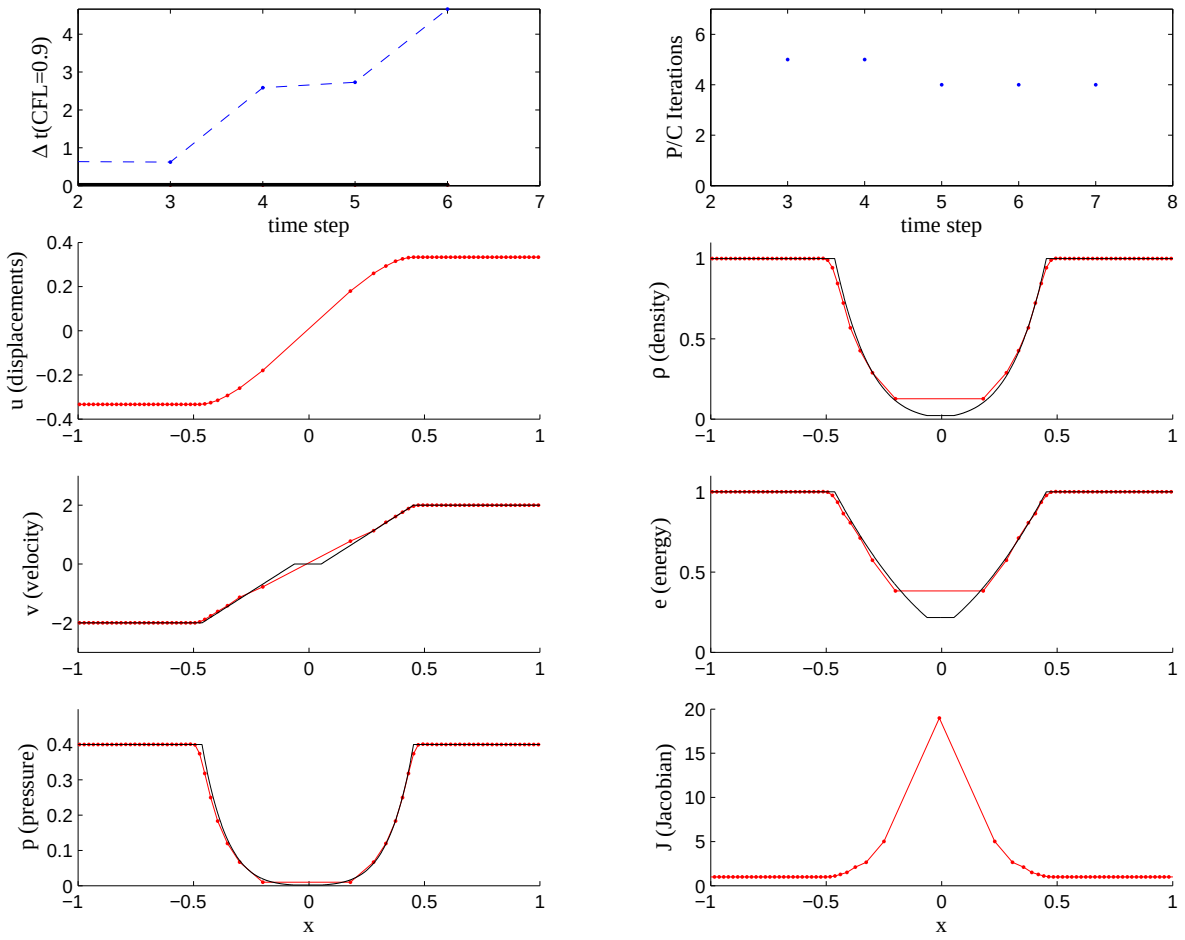

Figure 8.6: Hydro-SUPG second-order variant. Notice the very good performance for the energy component of the solution. This is due to a very accurate representation of velocities and displacements. The plots are organized as in Figure 8.5, the artificial viscous flux q replaced by the history of the number of iterations of the predictor/corrector per time step.

due to the artificial viscosity, a trend to be seen in most of the test. This behavior is not optimal, since the DC operator causes the simulation to slow down with respect the constraint imposed by the advective CFL condition.

As one can see in Figure 8.8, after the first two time-steps, the SUPG approach is optimal in terms of the CFL restriction, since the time-step constraint is entirely due to the advective CFL condition.

The hydrocode delivers good performance, but a pronounced overshoot is present in the energy plot and the velocity shows low accuracy in the representation of the solution past the shock front. The Hydro-SUPG results, instead, show a well captured energy plateau: The slight smearing of the contact discontinuity is due to the linear

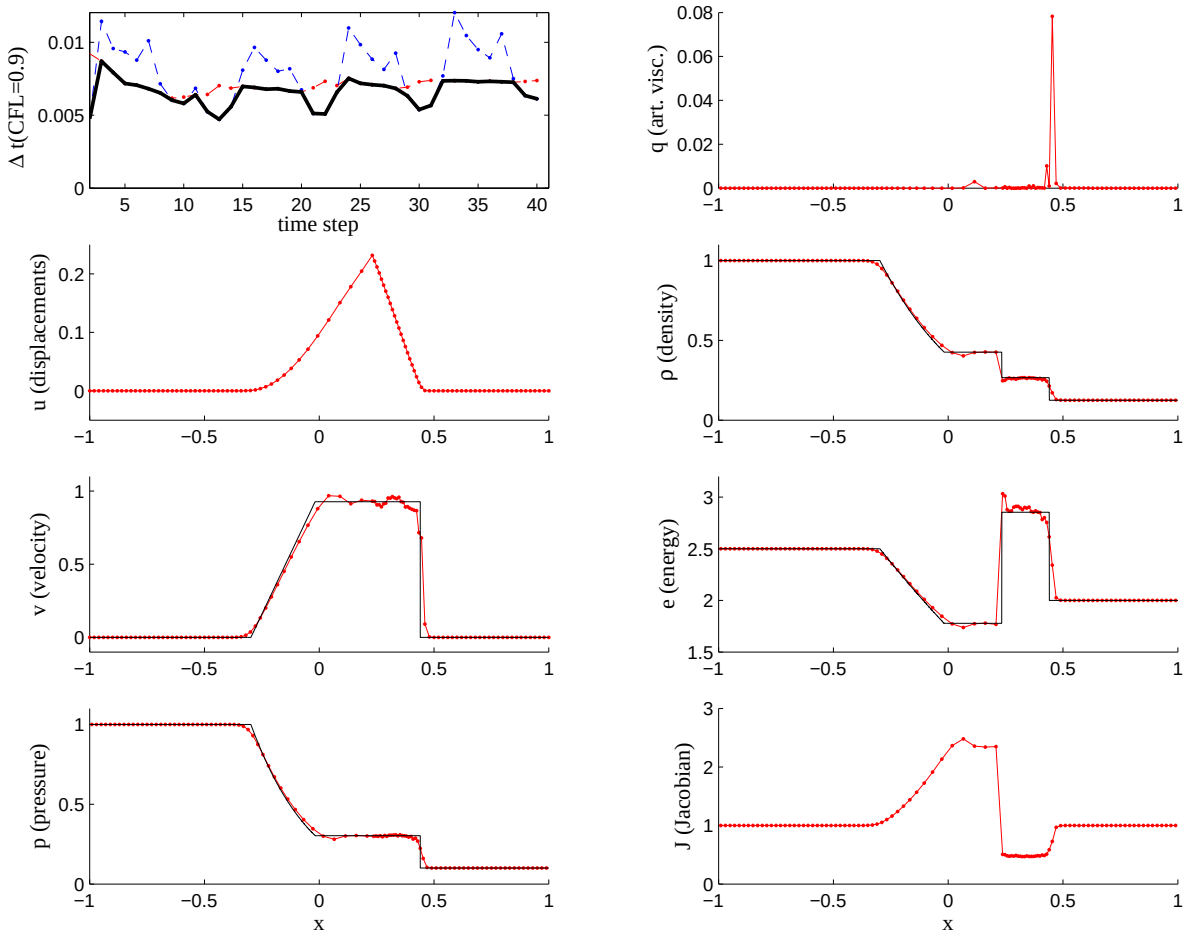

Figure 8.7: Hydrocode solution for the classical Sod's problem. The shock is captured with some wiggles, and the expansion presents a moderate overshoot. The energy is not computed accurately in the upper plateau. On the top-left corner the time-step history of $\Delta t_{CFL_{adv}}$ (red), Δt_{CFL_q} (blue), and $\Delta t_{CFL_{max}}$ (black).

approximation for the pressure. In particular, it was observed in mesh refinement studies that the contact discontinuity spans usually two-three elements, *independent* of the mesh spacing. The number of iterations is between three and four, although good results were obtained also with two-three iterations only.

8.3.3 Left-half of Colella-Woodward's blast

The present and following sections are devoted to test accuracy and robustness in blast-type computations, with shocks in increasing intensity.

The hydrocode computation shows again lack of optimality in the time-step limitations, even more evident than in the Sod's test case. In addition, mild wiggles are present

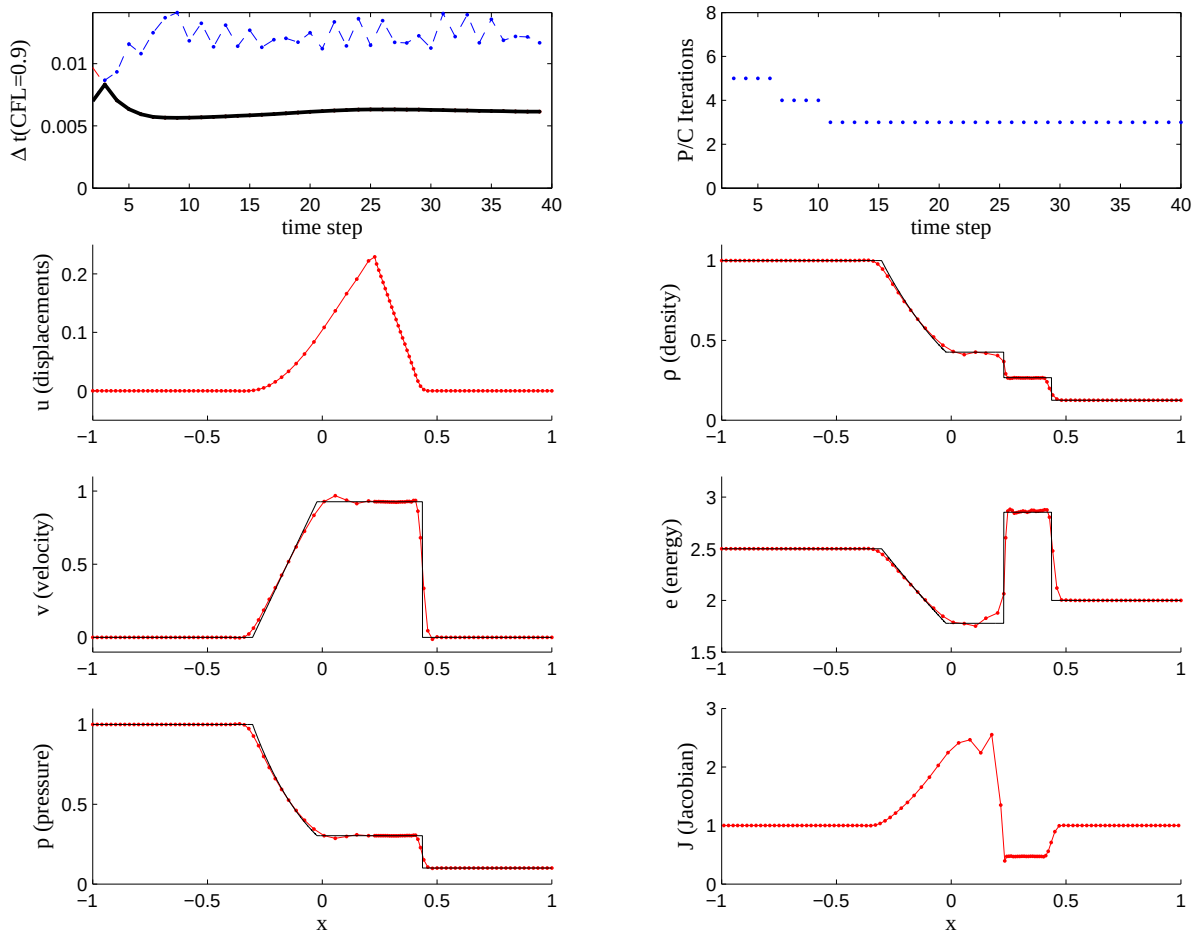

Figure 8.8: Hydro-SUPG, second-order scheme solution for the classical Sod's problem. The energy is very accurate, the contact is properly captured within two/three elements.

in the proximity of the shock front for velocity and pressure.

The second-order Hydro-SUPG scheme delivers much better performance, both in terms of monotonicity of the solution and time-step limitations. The overshoot in the velocity at the end of the expansion is more moderate than for the hydrocode plots. The fact that a scheme based on piecewise linear continuous interpolation of all variables could deliver such performance for blast-type computations is quite surprising, and indicative of flexibility of the DC operator design.

One detail that might escape the eye is that, occasionally, the pressure can take negative values ahead of the shock. The calculations of $\boldsymbol{\tau}$ and the CFL condition involve the speed of sound in the medium, for which the square root of the pressure is needed: If the pressure is negative, a complex sound speed is produced. As an academic rather

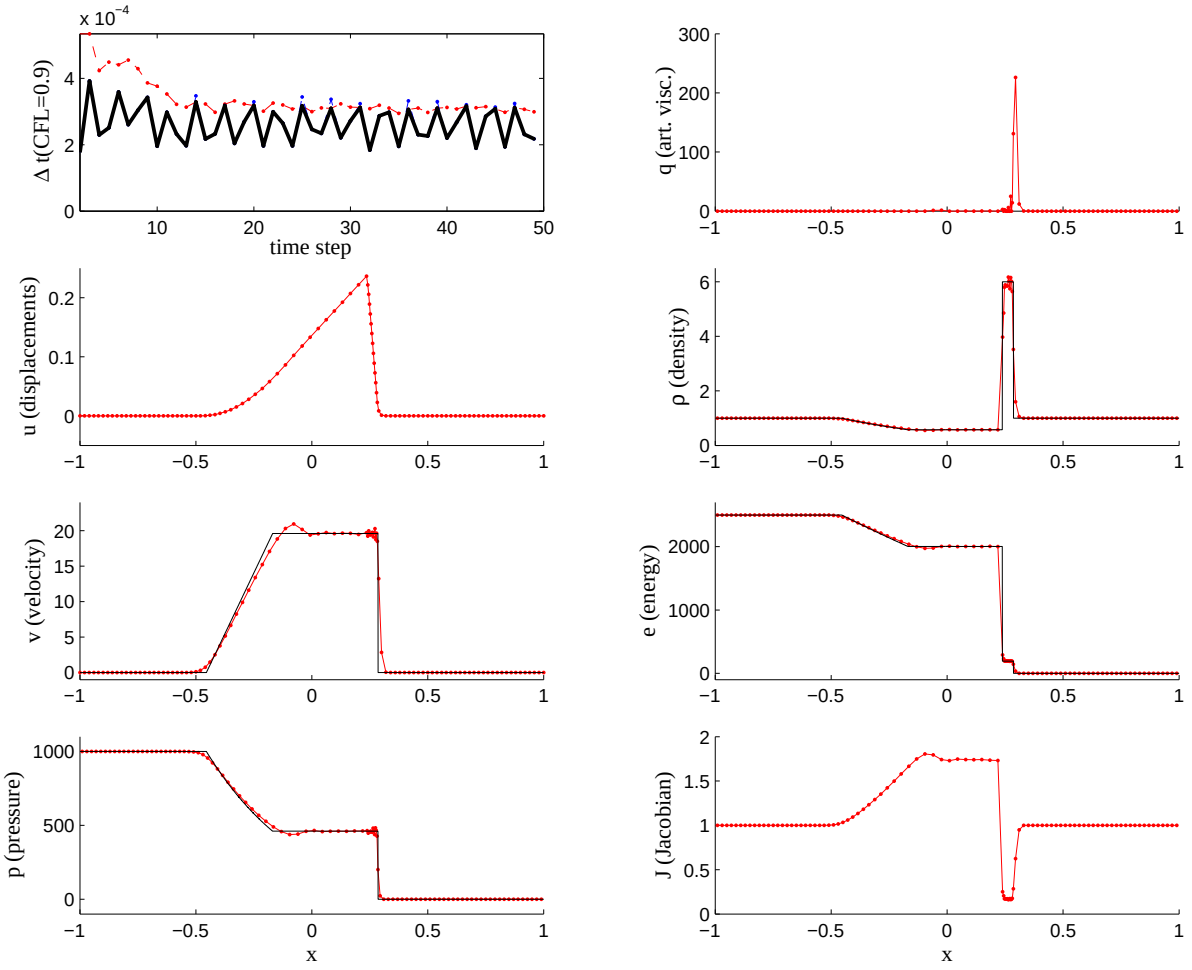

Figure 8.9: Left-half of the Colella-Woodward's blast: Hydrocode solution. Notice the pronounced over/under-shoot for the velocity/pressure at the beginning of the expansion (about $X = -.25$). Furthermore, wiggles are present past the shock front in the velocity and pressure plots.

than practical remark, when complex arithmetic was used in the computations (to allow complex sound speeds), the real part of the solution resulted very accurate. The inconvenience of complex arithmetic has been avoided by taking the positive part of the pressure (i.e., $\max\{0, p\}$) when computing the stable time-step and the $\boldsymbol{\tau}$ tensor.

The explanation for the good performance of the code has to do with the fact that for Lagrangian coordinates, a negative pressure *may not* lead to a negative density, since the density is computed using the initial density (positive) and the Jacobian (positive by definition). Therefore, a very mild negative pressure is not causing volume inversions, which were never experienced in this case.

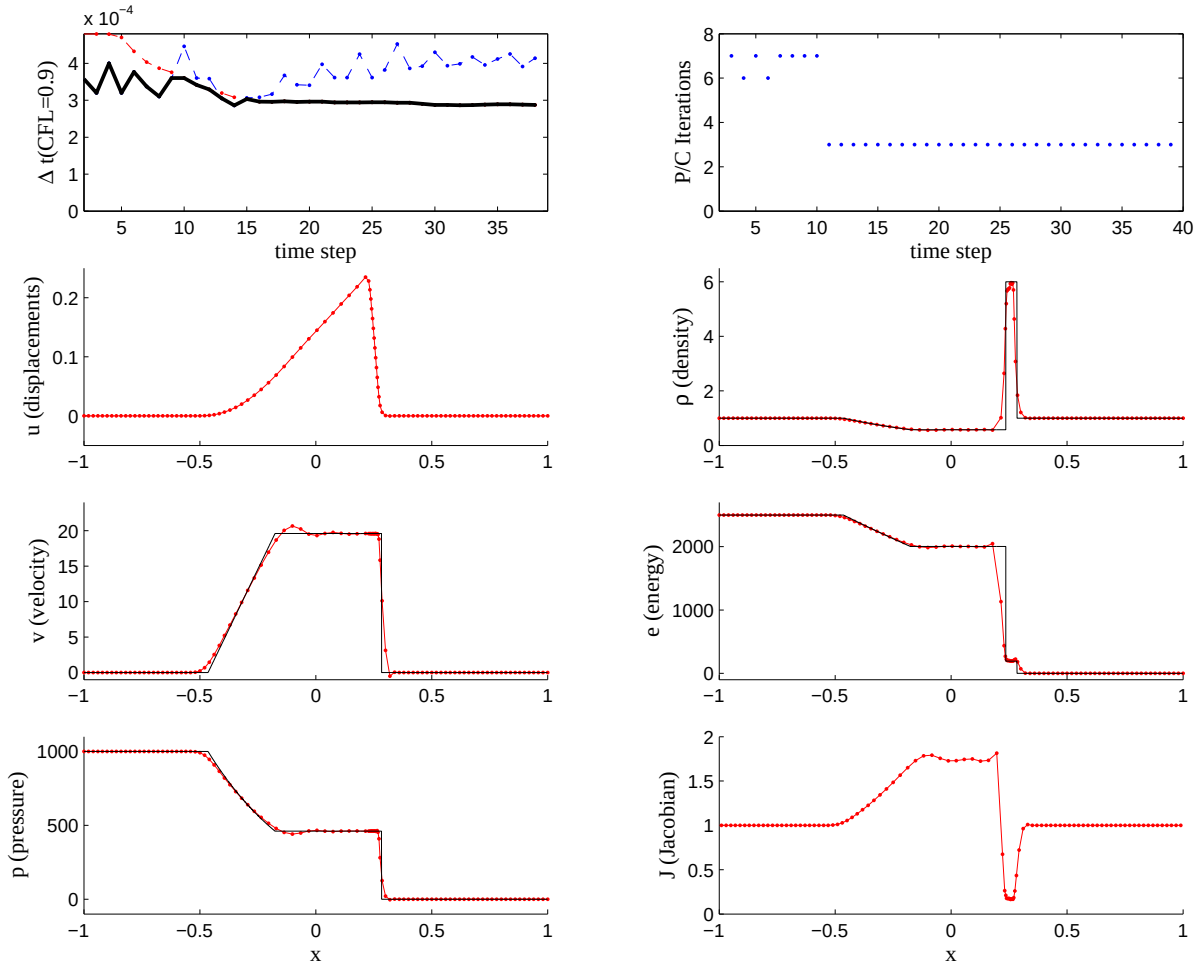

Figure 8.10: Left-half of the Colella-Woodward's blast: Second-order Hydro-SUPG solution. The over/under-shoots are practically absent for velocity and pressure. The plateau for the density is properly captured, and the limitations on the time-step are almost entirely due to the advective effects.

The energy does not show negative values, since the co-volume post-processing was used and nodes of negative pressure were usually isolated between nodes in which the pressure was positive.

8.3.4 Right-half of Colella-Woodward's blast

For the right-half of the Colella-Woodward's blast, considerations almost identical to the ones outlined in the previous section hold. The test is presented for completeness.

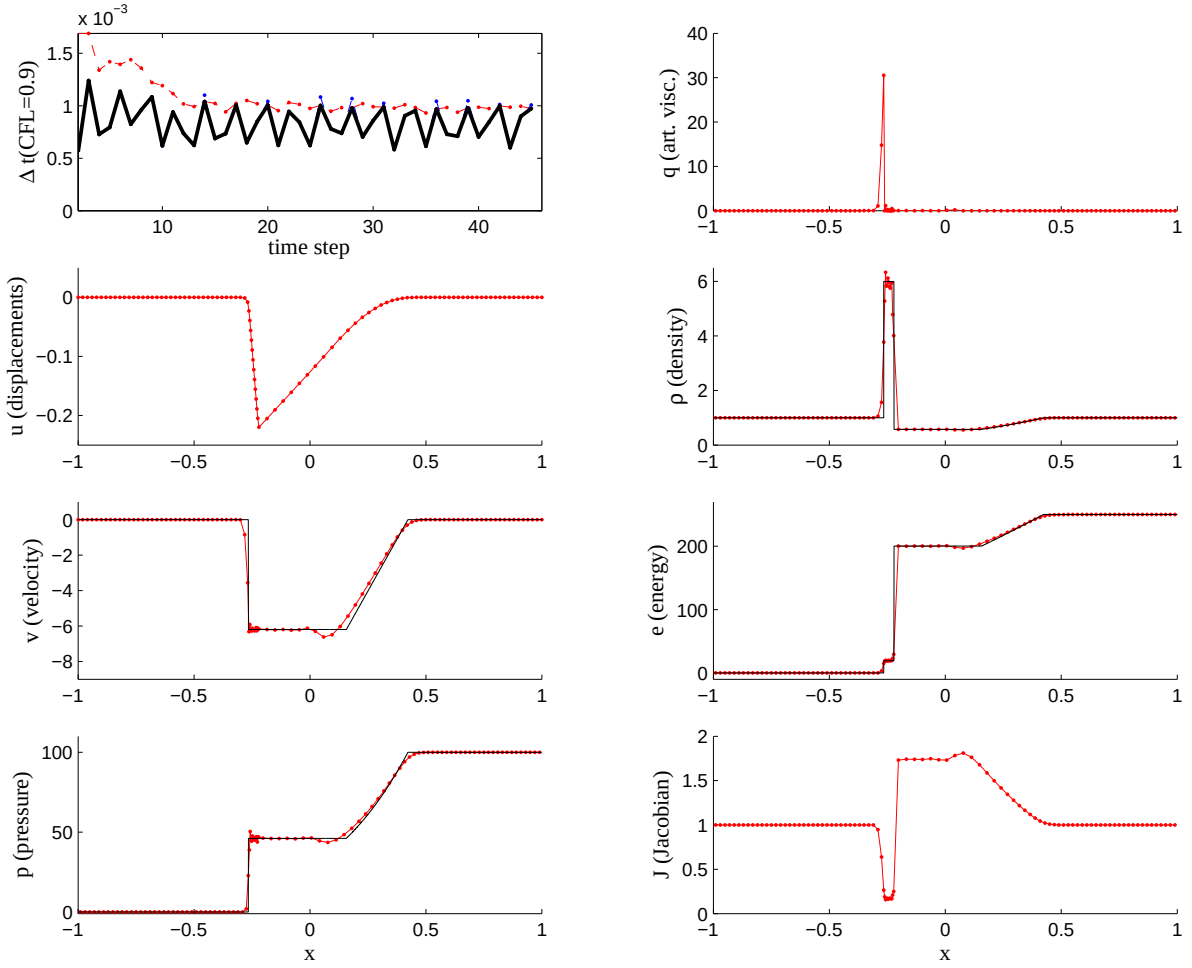

Figure 8.11: Right-half of the Colella-Woodward's blast: Hydrocode solution. Comments are analogous to the ones in Figure 8.9.

8.3.5 Two-shock problem

The two-shock test is one of the most demanding test of this suite in terms of robustness. This is mainly due to the fact that the interaction of two strong shocks produces a contact discontinuity which needs to be appropriately represented.

The hydrocode shows again sub-optimality of time-stepping, but it is otherwise fairly good in terms of performance, accuracy and robustness. It is worthwhile mentioning that the two-shock test has features very similar to implosion calculations, due to the fact that the shocks are mostly generated by kinetic (velocity) effects: it is very noticeable the strong compression undergone by the computational domain, by comparing its overall length in the final and initial (reference) configuration (the ends of the domain are originally at $X_{min} = -1$ and $X_{max} = +1$), in Figure 8.13, or 8.14.

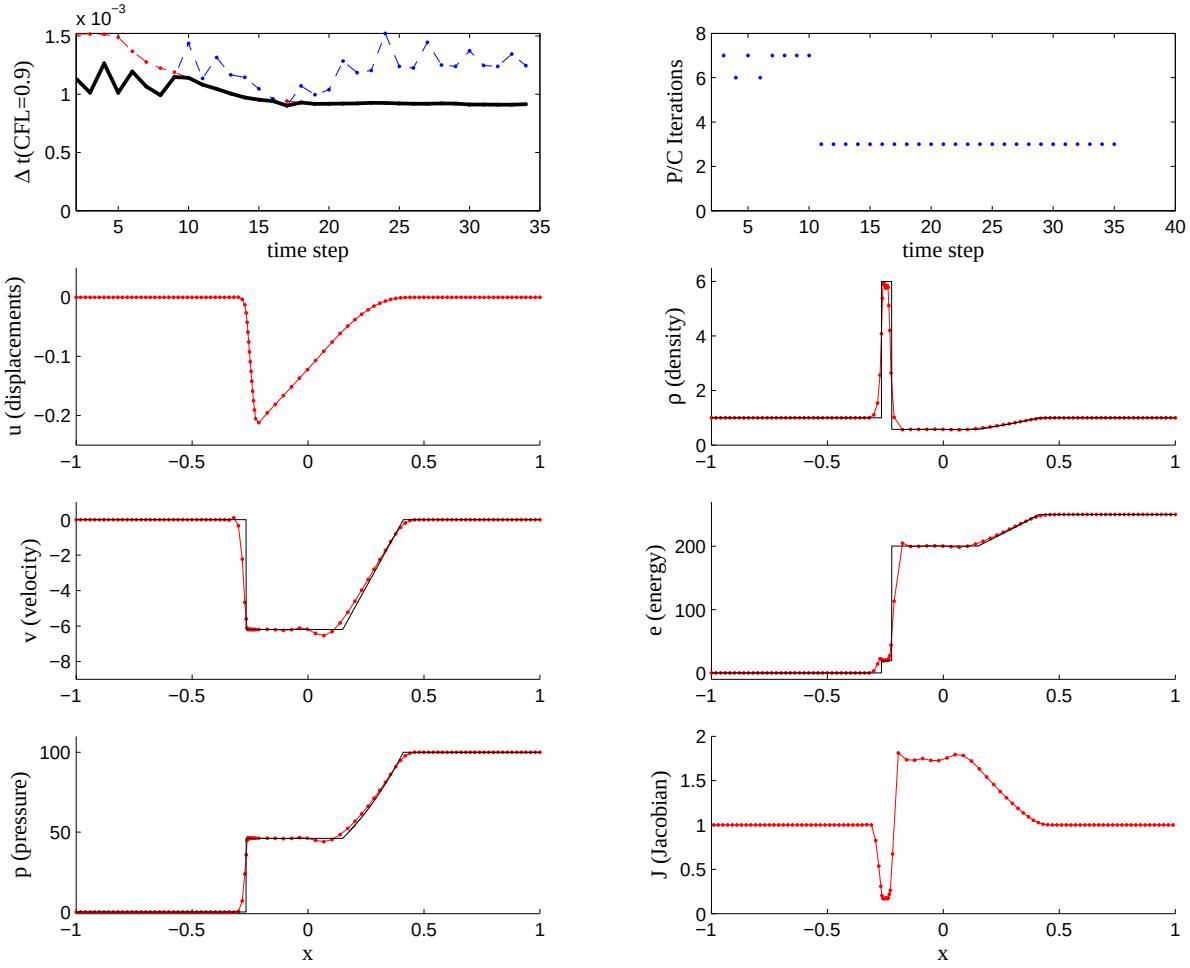

Figure 8.12: Right-half of the Colella-Woodward's blast: Second-order Hydro-SUPG solution. Comments are analogous to the ones in Figure 8.10.

The hydrocode solution suffers from a few wiggles in the velocity and pressure plots, still present in the energy, which suffers also from over/under-shoots as the density. Although over/under-shoots are present in the hydro-SUPG solution, wiggles are practically absent and the quality of the pressure and velocity solutions (directly computed) is practically perfect. The Hydro-SUPG method also shows practically optimal behavior at all times with regard to the time-stepping limitations.

8.3.6 Planar Noh's test

The Noh test is an implosion computation in which a bar of gas is rammed into a hard, zero velocity boundary condition (last node on the left of the computational domain). In particular, the initial pressure of the gas is set to zero, to obtain an infinite strength

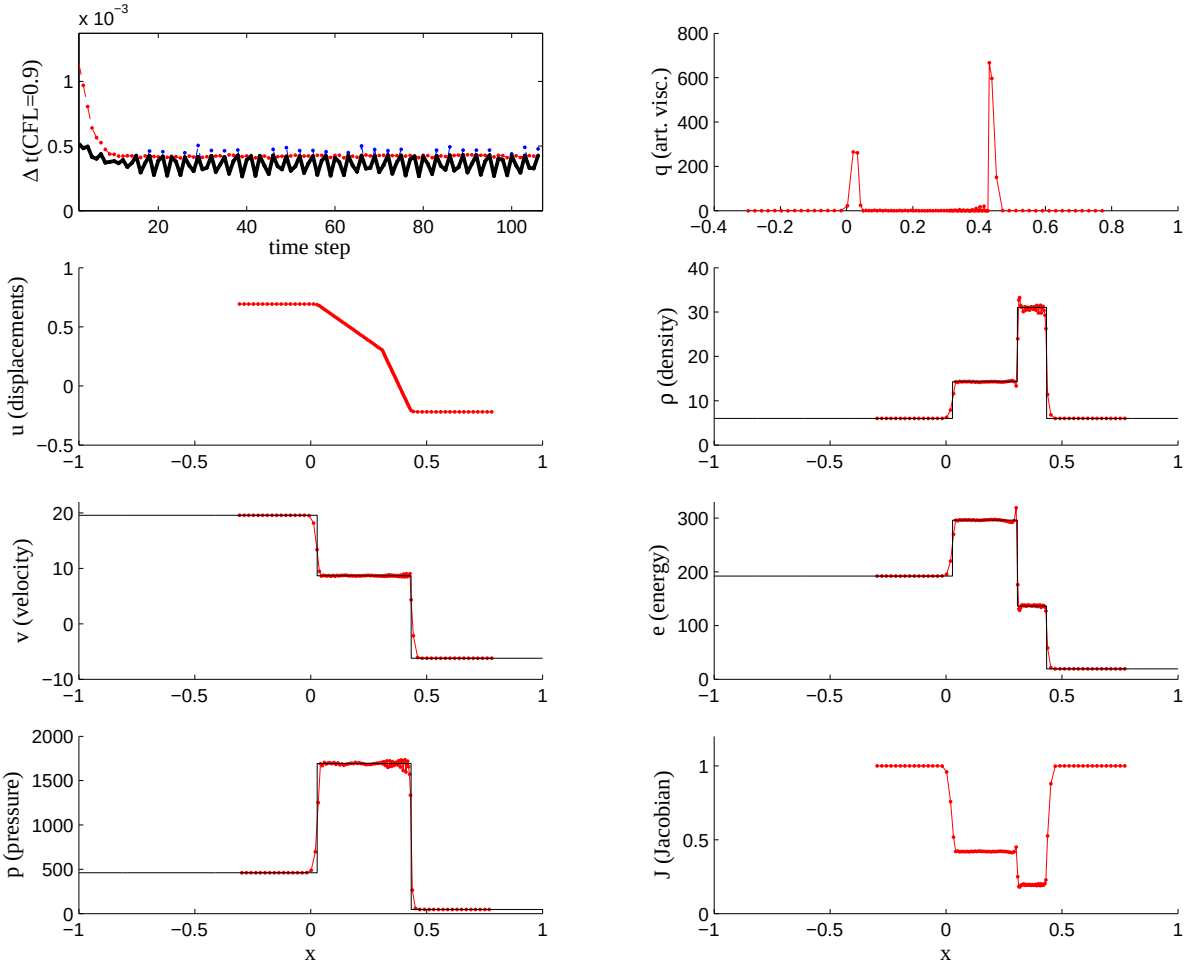

Figure 8.13: Hydrocode solution for the two-shock problem. Over/under-shoots are present for density and energy, and wiggles are present in energy, velocity and pressure plots. The solution is otherwise good.

shock. This test, together with the two-shock test, is the more challenging for robustness in the entire suite.

The hydrocode delivers good performance, it is still sub-optimal as far as time integration is concerned, and presents a few wiggles in the solution past the shock front.

However, the most remarkable feature is the presence of *overheating* in the element facing the left boundary, where it is very noticeable a spike in the energy and a severe dip in the density (see Fig. 8.15).

This is not the case for the Hydro-SUPG method (see Fig. 8.16), for which overheating is absent. This fact can be attributed to the more complex structure of the DC operator with respect to the standard Von Neumann-Richtmyer viscosity, and the

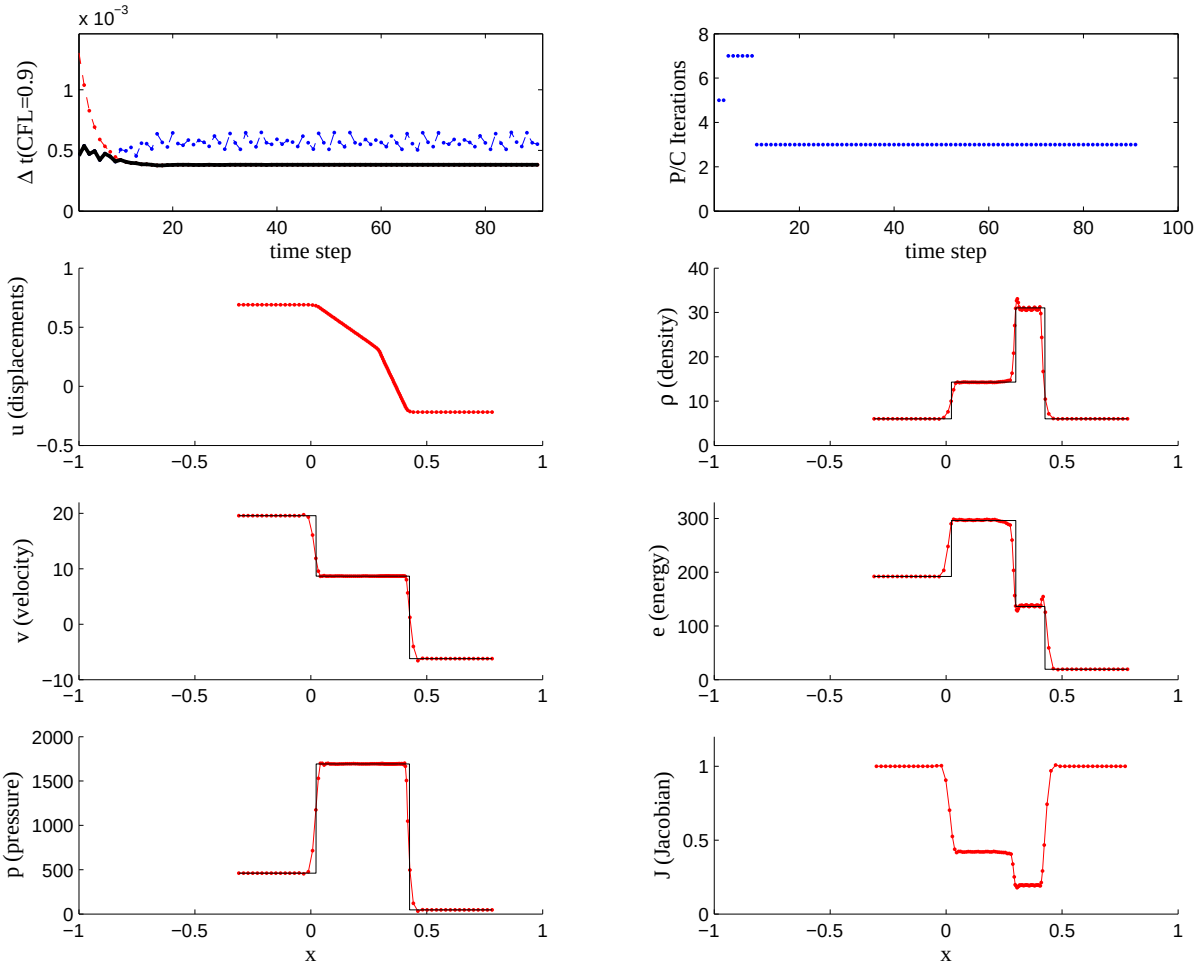

Figure 8.14: Second-order Hydro-SUPG solution for the two-shock problem. Overshoots and undershoots are present in the density and energy, but comparable in magnitude with the ones observed in the hydrocode simulation. Virtually no oscillations are present in the velocity and pressure plots, and the time-step limitations are almost always due to advective effects.

striking similarities of the resulting DC operator with the Noh viscosity [53], which, in fact, was designed to avoid the overheating phenomenon.

8.4 Conclusions and Future Directions

A robust and accurate SUPG approach to Lagrangian hydrodynamics has been proposed and successfully implemented in the case of one-dimensional gas dynamics problems. Notice also that only three iterations are sufficient for an accurate solution with Hydro-SUPG, a very important result for larger-scale implementations in multiple dimensions.

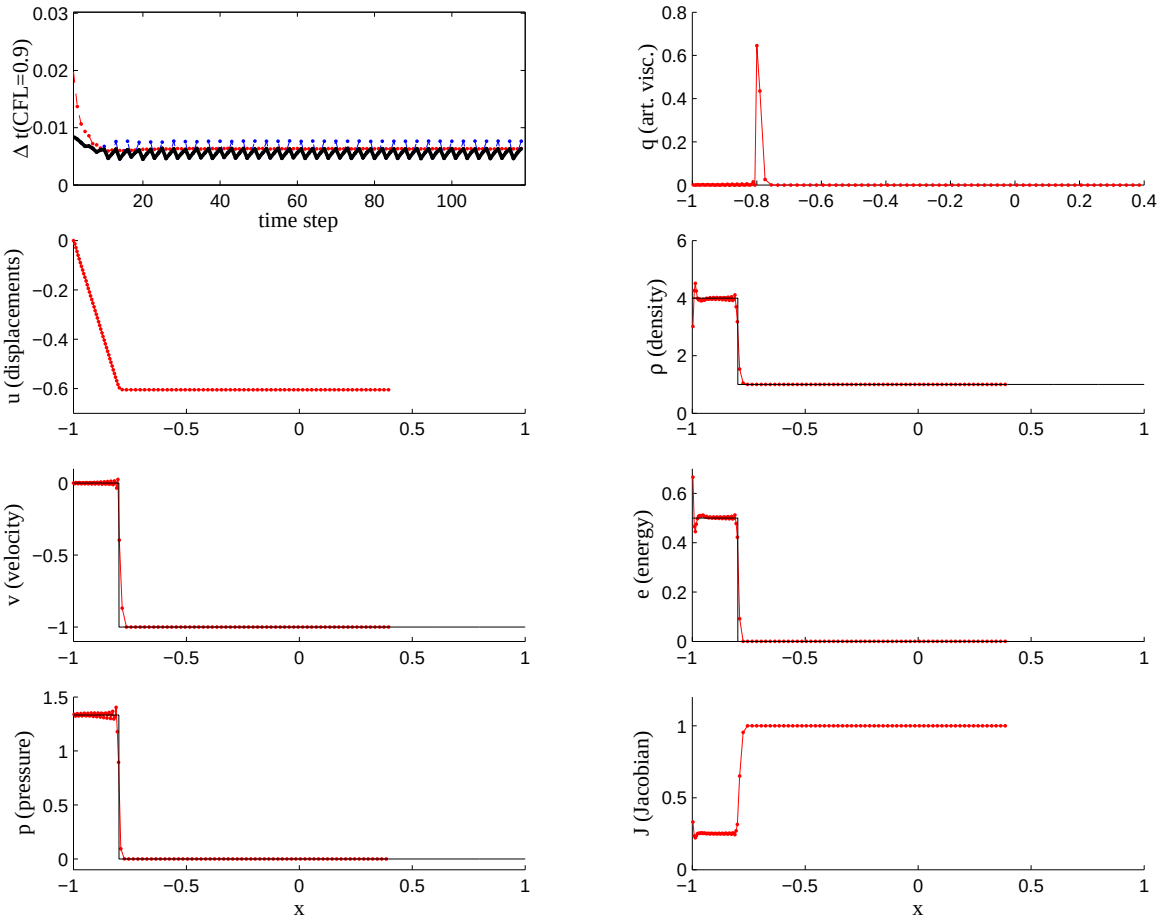

Figure 8.15: Hydrocode solution for the one-dimensional, planar Noh's test.

The algorithm proposed in this dissertation proves robust under very different conditions, from acoustic wave propagation simulations to blast computations, and in combination with careful time-stepping strategies delivers very interesting performance.

The approach originates from a multiscale analysis of the Lagrangian hydrodynamics equations, and possesses a residual nature as a key feature. The underlaying goal of the proposed approach is to tackle challenging problems in multiple dimensions on triangular/tetrahedral meshes, for which a vast literature documents the very good performance of SUPG-type methods.

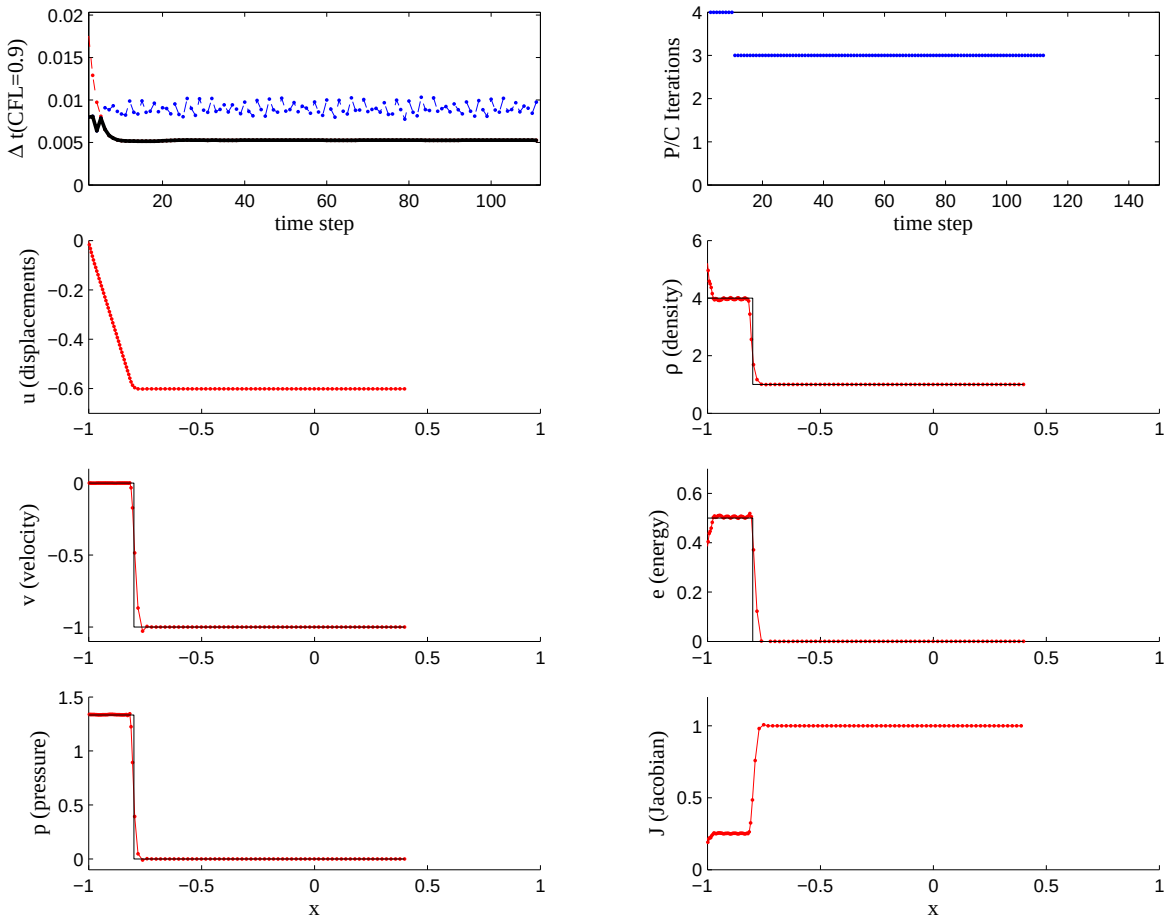

Figure 8.16: Second-order Hydro-SUPG for the one-dimensional, planar Noh's test.

In this sense, it is appears natural to foresee as a future research path the implementation of the proposed concept in multiple dimensions, using the already provided multi-dimensional framework, as outlined in previous chapters.

Part II

A Multiscale Link between the Continuous/Discontinuous Galerkin Worlds

Chapter 9

Linear Advection-Diffusion: Strong Form and Discontinuous Galerkin Notation

The present chapter is devoted to the definition of the advection-diffusion problem in a multi-dimensional domain. The chapter also develops the notation and definitions necessary for Discontinuous Galerkin formulations presented in chapter 10 and 11. Although the problem is linear, the introduction of a force term makes the discussion very interesting, since, typically, nonlinear problems are solved by means of local linearization, as in the case of Newton solvers, for which the force term is represented by the residual of the equations, driven to zero through the iterations.

9.1 The strong form of the problem

Let Ω be a bounded domain in $\mathbb{R}^{n_{\mathrm{d}}}$, and let $\boldsymbol{a}$ be a vector-valued function defined on $\overline{\Omega}$. It will prove useful to define also a partition of the boundary $\Gamma = \partial\Omega$ as follows:

$$\Gamma^- = \{x \in \Gamma : \boldsymbol{a}(x) \cdot \boldsymbol{n}(x) \leq 0\} \tag{9.1}$$

$$\Gamma^+ = \{x \in \Gamma : \boldsymbol{a}(x) \cdot \boldsymbol{n}(x) > 0\} \tag{9.2}$$

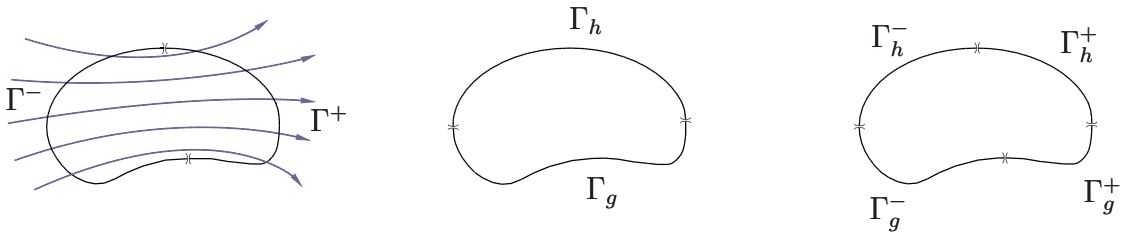

Figure 9.1: Illustration of boundary partitions.

where $\boldsymbol{n}$ is the outward unit normal with respect to Γ. Γ^- will be referred to as the *inflow* boundary and Γ^+ as the *outflow* boundary. The strong form of the problem to be solved is:

$$\begin{cases} \boldsymbol{a} \cdot \nabla\phi - \kappa \Delta\phi = f & \text{in } \Omega \\ \phi = g & \text{on } \Gamma_g \\ (-\boldsymbol{a}\phi\chi_{\Gamma_h^-} + \kappa\boldsymbol{\nabla}\phi) \cdot \boldsymbol{n} = h & \text{on } \Gamma_h \end{cases} \tag{9.3}$$

where Γ_g is the part of the boundary where Dirichlet boundary conditions are imposed, and Γ_h is the portion of the boundary where Neumann conditions are imposed. $\chi_{\Gamma_h^-}$ is the characteristic function of the set Γ_h^-. Note also that $\Gamma = \overline{\Gamma_h \cup \Gamma_g}$, $\Gamma_h \cap \Gamma_g = \emptyset$, and

$$\Gamma_g^{\mp} = \Gamma_g \cap \Gamma^{\mp} \tag{9.4}$$

$$\Gamma_h^{\mp} = \Gamma_h \cap \Gamma^{\mp} \tag{9.5}$$

Therefore, the meaning of the third equality in (9.3) is that the total flux (convective plus diffusive) is imposed at the Neumann inflow boundary Γ_h^- and the diffusive flux only is specified at the Neumann outflow boundary Γ_h^+. The vector $\boldsymbol{a} \in \mathbb{R}^{n_{\mathrm{d}}}$, $n_{\mathrm{d}} \in \{1, 2, 3\}$ is the convective field, assumed divergence-free, with $a_i \in C^1(\Omega)$, $i \in \{1, \ldots, n_{\mathrm{d}}\}$, and κ, a positive constant, is the diffusivity parameter.

9.2 General definitions and notation for the Discontinuous Galerkin method

A number of very important definitions and identities must be presented in order to pave the way for the global Discontinuous Galerkin (gDG) formulation of chapter 10.

In order to make the following exposition consistent, we will assume that the domain Ω in $\mathbb{R}^{n_{\rm d}}$ is *polygonal*. This can be considered as an intermediate step introducing a *discretized geometry*, in which the boundaries may no longer be smooth as they were depicted in Figure 9.1. The notation for inflow and outflow boundary partitions will be maintained, but normals will be defined on boundary (polygonal) edges and so forth for all other relevant quantities.

Let $\mathcal{T}_h$ be a regular family of elements T generating a partition of Ω. For example, T can be thought of as a triangle/tetrahedron, or quadrilateral/hexahedron, in two/three dimensions, respectively. Let h_T denote the diameter of T and $h = \max_{T\in\mathcal{T}_h} h_T$. Let also $\mathcal{E}_h$ be the set of all edges (*including* edges on the boundary Γ) and $\mathcal{E}_h^o$ the set of internal edges (*excluding* edges on the boundary Γ), that is

$$\mathcal{E}_h = \mathcal{E}_h^o \cup \Gamma \tag{9.6}$$

$$\mathcal{E}_h^g = \mathcal{E}_h^o \cup \Gamma_g \tag{9.7}$$

$$\mathcal{E}_h^h = \mathcal{E}_h^o \cup \Gamma_h \tag{9.8}$$

By edges, we mean a manifold of dimension $n_{\rm d} - 1$, representing the interface between two elements: for example, true edges for a two-dimensional triangular mesh, or faces for a three-dimensional tetrahedral mesh. It is also natural to define a partition of the element boundary ∂T (see Fig. 9.3):

$$\Gamma_T^- = \{x \in \partial T : \boldsymbol{a}(x)\cdot\boldsymbol{n}(x) \le 0\} \tag{9.9}$$

$$\Gamma_T^+ = \{x \in \partial T : \boldsymbol{a}(x)\cdot\boldsymbol{n}(x) > 0\} \tag{9.10}$$

Here $\Gamma_T^{\mp}$ represents the element inflow/outflow boundary, respectively, so that $\partial T = \Gamma_T = \Gamma_T^+ \cup \Gamma_T^-$. In order to derive a DG formulation, *jumps* and *averages* for scalar and vector

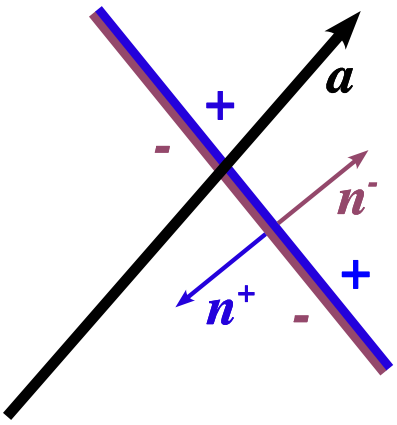

Figure 9.2: Schematics of the normals and +/− regions for an edge with respect to the convective field $\boldsymbol{a}$.

functions have to be defined on the edges of $\mathcal{T}_h$. Therefore, for an interior edge $e \in \mathcal{E}_h^o$, let T_1 and T_2 be the two elements sharing it, and by $\boldsymbol{n}^1$ and $\boldsymbol{n}^2$ their respective outward-pointing unit normals. Accordingly, let φ be a scalar field, and $\varphi^i := \varphi|_{T_i}, \quad i = 1, 2$. For $e \in \mathcal{E}_h^o$:

$$\langle \varphi \rangle := \frac{1}{2}(\varphi^1 + \varphi^2) \tag{9.11}$$

$$[\![\varphi]\!] := \varphi^1 \boldsymbol{n}^1 + \varphi^2 \boldsymbol{n}^2 \tag{9.12}$$

Analogously, let $\boldsymbol{\tau}$ be a vector field, piecewise smooth on $\mathcal{T}_h$, therefore on $e \in \mathcal{E}_h^o$:

$$\langle \boldsymbol{\tau} \rangle := \frac{1}{2}(\boldsymbol{\tau}^1 + \boldsymbol{\tau}^2) \tag{9.13}$$

$$[\![\boldsymbol{\tau}]\!] := \boldsymbol{\tau}^1 \cdot \boldsymbol{n}^1 + \boldsymbol{\tau}^2 \cdot \boldsymbol{n}^2 \tag{9.14}$$

Notice that, by definition of "[\![ ]\!]", the jump of a scalar quantity is a vector and the jump of a vector quantity is a scalar. Definitions (9.12) and (9.14) do not depend on the ordering of the elements T_i.

However, because the formulations to be analyzed involve advection (*skew-symmetric* with respect to sign reversal), $\varphi^-/\boldsymbol{\tau}^-$ (resp. $\varphi^+/\boldsymbol{\tau}^+$) will be used throughout to indicate the *upwind* (resp. *downwind*) values of the the scalar/vector field $\varphi/\boldsymbol{\tau}$ with respect to edge e (see Fig. 9.2 for a sketch of the geometrical setting). This definition is obviously

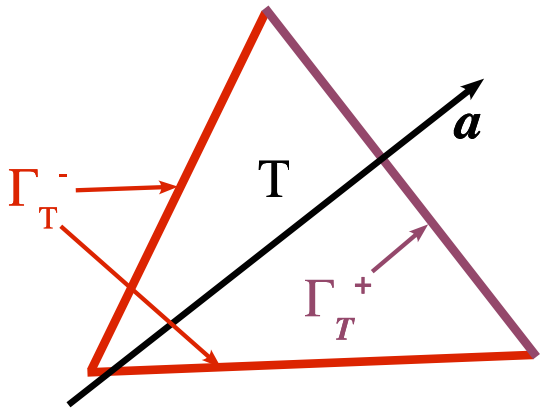

Figure 9.3: Schematics of the inflow and outflow boundaries for an element with respect to the convective field $\boldsymbol{a}$.

not invariant to ordering of the elements, because advection is not symmetric with respect to orientation reversal.

It is now important to specialize the previous formulas to the edges/faces on the boundary Γ of Ω.

$$[\![\varphi]\!] = \varphi\ \boldsymbol{n}, \qquad \langle \boldsymbol{\tau} \rangle = \boldsymbol{\tau}, \qquad \forall e \in \Gamma \tag{9.15}$$

It will not be necessary to specify $\langle \varphi \rangle$ and $[\![\boldsymbol{\tau}]\!]$ on the boundary Γ, and their value there can therefore remain undefined. By observing that

$$\begin{aligned}
[\![\varphi\boldsymbol{\tau}]\!] &= \varphi^+\boldsymbol{\tau}^+ \cdot \boldsymbol{n}^+ + \varphi^-\boldsymbol{\tau}^- \cdot \boldsymbol{n}^- \\
&= \frac{1}{2}\left(2\varphi^+\boldsymbol{\tau}^+ \cdot \boldsymbol{n}^+ + 2\varphi^-\boldsymbol{\tau}^- \cdot \boldsymbol{n}^-\right) \\
&= \frac{\varphi^+ + \varphi^-}{2}\left(\boldsymbol{\tau}^+ \cdot \boldsymbol{n}^+ + \boldsymbol{\tau}^- \cdot \boldsymbol{n}^-\right) + \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2} \cdot \left(\varphi^+\boldsymbol{n}^+ + \varphi^-\boldsymbol{n}^-\right) \\
&= \langle\varphi\rangle[\![\boldsymbol{\tau}]\!] + \langle\boldsymbol{\tau}\rangle \cdot [\![\varphi]\!]
\end{aligned} \tag{9.16}$$

and accounting for (9.15), the following *unbiased* or *ordering invariant* identity, which is very useful in gDG formulations, is readily obtained:

$$\sum_{T\in\mathcal{T}_h} \int_{\partial T} \boldsymbol{\tau}\cdot\boldsymbol{n}\ \varphi = \sum_{e\in\mathcal{E}_h} \int_e \langle\boldsymbol{\tau}\rangle \cdot [\![\varphi]\!] + \sum_{e\in\mathcal{E}_h^o} \int_e [\![\boldsymbol{\tau}]\!]\langle\varphi\rangle \tag{9.17}$$

Also, when manipulating expressions involving the convective fluxes, the following observation will prove very useful:

$$\begin{aligned}
[\![\varphi\boldsymbol{\tau}]\!] &= \varphi^+\boldsymbol{\tau}^+\cdot\boldsymbol{n}^+ + \varphi^-\boldsymbol{\tau}^-\cdot\boldsymbol{n}^- \\
&= \varphi^+\boldsymbol{\tau}^+\cdot\boldsymbol{n}^+ + \varphi^\pm\boldsymbol{\tau}^\mp\cdot\boldsymbol{n}^- + \varphi^\pm\boldsymbol{\tau}^\mp\cdot\boldsymbol{n}^+ + \varphi^-\boldsymbol{\tau}^-\cdot\boldsymbol{n}^- \\
&= \varphi^\pm[\![\boldsymbol{\tau}]\!] + [\![\varphi]\!]\cdot\boldsymbol{\tau}^\mp
\end{aligned} \tag{9.18}$$

which implies the following integral identity:

$$\sum_{T\in\mathcal{T}_h}\int_{\partial T}\boldsymbol{\tau}\cdot\boldsymbol{n}\;\varphi = \sum_{e\in\mathcal{E}_h^o}\left(\int_e\varphi^\pm[\![\boldsymbol{\tau}]\!] + \int_e[\![\varphi]\!]\cdot\boldsymbol{\tau}^\mp\right) + \sum_{e\in\Gamma}\int_e\varphi\;\boldsymbol{\tau}\cdot\boldsymbol{n} \tag{9.19}$$

This last result will also be used to recover the Euler-Lagrange form of variational problems. Finally, the following approximation space is introduced:

$$V_h^k = \{v\in L^2(\Omega) : v|_T\in\mathcal{P}^k(T),\quad\forall T\in\mathcal{T}_h\} \tag{9.20}$$

where $\mathcal{P}^k$ is the space of polynomials of degree less than or equal to k.

Chapter 10

Global Weak Formulations

In this chapter a global DG (gDG) method is presented and will serve as an underlying framework for the solution of the advection/diffusion problem. Furthermore, it will be shown in chapter 11, that the local DG formulation (lDG) corresponds to the restriction of the gDG formulation to a single element. Therefore, the gDG formulation presented herein can be considered as the foundation of the entire method.

A *skew-symmetric*, a *neutral*, and a *symmetric* primal formulation of the gDG method will be introduced. They will be integrated in a single global formulation by introducing an integer *switch* s, taking the values +1, 0, and -1, respectively.

The symmetric form is the only one to yield a symmetric discretization of the Laplace operator, with very important consequences. In fact the symmetric form is the only one to be *adjoint consistent*, following the definition in Arnold et al. [1], with important implications on convergence rates. Methods that are not adjoint consistent cannot be proved to converge with optimal rates, and it will be shown in chapter 12 that both the skew-symmetric and neutral gDG formulations *do not* converge with optimal rates.

Furthermore, it will be shown numerically (again in chapter 12) that the symmetric discretization leads to monotone solutions inside the boundary layer regions, even for coarse meshes, while this is not the case for the other two discretizations.

10.1 A conservative primal formulation

One of the most important design requirements for DG formulations is conservation. In the present primal formulation a new approach is taken to enforce conservation of the flux $\boldsymbol{\sigma} := \boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h$ of the solution ϕ_h. The global DG formulation reads:

$$\begin{aligned}
0 \;=\; & \sum_{T\in\mathcal{T}_h}\left(-\int_T \boldsymbol{\nabla}\mu_h\cdot(\boldsymbol{a}\phi_h-\kappa\boldsymbol{\nabla}\phi_h)-\int_T\mu_h f\right)\\
& +\sum_{e\in\mathcal{E}_h^o}\int_e [\![\mu_h]\!]\cdot(\boldsymbol{a}\phi_h^- - \kappa\boldsymbol{\nabla}\phi_h^-) + s\;\kappa\nabla\mu_h^-\cdot[\![\phi_h]\!]\\
& +\sum_{e\in\Gamma_g}\int_e\Big(\mu_h\boldsymbol{a}\cdot\boldsymbol{n}(\chi_{\Gamma_g^-}g+\chi_{\Gamma_g^+}\phi_h)+s\;\kappa\boldsymbol{\nabla}\mu_h\cdot\boldsymbol{n}(\phi_h-g)-\kappa\boldsymbol{\nabla}\phi_h\cdot\boldsymbol{n}\mu_h\Big)\\
& +\sum_{e\in\Gamma_h}\int_e\mu_h\Big((\boldsymbol{a}\phi_h)\chi_{\Gamma_h^+}\cdot\boldsymbol{n}-h\Big)\\
& +\epsilon\sum_{e\in\mathcal{E}_h^o}\int_e\frac{\kappa}{h_\perp}[\![\mu_h]\!]\cdot[\![\phi_h]\!]+\epsilon\sum_{e\in\Gamma_g}\int_e\frac{\kappa}{h_\perp}\mu_h(\phi_h-g).
\end{aligned}\tag{10.1}$$

Note that s is an integer with sign, that can take the values $+1$ (skew-symmetric form of the discretized diffusion operator), -1 (symmetric form) or 0 (neutral form). The non-dimensional parameter ϵ is to be chosen small, and the following definition will be used:

$$h_\perp = \frac{\text{meas}(T_+)+\text{meas}(T_-)}{2\ \text{meas}(e)}\tag{10.2}$$

where T_-/T_+ are the *upwind/downwind* elements with respect to the edge e. Roughly speaking, $h_\perp$ is a length scale in the direction perpendicular to the edge e, of size close to the segment joining the barycenters of $T_\mp$.

Euler-Lagrange equations for the global problem

In order to make sense of the proposed formulation, it is very instructive to recover the Euler-Lagrange equations by means of an integration-by-parts.

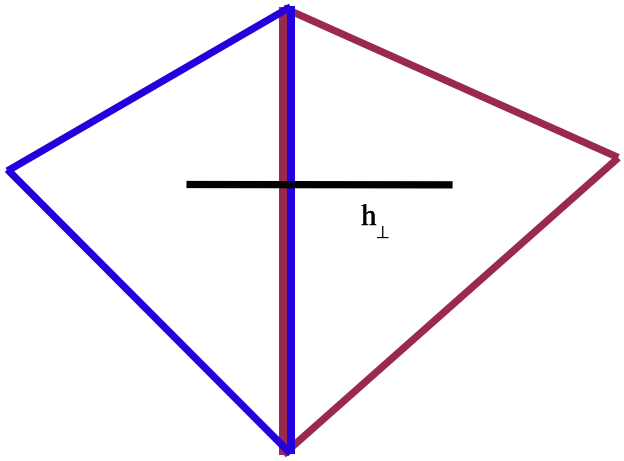

Figure 10.1: Sketch portraying the $h_\perp$ definition for two triangular adjacent triangular elements.

Use of identity (9.17) with $\boldsymbol{\tau} = \kappa\boldsymbol{\nabla}\phi_h$ and identity (9.19) with $\boldsymbol{\tau} = \boldsymbol{a}\phi_h$ in (10.1) yields:

$$
\begin{aligned}
0 \;=\; & \sum_{T\in\mathcal{T}_h}\left(\int_T \mu_h\left(\boldsymbol{\nabla}\cdot\left(\boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h\right) - f\right)\right) \\
& + \sum_{e\in\mathcal{E}_h^o}\int_e -\mu_h^+ [\![\boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h]\!] + s\;\kappa\boldsymbol{\nabla}\mu_h^- \cdot [\![\phi_h]\!] \\
& + \sum_{e\in\Gamma_g}\int_e (-\mu_h\boldsymbol{a}\chi_{\Gamma_g^-} + s\;\kappa\boldsymbol{\nabla}\mu_h)\cdot\boldsymbol{n}(\phi_h - g) \\
& + \sum_{e\in\Gamma_h}\int_e \mu_h\left((-\boldsymbol{a}\phi_h\chi_{\Gamma_h^-} + \kappa\boldsymbol{\nabla}\phi_h)\cdot\boldsymbol{n} - h\right) \\
& + \epsilon\left(\sum_{e\in\mathcal{E}_h^o}\int_e \frac{\kappa}{h_\perp}[\![\mu_h]\!]\cdot[\![\phi_h]\!] + \sum_{e\in\Gamma_g}\int_e \frac{\kappa}{h_\perp}\mu_h(\phi_h - g)\right)
\end{aligned}
\tag{10.3}
$$

The reader can easily see how the global DG formulation has been designed:

- The first term in the second line of (10.3) is enforcing continuity of the total flux across internal element interfaces. In particular, it is easy to realize that the term represents an *upwinded total flux*, since the jump of the fluxes upwind and downwind of an edge are weighted by the downwind test function μ_h^+. Therefore, in one single step, the fluxes are conserved and "upwinded".

- In the third line of (10.3), Dirichlet boundary conditions are weakly enforced and tested against the *test* total flux (the total flux constructed with the test function μ_h) at the inflow, while at the outflow they are tested only against the adjoint viscous flux. This choice has a natural meaning when analyzing two important limit cases: In the advection dominated limit, the outflow boundary condition is severely weakened, so that the flow is substantially influenced by the inflow condition, as expected in the purely hyperbolic case; when diffusion dominates, the problem gains strong ellipticity, so that the boundary condition must be felt everywhere in the Dirichlet portion of the boundary.

- In the fourth line of (10.3), Neumann conditions are imposed according to the same rationale for the Dirichlet conditions. The total flux $\boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h$ is imposed at the inflow, while only the diffusive flux is specified at the outflow.

- The terms multiplied by the parameter ϵ, in the fifth line of (10.3), serve the purpose of eliminating a nontrivial kernel in the discrete diffusive operator, in the limit as $\boldsymbol{a} \to \boldsymbol{0}$.

- There is an extra interior penalty term $\sum_{e\in\mathcal{E}_h^o} \int_e \kappa\boldsymbol{\nabla}\mu_h^- \cdot [\![\phi_h]\!]$ that arises from consistency requirements. This is a residual term, in the sense that substitution of the exact (smooth) solution causes it to vanish.

Chapter 11

Local Formulations

The following discussion represents the core of the entire second part of the presented dissertation, and a few words are felt necessary to guide the reader through the steps involved.

So far, a global DG formulation has been introduced, and three variants have been outlined. The key of the method is now to *condense* degrees-of-freedom (DOF) at a local (element) level, in order to obtain a reduced representation of the global system by means of the data structure typical of the Continuous Galerkin (CG) method.

The gain in terms of both memory storage and computational time is therefore surprisingly high. For example, for linear isoparametric elements on a three-dimensional tetrahedral mesh, the ratio of the number of DOF per mesh node for a DG method versus the corresponding CG method is 28.

The framework in which the goal of DOF reduction will be achieved is a multiscale analysis of the gDG method, in which the discontinuous, *discrete* solution ϕ_h is decomposed into a *continuous part* $\overline{\phi}_h$ and a *discontinuous correction* ϕ'_h.

At a local level, the discontinuous correction ϕ'_h is condensed in terms of the continuous part $\overline{\phi}_h$ by means of local DG problems in which weak continuity constraint is imposed between ϕ_h and $\overline{\phi}_h$.

The condensed discontinuous solution $\phi_h = \overline{\phi}_h + \phi'_h$ can then be substituted into the global formulation, and the final result is a system in which only the continuous DOF relative to $\overline{\phi}_h$ are to be computed.

The local problems may be then used to reconstruct ϕ_h, from the computed $\overline{\phi}_h$.

For symmetry purposes, condensation is also applied to the local test functions. From numerical experiments, it appears clear that this step is responsible for preserving the accuracy of the method. The condensation of the DG test functions is performed without introducing the body-force term in the local problems, in order to generate a standard square system of linear algebraic equations.

In the one-dimensional case, it will be shown that the overall procedure not only is very effective in reconstructing the DG solution, but also it usually yields better quality (i.e., more stable, more accurate) solutions, even with respect to a full-scale global computation, in which no condensation is performed. The reason for this amazing fact is that the local problem acts as a stabilization for the gDG formulation. As will become clear from the following analysis, the method forces the discontinuous correction to vanish in the diffusion dominated limit without forcing, so that the global formulation tends to the continuous one, as viscosity dominates. As the numerical computations will demonstrate, even non-self-adjoint consistent methods deliver optimal convergence rates, when the proposed condensation is applied. The quality of the solution for non-self-adjoint methods is still doomed by lack of monotonicity whenever coarse meshes are used to capture boundary layers, but at least from the point of view of convergence rates, problems disappear.

11.1 Local problem I: linking the continuous and discontinuous solutions

Two approaches to the local DG (lDG) problem will be presented. The first approach is based purely on the exchange of "information" from $\overline{\phi}_h$ to ϕ_h along the boundaries of the element by means of *weakly enforced continuity constraints*. The second, instead, is based on a *full multiscale decomposition* of the discrete solution ϕ_h and can be applied to any type of finite element discretization. It is important to realize the equivalence between the two approaches when the local basis is such that the local CG/DG solutions can be uniquely defined by their traces on element edges (e.g., isoparametric elements on triangles and tetrahedra, or serendipity elements on quadrilateral or hexahedral meshes).

11.1.1 Local problem solution via weak continuity constraints

In order to link the discontinuous field ϕ_h to a continuous field $\overline{\phi}_h$ (amenable to a much less expensive data structure) the following local (i.e., element-by-element) DG problem is solved, imposing weak continuity between ϕ_h and $\overline{\phi}_h$ on Γ_T:

Find $\phi_h \in V_h^k(T)$ *such that,* $\forall v \in V_h^k(T)$*:*

$$\begin{aligned} 0 &= -\int_T \boldsymbol{\nabla} v \cdot (\boldsymbol{a}\phi_h - \kappa \boldsymbol{\nabla}\phi_h) - \int_T vf + \epsilon \int_{\Gamma_T} \frac{\kappa}{h_\perp}\, v\, (\phi_h - \overline{\phi}_h) \\ &\quad + \int_{\Gamma_T^+} v\phi_h \boldsymbol{a}\cdot\boldsymbol{n} + \int_{\Gamma_T^-} v\overline{\phi}_h \boldsymbol{a}\cdot\boldsymbol{n} \\ &\quad + \int_{\Gamma_T} s\ \kappa \boldsymbol{\nabla} v \cdot \boldsymbol{n}(\phi_h - \overline{\phi}_h) - \int_{\Gamma_T} \kappa \nabla\phi_h \cdot \boldsymbol{n} v \end{aligned} \tag{11.1}$$

where v is a *local* test function. As usual, the local formulation is best understood once the Euler-Lagrange equations are presented:

$$\begin{aligned} 0 &= \int_T v\,(\boldsymbol{\nabla}\cdot(\boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h) - f) + \epsilon \int_{\Gamma_T} \frac{\kappa}{h_\perp}\, v\, (\phi_h - \overline{\phi}_h) \\ &\quad + \int_{\Gamma_T^-} v\boldsymbol{a}\cdot\boldsymbol{n}(\overline{\phi}_h - \phi_h) - \int_{\Gamma_T} s\ \kappa\boldsymbol{\nabla} v\cdot\boldsymbol{n}(\overline{\phi}_h - \phi_h) \end{aligned} \tag{11.2}$$

More succinctly, (11.1) can be expressed as:

Find $\phi_h \in V_h^k(T)$ *such that,* $\forall v \in V_h^k(T)$*:*

$$B(v,\phi_h) = F(v; f, \overline{\phi}_h) \tag{11.3}$$

with

$$\begin{aligned} B(v,\phi_h) &= -\int_T \boldsymbol{\nabla} v\cdot(\boldsymbol{a}\phi_h - \kappa\boldsymbol{\nabla}\phi_h) + \epsilon\int_{\Gamma_T} \frac{\kappa}{h_\perp} v\phi_h + \int_{\Gamma_T^+} v\phi_h \boldsymbol{a}\cdot\boldsymbol{n} \\ &\quad + \int_{\Gamma_T} s\ \kappa\boldsymbol{\nabla} v\cdot\boldsymbol{n}\phi_h - \int_{\Gamma_T} \kappa\nabla\phi_h\cdot\boldsymbol{n} v \end{aligned} \tag{11.4}$$

$$F(v; f, \overline{\phi}_h) = \int_T vf + B_\Gamma(v, \overline{\phi}_h) \tag{11.5}$$

where $B(v,\phi_h)$ is the bilinear form generated by the DG method, and $F(v; f, \overline{\phi}_h)$ is a linear form in v which is also linear both in $\overline{\phi}_h$ and f (therefore trilinear in v, $\overline{\phi}_h$, and f)

In particular, by comparison, it can be easily seen that

$$B_\Gamma(v,\overline{\phi}_h) \;=\; \epsilon\int_{\Gamma_T}\frac{\kappa}{h_\perp}v\overline{\phi}_h - \int_{\Gamma_T^-} v\overline{\phi}_h\boldsymbol{a}\cdot\boldsymbol{n} + \int_{\Gamma_T} s\;\kappa\boldsymbol{\nabla}v\cdot\boldsymbol{n}\overline{\phi}_h \tag{11.6}$$

Once a local basis is selected for both the test and trial functions, equation (11.3) yields a system of algebraic equations. Applying a Galerkin discretization yields:

$$v \;=\; \sum_j v_j\psi_j \tag{11.7}$$

$$\phi_h \;=\; \sum_j \Phi_j\psi_j \tag{11.8}$$

$$\overline{\phi}_h \;=\; \sum_j \overline{\Phi}_j\psi_j \tag{11.9}$$

$$f \;=\; \sum_j f_j\psi_j \tag{11.10}$$

where ψ_j are the *local* test/trial functions and v_j, Φ_j, and $\overline{\Phi}_j$ are the *local* DOF for the test function, the $\phi_h|_T$, and $\overline{\phi}_h|_T$ solution, respectively. Using linear independence of the basis, (11.3) reduces to

$$\boldsymbol{S}\;\boldsymbol{\Phi}_{el} \;=\; \overline{\boldsymbol{S}}\;\overline{\boldsymbol{\Phi}}_{el} \;+\; \boldsymbol{M}\;\boldsymbol{f}_{el} \tag{11.11}$$

$$S_{ij} \;=\; B(\psi_i,\psi_j) \tag{11.12}$$

$$\overline{S}_{ij} \;=\; B_\Gamma(\psi_i,\psi_j) \tag{11.13}$$

$$M_{ij} \;=\; \int_T \psi_i\psi_j \tag{11.14}$$

where $\overline{\boldsymbol{\Phi}}_{el} = [\overline{\Phi}_1,\overline{\Phi}_2,\ldots,\overline{\Phi}_{n_{\rm el}}]^t$ and $\boldsymbol{\Phi}_{el} = [\Phi_1,\Phi_2,\ldots,\Phi_{n_{\rm el}}]^t$ are the vectors for the local DOF, and $\boldsymbol{f}_{el} = [f_1,f_2,\ldots,f_{n_{\rm el}}]^t$ is the vector of corresponding (local) body-force values.

In order to solve problem (11.11), the matrix $\boldsymbol{S}$ must be invertible. Hence, for higher-order elements, with interior nodes, $\overline{\boldsymbol{\Phi}}_{el}$ will not include interior DOF. Consequently this method fails. However, it is applicable to the class of isoparametric elements that includes triangular/tetrahedral elements, and quadrilateral/hexahedral serendipity elements up to quadratics.

Provided $\boldsymbol{S}$ is invertible, it is possible to express $\boldsymbol{\Phi}_{el}$ in terms of $\overline{\boldsymbol{\Phi}}_{el}$ and $\boldsymbol{f}_{el}$:

$$\boldsymbol{\Phi}_{el} = \boldsymbol{T}^h_{\phi_h \overline{\phi}_h} \, \overline{\boldsymbol{\Phi}}_{el} + \boldsymbol{T}^h_{\phi_h f} \, \boldsymbol{f}_{el} \tag{11.15}$$

where $\boldsymbol{T}^h_{\phi_h \overline{\phi}_h} = \boldsymbol{S}^{-1} \, \overline{\boldsymbol{S}}$ and $\boldsymbol{T}^h_{\phi_h f} = \boldsymbol{S}^{-1} \, \boldsymbol{M}$. We refer to this expression as the *resolvent*.

11.1.2 Local problem solution via multiscale decomposition

The following discussion describes a more general *multiscale* technique applicable under the condition that the number of local DOF is the same for both $\phi_h|_T$ and $\overline{\phi}|_T$, and the degrees-of-freedom are defined in such a way that the resulting matrix $\boldsymbol{S}$ is invertible. This is the case for p-refinement on higher-order isoparametric elements in which the local degrees-of-freedom for the continuous and discontinuous parts of the solution are exactly the same (locally), so that invertibility is straightforward. Other choices are possible, with fairly general finite element discretizations, even *outside* the isoparametric class.

The alternative approach consists in splitting the solution as $\phi_h = \overline{\phi}_h + \phi'_h$, and using bilinearity of $B(\cdot, \cdot)$ to perform the following manipulation:

$$B(v, \phi_h) \;\; = \;\; B(v, \overline{\phi}_h + \phi'_h) \;\; = \;\; B(v, \overline{\phi}_h) + B(v, \phi'_h) \tag{11.16}$$

so that the original problem (11.3) is recast as:

Find $\phi'_h \in V^k_h(T)$ *such that,* $\forall v \in V^k_h(T)$*:*

$$B(v, \phi'_h) \;\; = \;\; Res(v; f, \overline{\phi}_h) \tag{11.17}$$

where $Res(v; f, \overline{\phi}_h)$ is the residual of the *continuous* (coarse-scale) solution. Notice in particular that the term $Res(v; f, \overline{\phi}_h)$ reads:

$$Res(v; f, \overline{\phi}_h) = F(v; f, \overline{\phi}_h) - B(v, \overline{\phi}_h) \tag{11.18}$$

or, specifically,

$$Res(v; f, \overline{\phi}_h) = -\int_T \left(\boldsymbol{\nabla} v \cdot \left(\boldsymbol{a} \overline{\phi}_h - \kappa \boldsymbol{\nabla} \overline{\phi}_h \right) + v f \right) + \int_{\Gamma_T} v (\boldsymbol{a} \overline{\phi}_h - \kappa \nabla \overline{\phi}_h) \cdot \boldsymbol{n} \tag{11.19}$$

Comparing (11.19) with (11.4) and (11.5), it is immediately realized that the local problem for the discontinuous correction ϕ'_h corresponds to the localized DG method with weakly enforced *homogeneous* Dirichlet boundary conditions and a a right hand side driven by the residual of the coarse-scale component $\overline{\phi}_h$. The strict relationship with the multiscale analysis as presented in Hughes et al. [34] is then evident.

The similarity can be appreciated more easily in the local algebraic problem,

$$\boldsymbol{S}\ \boldsymbol{\Phi}'_{el} \ = \ \overline{\boldsymbol{S}}\ \overline{\boldsymbol{\Phi}}_{el} \ - \boldsymbol{S}\ \overline{\boldsymbol{\Phi}}_{el} \ + \ \boldsymbol{M}\ \boldsymbol{f}_{el} \tag{11.20}$$

from which:

$$\boldsymbol{\Phi}'_{el} \ = \ \boldsymbol{T}^h_{\phi'_h \overline{\phi}_h}\ \overline{\boldsymbol{\Phi}}_{el} + \boldsymbol{T}^h_{\phi_h f}\ \boldsymbol{f}_{el} \tag{11.21}$$

with

$$\boldsymbol{T}^h_{\phi'_h \overline{\phi}_h} \ = \ \boldsymbol{T}^h_{\phi_h \overline{\phi}_h} - \boldsymbol{I}_{n_{\text{lDOF}} \times n_{\text{lDOF}}} \tag{11.22}$$

where n_{lDOF} is the number of local DOF.

11.2 Local problem II: linking the continuous and discontinuous test functions

The discontinuous test field μ_h is also linked to a continuous test field $\overline{\mu}_h$. From an algebraic point of view, unless this link is performed, the matrix generated by the linear problem in question will not be square (and therefore not invertible).

Weak continuity constraint approach

Find $\mu_h \in V_h^k(T)$ *such that,* $\forall v \in V_h^k(T)$*:*

$$B(v, \mu_h) \ = \ F(v; 0, \overline{\mu}_h) \tag{11.23}$$

Multiscale approach

Find $\mu'_h \in V_h^k(T)$ *such that,* $\forall v \in V_h^k(T)$*:*

$$B(v, \mu'_h) \quad = \quad Res(v; 0, \overline{\mu}_h) \tag{11.24}$$

The choice $f = 0$ is dictated by the fact that it leads to a square algebraic system for the global formulation. Inserting the f would have produced a constrained algebraic problem whose solution feasibility is unclear.

Chapter 12

Numerical Experiments and Conclusions

The numerical experiments presented herein are intended to outline the implementation details of the method and, most importantly, to assess its overall performance. As it will be clear from the analysis of the results, the proposed approach allows an intrinsic stabilization of the underlying global DG formulation, even in cases in which adjoint consistency, hence L^2 error estimates, are not ensured.

The detailed analysis of the one-dimensional case will also highlight the nature of the local problem resolvent mapping, which links the discontinuous and continuous solutions. As will be clear from the derivations, the method circumvents the adjoint consistency issues since, in the limit for diffusion dominated flows (in the absence of body-force), the affine mapping tends to the identity, forcing the full solution ϕ_h to become continuous, and identical to $\overline{\phi}_h$.

As a consequence, the solution improves dramatically, since continuous discretizations are much better suited for the Laplace operator than discontinuous ones.

12.1 One-dimensional scalar equation

Without loss of generality, it will be assumed throughout the following sections $\boldsymbol{a} = a\boldsymbol{e}_x$, with a positive. It will also be assumed, for simplicity, $f = c_f$, constant.

The exact solution of the strong form (9.3) is therefore easily computed as:

$$\phi(x) \;=\; \phi_0 + (\phi_L - \phi_0)\frac{1 - e^{Pe_L \frac{x}{L}}}{1 - e^{Pe_L}} + \frac{2\mathcal{F}}{Pe_L}\left(\frac{x}{L} - \frac{1 - e^{Pe_L \frac{x}{L}}}{1 - e^{Pe_L}}\right) \tag{12.1}$$

where ϕ_0 and ϕ_L are the Dirichlet boundary conditions imposed at $x = 0$ and $x = L$, $Pe_L = |a|L/\kappa$, and

$$\mathcal{F} \;=\; \frac{c_f\, L^2}{2\kappa} \tag{12.2}$$

is a non-dimensional body-force coefficient. In the limit $Pe_L \to 0$, (12.1) yields:

$$\phi(x) \;=\; \phi_0 + (\phi_L - \phi_0 + \mathcal{F})\,\frac{x}{L} - \mathcal{F}\left(\frac{x}{L}\right)^2 \tag{12.3}$$

12.1.1 Weak (primal) formulation in one dimension

It is now worthwhile to recast (10.1) for the one-dimensional case, since many simplifications arise. For simplicity, Neumann-type boundary conditions have been omitted:

Find $\phi_h \in V_h^k([x_e, x_{e+1}])$, $e \in \{1, 2, \ldots, n_{\text{el}}\}$, *such that,* $\forall \mu_h \in V_h^k([x_e, x_{e+1}])$*:*

$$\begin{aligned}
0 \;=\; & -\sum_{el=1}^{n_{\text{el}}} \int_{x_e}^{x_{e+1}} \left(\partial_x \mu_h \left(a\phi_h - \kappa\, \partial_x \phi_h\right) + \mu_h f \right) \\
& + \sum_{e=2}^{n_{\text{el}}} \Big\{ (-\mu_h^+ + \mu_h^-)(a\phi_h^- - \kappa\, \partial_x \phi_h^-) + s\, \kappa\, \partial_x \mu_h^- (-\phi_h^+ + \phi_h^-) \\
& \qquad + \epsilon\kappa/h_\perp (-\mu_h^+ + \mu_h^-)(-\phi_h^+ + \phi_h^-) \Big\}_{x=x_e} \\
& + \left\{ +\mu_h a\phi_h + \epsilon \frac{\kappa}{h_\perp} \mu_h (\phi_h - \phi_L) + s\, \kappa\, \partial_x \mu_h (\phi_h - \phi_L) - \kappa\, \partial_x \phi_h\, \mu_h \right\}_{x=L} \\
& + \left\{ -\mu_h a\phi_0 + \epsilon \frac{\kappa}{h_\perp} \mu_h (\phi_h - \phi_0) - s\, \kappa\, \partial_x \mu_h (\phi_h - \phi_0) + \kappa\, \partial_x \phi_h\, \mu_h \right\}_{x=0}
\end{aligned} \tag{12.4}$$

where the notation $\{\eta\}_{x=\tilde{x}}$ stands for η evaluated at $\tilde{x}$, $e \in \{1, 2, \ldots, n_{\text{np}}\}$ are the nodes of the mesh, and $\{x_e |\ e = 2, \ldots, n_{\text{np}} - 1 = n_{\text{el}}\}$ is the set of interior nodes.

12.1.2 Local problem I: from ϕ_h to $\overline{\phi}_h$

A fully detailed analysis of the local problems will be presented next, including limit behavior and feasibility regions for the parameter ϵ.

Weak continuity constraint approach

The local problem reads:

$$B(v,\phi_h) \;=\; F(v;f,\overline{\phi}_h) \tag{12.5}$$

with

$$\begin{aligned} B(v,\phi_h) \;=\; & -\int_{x_e}^{x_{e+1}} \partial_x v\ (a\phi_h - \kappa\ \partial_x\phi_h) \\ & + \left\{ v\ a\phi_h + \epsilon\frac{\kappa}{h_\perp} v\ \phi_h + s\ \kappa\ \partial_x v\ \phi_h - \kappa\ \partial_x\phi_h\ v\ \right\}_{x=x_{e+1}} \\ & + \left\{ \epsilon\frac{\kappa}{h_\perp} v\ \phi_h - s\ \kappa\ \partial_x v\ \phi_h + \kappa\ \partial_x\phi_h\ v\ \right\}_{x=x_e} \end{aligned} \tag{12.6}$$

$$F(v;f,\overline{\phi}_h) \;=\; \int_T vf + B_\Gamma(v,\overline{\phi}_h) \tag{12.7}$$

$$\begin{aligned} B_\Gamma(v,\overline{\phi}_h) \;=\; & + \left\{ \epsilon\frac{\kappa}{h_\perp} v\ \overline{\phi}_h + s\ \kappa\ \partial_x v\ \overline{\phi}_h \right\}_{x=x_{e+1}} \\ & + \left\{ -v\ a\overline{\phi}_h + \epsilon\frac{\kappa}{h_\perp} v\ \overline{\phi}_h - s\ \kappa\ \partial_x v\ \overline{\phi}_h \right\}_{x=x_e} \end{aligned} \tag{12.8}$$

Piecewise linear interpolation will be used for the continuous and discontinuous parts of the solution, as well as the test functions. Once the local problem is solved, the discontinuous solution ϕ_h can be linked to its continuous part $\overline{\phi}_h$ and the forcing term f by means of a resolvent mapping. Let

$$\boldsymbol{\Phi}_{el} = \begin{bmatrix} \phi_h^l \\ \phi_h^r \end{bmatrix}, \qquad \overline{\boldsymbol{\Phi}}_{el} = \begin{bmatrix} \overline{\phi}_h^l \\ \overline{\phi}_h^r \end{bmatrix}, \qquad \boldsymbol{f}_{el} = \begin{bmatrix} f^l \\ f^r \end{bmatrix} \tag{12.9}$$

where the index l and r stand for the *left* and *right* value of the element domain, the local resolvent affine mapping is:

$$\boldsymbol{\Phi}_{el} = \boldsymbol{T}^h_{\phi\overline{\phi}}\ \overline{\boldsymbol{\Phi}}_{el} + \boldsymbol{T}^h_{\phi f}\ \boldsymbol{f}_{el} \tag{12.10}$$

Calculations allow $\boldsymbol{T}^h_{\phi\overline{\phi}}$ and $\boldsymbol{T}^h_{\phi f}$ to be expressed as

$$\boldsymbol{T}^h_{\phi\overline{\phi}} = \frac{1/2}{\Delta(Pe_h, s, \epsilon)} \begin{bmatrix} (2Pe_h+\epsilon)(Pe_h+2s+\epsilon) & -(2s+\epsilon)Pe_h \\ Pe_h(2Pe_h+2s+\epsilon) & \epsilon(Pe_h+2s+\epsilon) \end{bmatrix} \tag{12.11}$$

$$\boldsymbol{T}^h_{\phi f} = \frac{h^2/(12\kappa)}{\Delta(Pe_h, s, \epsilon)} \begin{bmatrix} Pe_h+3s+2\epsilon & -Pe_h+3s+\epsilon \\ 3Pe_h+3s+\epsilon & 3Pe_h+3s+2\epsilon \end{bmatrix} \tag{12.12}$$

where

$$\Delta(Pe_h, s, \epsilon) = Pe_h^2 + (s+\epsilon)Pe_h + \epsilon(s+\epsilon/2) \tag{12.13}$$

$$Pe_h = \frac{|a|h}{2\kappa} \tag{12.14}$$

The previous expressions account for the possible choices of s, and all three forms will

	$s=+1$	$s=0$	$s=-1$
Δ	$Pe_h(Pe_h+1)$ $+\epsilon(Pe_h+1)+\epsilon^2/2$	Pe_h^2 $+(Pe_h+\epsilon)^2$	$Pe_h(Pe_h-1)$ $+\epsilon(Pe_h-1)+\epsilon^2/2$
$\Delta>0$?	always for $\epsilon>0$	always for $\epsilon>0$	for $\epsilon>1-Pe_h+\sqrt{1-Pe_h^2}$ and $\epsilon<1-Pe_h-\sqrt{1-Pe_h^2}$

Table 12.1: Analysis on the sign of the determinant $\Delta(Pe_h, s, \epsilon)$.

be compared in a later section.

Notice that special care has to be taken for the determinant, since for $s=-1$, it can vanish for certain combinations of Pe_h and ϵ. An analysis of the sign of the determinant is presented in Table 12.1, assuming $a>0 \Rightarrow Pe_h>0$. The locus of $\Delta=0$ in the plane $[Pe_h/\epsilon]$ is shown in Figure 12.1, together with a three-dimensional plot of the function $\Delta(Pe_h, \epsilon)$.

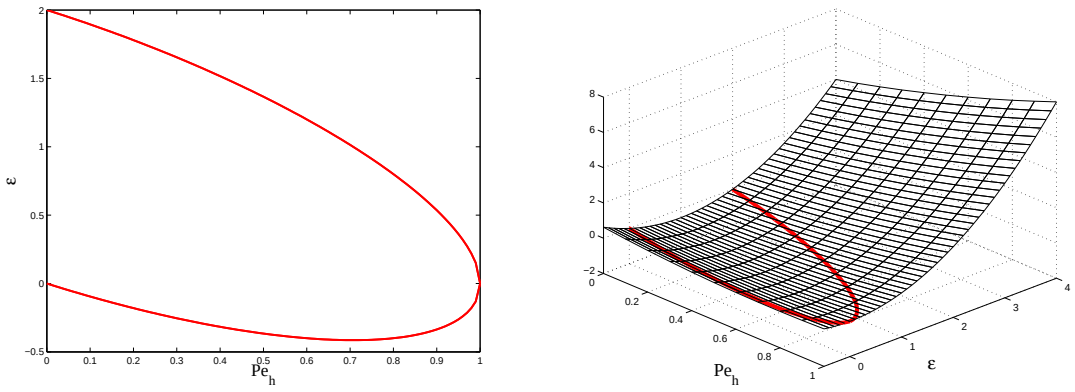

Figure 12.1: Locus of $\Delta = 0$ for $s = -1$ on the $[Pe_h/\epsilon]$-plane (left) and on a three dimensional view of the function $\Delta(Pe_h, \epsilon)$ (right). It is easily seen that $\epsilon > 2$ prevents the determinant to vanish for any Péclet number.

Multiscale approach

Since the elements are isoparametric and the degrees-of-freedom of the local solution are its nodal values, the multiscale approach is perfectly equivalent to the weak continuity constraint approach. Only the symbolic form and the final mapping will be therefore presented, for the sake of completeness.

$$\boldsymbol{\Phi}'_{el} \;=\; \boldsymbol{T}^h_{\phi'_h\overline{\phi}_h}\,\overline{\boldsymbol{\Phi}}_{el} + \boldsymbol{T}^h_{\phi_h f}\,\boldsymbol{f}_{el} \tag{12.15}$$

with $\boldsymbol{T}^h_{\phi'_h\overline{\phi}_h} = \boldsymbol{T}^h_{\phi_h\overline{\phi}_h} - \boldsymbol{I}_{2\times 2}$ given by:

$$\boldsymbol{T}^h_{\phi'_h\overline{\phi}_h} = \frac{1}{2\Delta}\begin{bmatrix} 2sPe_h - \epsilon(s+\epsilon) & -s(2Pe_h+\epsilon) \\ 2Pe_h^2 + s(2Pe_h-\epsilon) & -2Pe_h^2 - (2Pe_h+\epsilon)(s+\epsilon) \end{bmatrix} \tag{12.16}$$

12.1.3 Limit behavior of resolvent mapping

The aim of the present section is to clarify the nature of the local problem. In fact, it is important to consider the diffusion and advection dominated limits to grasp the nature of the local condensation. Taking limits:

$$\lim_{Pe_h\to 0} \boldsymbol{T}^h_{\phi_h\overline{\phi}_h} \;=\; \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{12.17}$$

$$\lim_{Pe_h \to \infty} \boldsymbol{T}^h_{\phi_h \overline{\phi}_h} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \tag{12.18}$$

$$\lim_{Pe_h \to 0} \boldsymbol{T}^h_{\phi_h f} = \frac{h}{6\kappa\epsilon(2s+\epsilon)} \begin{bmatrix} 3s+2\epsilon & 3s+\epsilon \\ 3s+\epsilon & 3s+2\epsilon \end{bmatrix} \tag{12.19}$$

$$\lim_{Pe_h \to \infty} \boldsymbol{T}^h_{\phi_h f} = \mathbf{0}_{2\times 2} \tag{12.20}$$

From (12.17) it is easily seen that, $\phi_h \to \overline{\phi}_h$ in the diffusive limit without body-force, while from (12.18) it can be observed that, in the advective limit, full upwinding is performed, that is $\phi_h|_{[x_e,x_{e+1}]} \to \overline{\phi}_h(x_e)\chi|_{[x_e,x_{e+1}]}$, for a positive. Notice also that, due to the fact that in the diffusive limit $\boldsymbol{T}^h_{\phi_h f}$ does not vanish, in general the continuous part of the solution $\overline{\phi}_h$ will not be equal to the full solution ϕ_h when a force f is present.

12.1.4 Local problem II: from μ to $\overline{\mu}$

Given the fact that the problem of linking μ to $\overline{\mu}$ is the same as the one of linking ϕ_h to $\overline{\phi}_h$, with the exception that the forcing term f is not accounted for, the result is

$$\begin{bmatrix} \mu^l_h \\ \mu^r_h \end{bmatrix} = \boldsymbol{T}^h_{\mu_h \overline{\mu}_h} \begin{bmatrix} \overline{\mu}^l_h \\ \overline{\mu}^r_h \end{bmatrix} \tag{12.21}$$

or

$$\begin{bmatrix} \mu'^l_h \\ \mu'^r_h \end{bmatrix} = \boldsymbol{T}^h_{\mu'_h \overline{\mu}_h} \begin{bmatrix} \overline{\mu}^l_h \\ \overline{\mu}^r_h \end{bmatrix} \tag{12.22}$$

with $\boldsymbol{T}^h_{\mu\overline{\mu}} = \boldsymbol{T}^h_{\phi\overline{\phi}}$ and $\boldsymbol{T}^h_{\mu'\overline{\mu}} = \boldsymbol{T}^h_{\phi'\overline{\phi}}$.

12.1.5 Numerical results

The numerical experiments show that the proposed method delivers very good performance when the symmetric ($s = -1$) formulation is used. Because of the lack of adjoint consistency in the global formulation, the quality of the solution in terms of accuracy and (possibly) monotonicity is not as good in the case of the neutral ($s = 0$) and skew-symmetric ($s = +1$) forms.

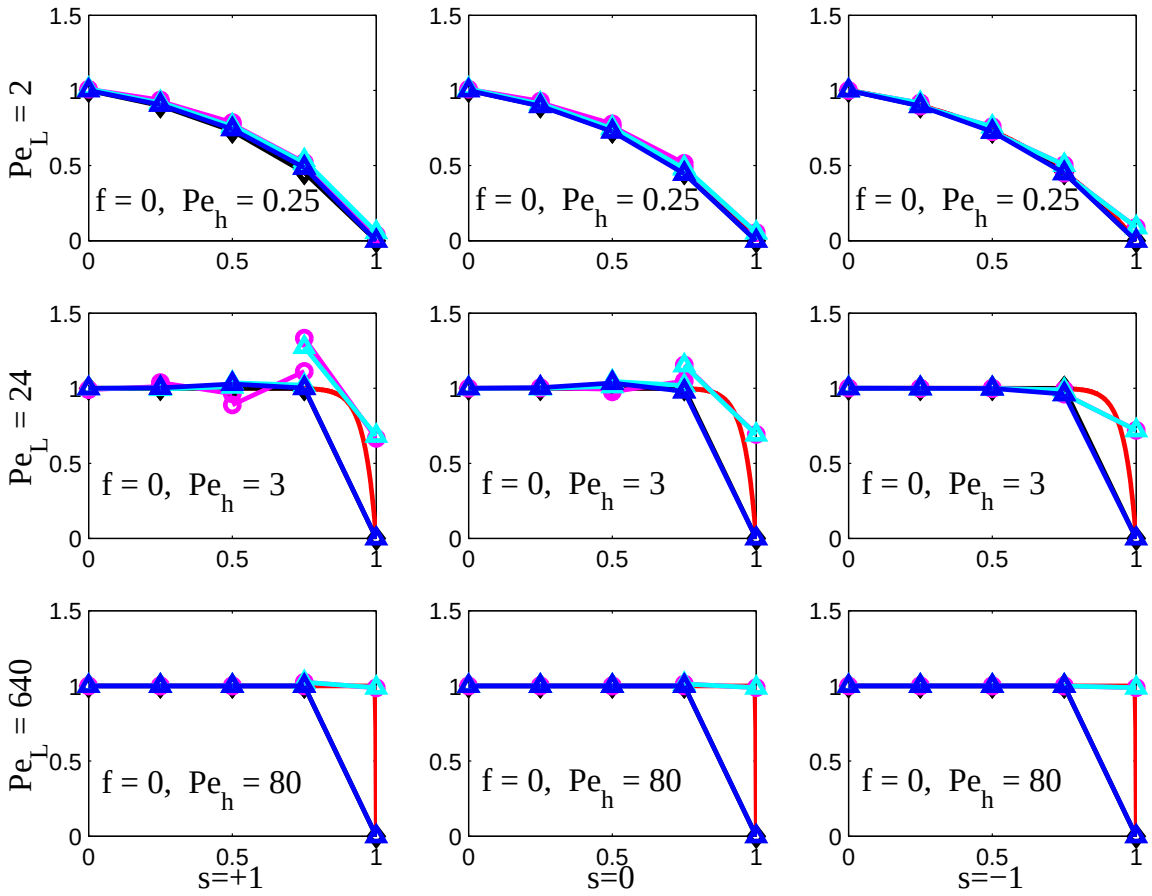

Figure 12.2: Solution plots in terms of varying Péclet number, on uniform grid of 4 elements, with no body-force: Left, $s = +1$ (skew); Middle, $s = 0$ (neutral); Right, $s = -1$ (symmetric). Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, exact solution; black, Galerkin SUPG-stabilized solution with optimal τ (almost always overlapped with the $\overline{\phi}_h$ solution).

Nevertheless, while the latter two gDG formulations do not even show second-order convergence rate in the L^2-norm, it is quite remarkable that optimal convergence rate is recovered in the corresponding version with local condensation. As is well understood, a continuous discretization is optimal for the Laplace equation, and the local condensation, as the mesh is refined, forces the discontinuous part ϕ_h closer and closer to the continuous part $\overline{\phi}_h$.

Effect of Péclet number

In Figures 12.2 and 12.3, the three proposed versions of the primal global formulation are compared – with and without the body-force f – for $Pe_L = 1$, 24, and 640 (L is the

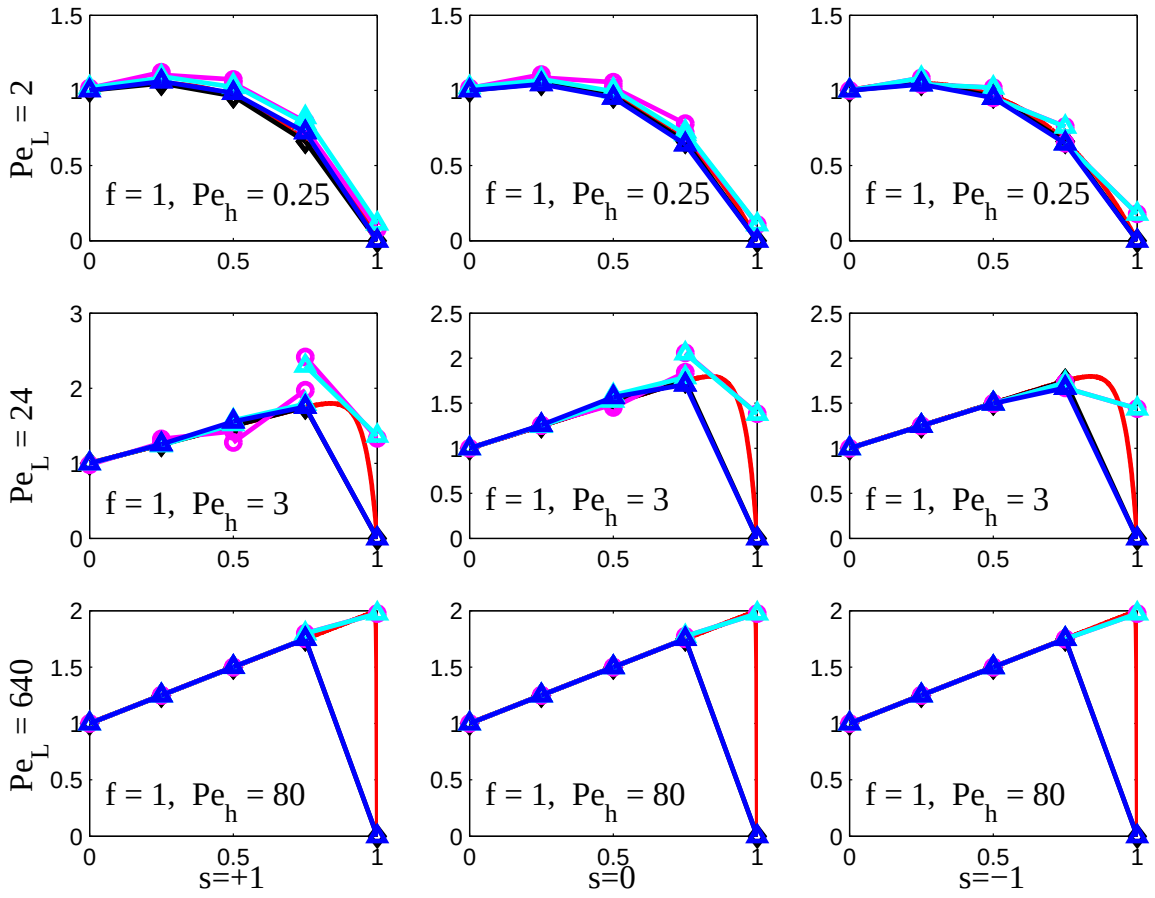

Figure 12.3: Solution plots in terms of varying Péclet number, on uniform grid of 4 elements, with body-force $f = 1$: Left, $s = +1$ (skew); Middle, $s = 0$ (neutral); Right, $s = -1$ (symmetric). Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, exact solution; black, Galerkin SUPG-stabilized solution with optimal τ (almost always overlapped with the $\overline{\phi}_h$ solution).

overall length of the computational domain).

As one can see, the results are practically perfect for the plots on the right column of each figure, corresponding to the symmetric formulation. Unwanted oscillations are present for the other formulations, particularly evident in the left column plots (skew form).

The following observations may be made:

1. The solution ϕ_h of the multiscale approach (cyan) is practically on top of the gDG solution (without local condensation, magenta), whenever the gDG solution is accurate, and tends to improve on it whenever the latter shows over/under-shoots. This beneficial effect is due to the fact that, in practice, the local problem

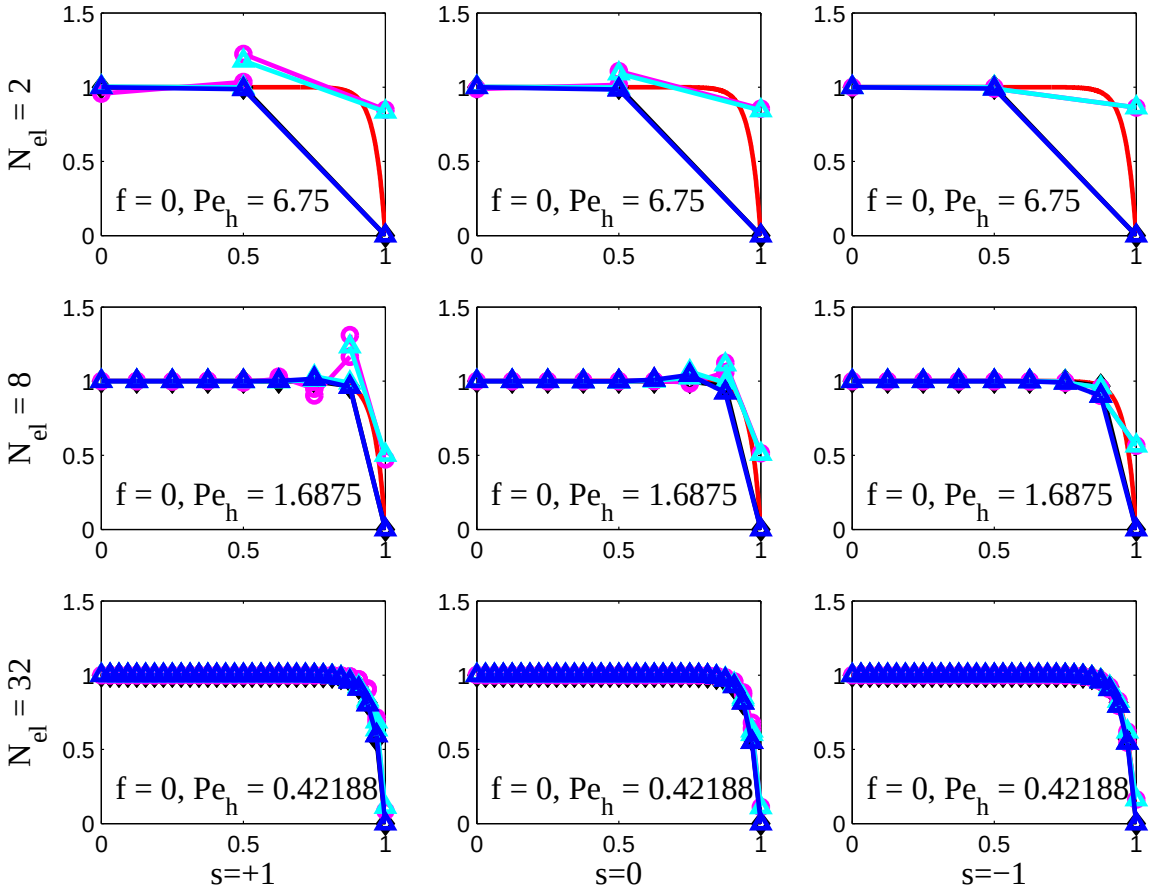

Figure 12.4: Solution plots in terms of varying mesh size, on uniform grid of 2/8/32 elements, $Pe_L = 27$, without body-force. Left, $s = +1$ (skew); Middle, $s = 0$ (neutral); Right, $s = -1$ (symmetric). Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, exact solution; black, Galerkin SUPG-stabilized solution with optimal τ (almost always overlapped with the $\overline{\phi}_h$ solution).

acts as a stabilization, with very important consequences on the convergence rates.

2. The choice of the parameter ϵ is crucial in order to have a good method. It was found that the choice $\epsilon = 2.001$ (slightly larger than the threshold 2) was very effective for the symmetric method, while in the case of non-symmetric formulations, the results did not show strong dependence on the choice of ϵ.

Effect of mesh refinement

It is now important to observe that accuracy is affected in particular in the range of medium-to-high Péclet numbers. The quality of the solution does not seem extremely

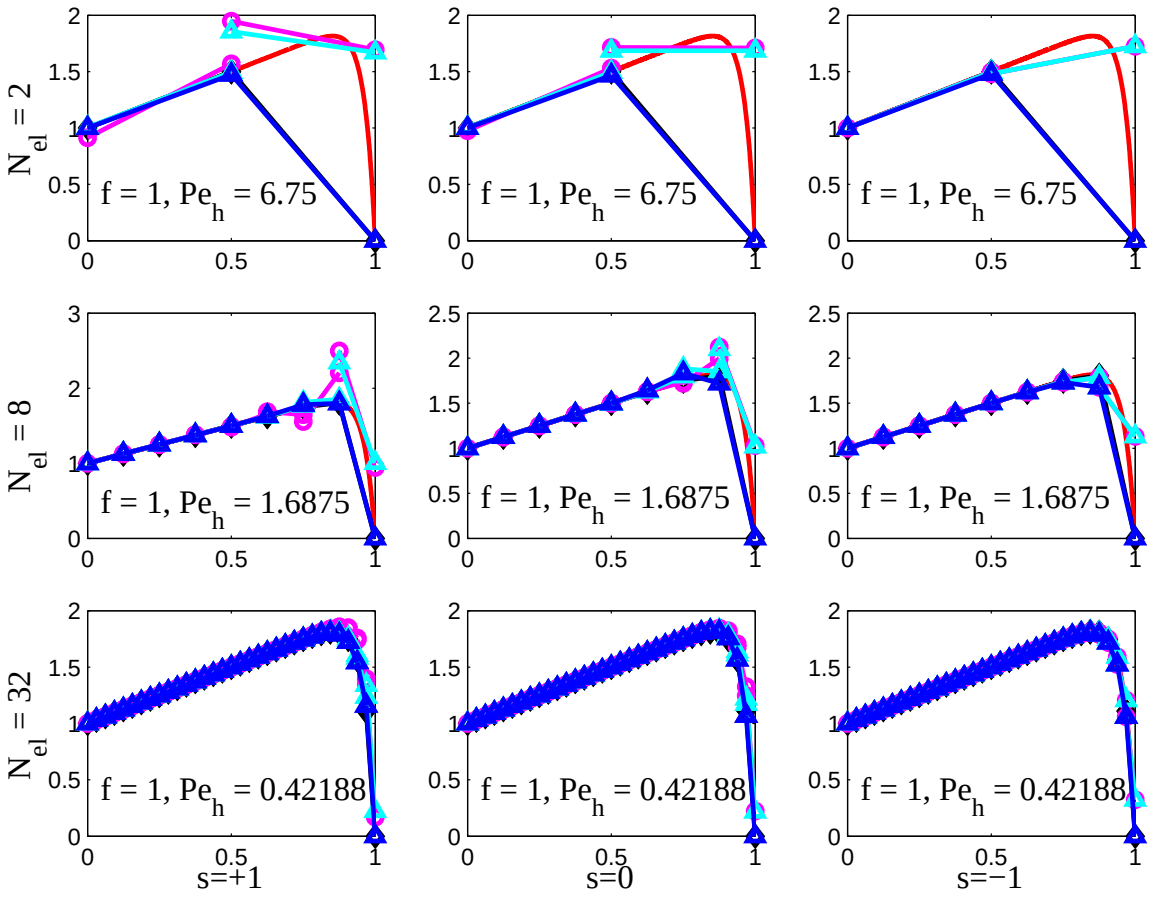

Figure 12.5: Solution plots in terms of varying mesh size, on uniform grid of 2/8/32 elements, $Pe_L = 27$, with body-force $f = 1$: Left, $s = +1$ (skew); Middle, $s = 0$ (neutral); Right, $s = -1$ (symmetric). Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, exact solution; black, Galerkin SUPG-stabilized solution with optimal τ (almost always overlapped with the $\overline{\phi}_h$ solution).

poor for the non-symmetric methods when Pe_L is very small.

Therefore, Figures 12.4 and 12.5 are presented to highlight the behavior of the solutions in the intermediate range ($Pe_L = 27$), with zero and non-zero body-force, respectively.

Notice that, as the mesh is refined, the quality of the solution is fairly good for all methods, but oscillations are still present for the non-symmetric versions. Also, notice the poorer accuracy for the global DG solutions for the non-symmetric versions, when compared to the corresponding locally stabilized versions.

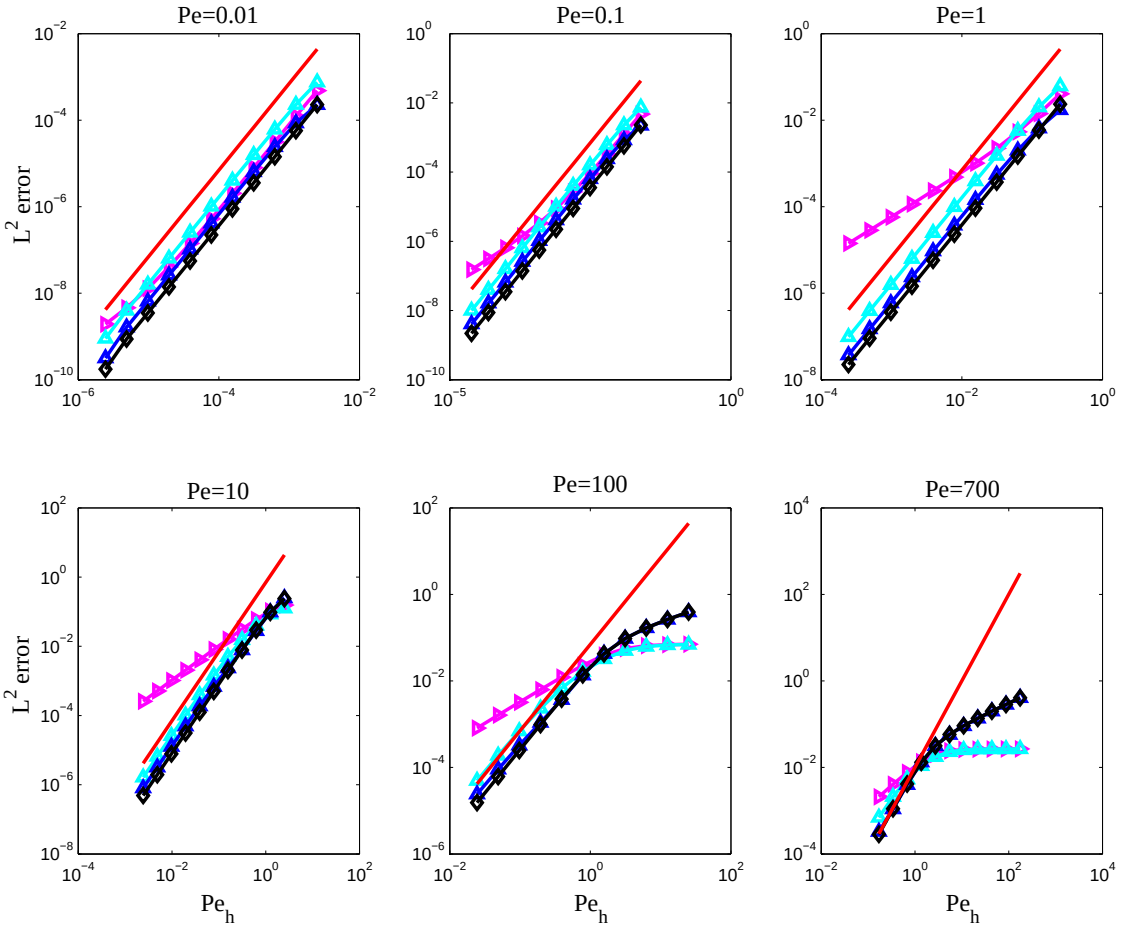

Figure 12.6: Convergence rates, skew-symmetric ($s = +1$) version, without body-force. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^2 slope; black, Galerkin SUPG-stabilized solution with optimal τ. Again, even if the global DG formulation alone does not show second-order convergence rate, the *locally condensed* version ($\overline{\phi}_h$) regains optimal convergence.

Convergence rates

The symmetric version of the method clearly shows second-order convergence rate (Fig. 12.8). In addition, it is worthwhile noticing that:

1. The discontinuous part of the solution is about as accurate as the solution of the gDG method (magenta curves).

2. The solutions of the proposed multiscale method are better than the SUPG solutions for all Péclet numbers Pe_h larger than (approximately) unity. Note that the SUPG solution with the optimal choice of the parameter τ is nodally exact in the

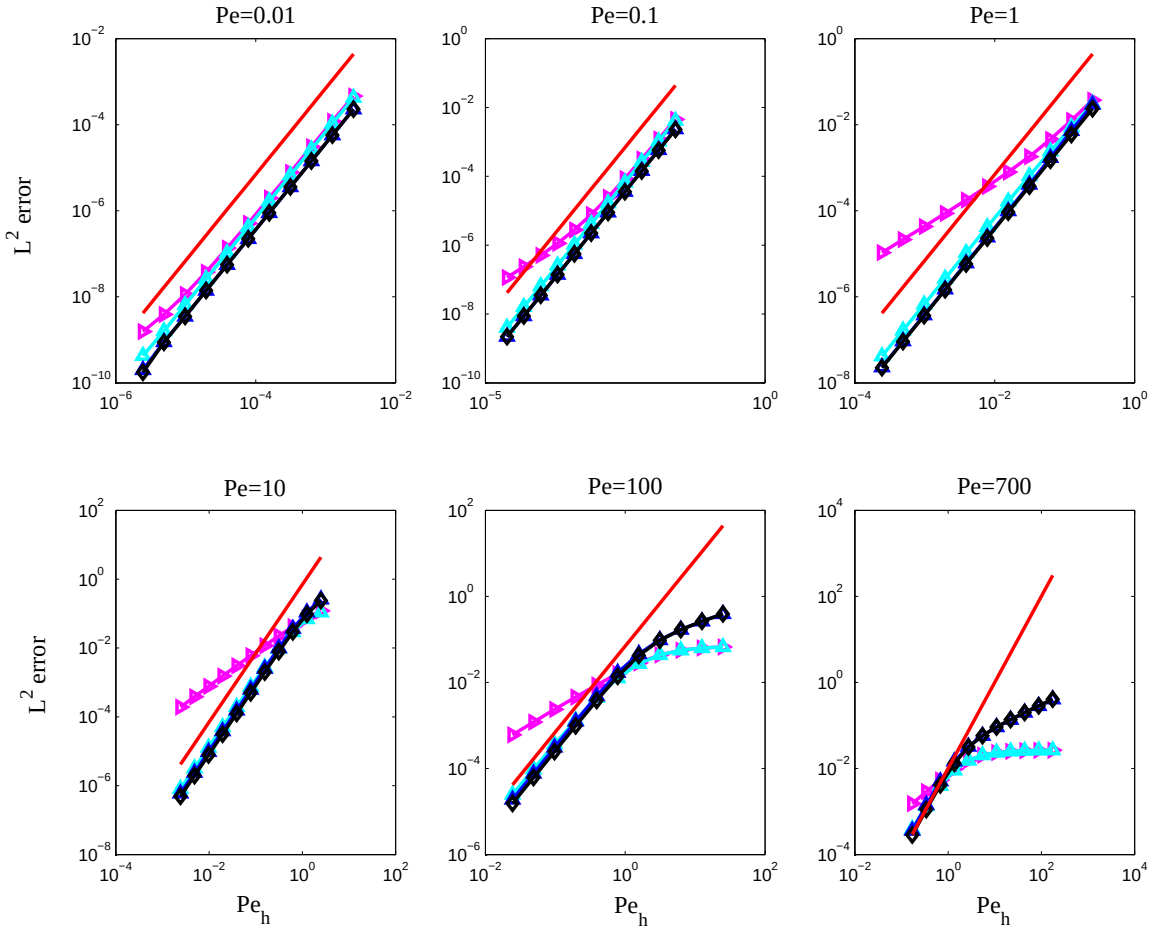

Figure 12.7: Convergence rates, neutral ($s = 0$) version, without body-force. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^2 slope; black, Galerkin SUPG-stabilized solution with optimal τ. Notice that even if the global DG formulation alone does not show second-order convergence rate, the *locally condensed* version ($\overline{\phi}_h$) regains optimal convergence.

one-dimensional case, and, as a consequence, all the error is due to interpolation. The multiscale method, instead, does have both numerical and interpolation error components, but the discontinuous approximation allows large gains in terms of the L^2-error inside the boundary layer region. The gain is more and more consistent as Pe_h increases, while for $Pe_h \ll 1$, the continuous discretization used in the SUPG approach yields the best results, since diffusion dominates.

3. One remarkable fact, clear from the result in Figures 12.6 and 12.7 is that local condensation provides a very beneficial effect on the convergence rate of the non-symmetric methods. Due to the lack of adjoint consistency, non-symmetric gDG

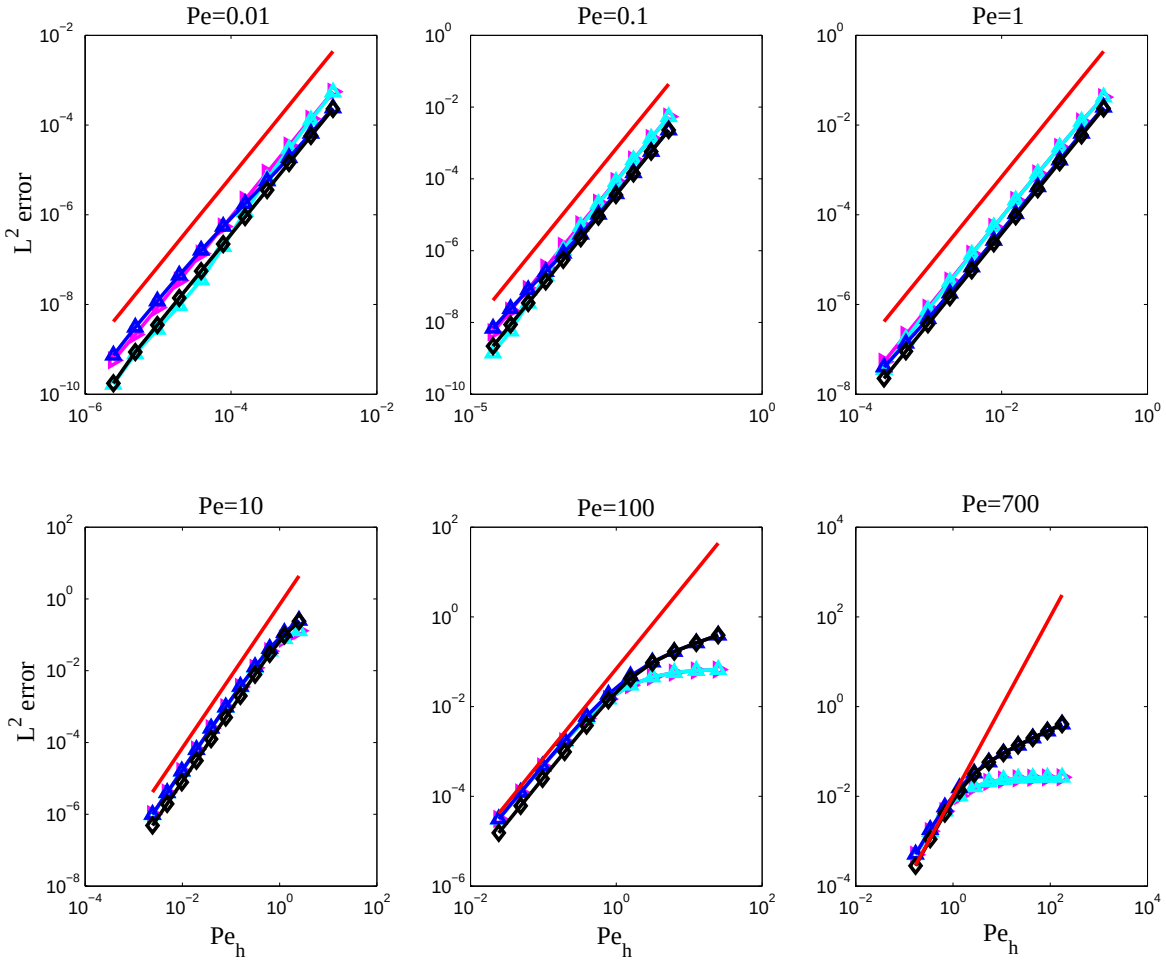

Figure 12.8: Convergence rates, symmetric ($s = -1$) version, without body-force. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^2 slope; black, Galerkin SUPG-stabilized solution with optimal τ.

methods deliver convergence rates lower than second-order, while the respective locally condensed solutions are second-order accurate. The non-symmetric formulations are not preferred due to the fact that they lack monotonicity in the boundary layer, a very important issue when tackling more challenging (e.g., nonlinear) problems.

4. The L^1-norm of the error is plotted in Figure 12.11, showing a behavior very similar to the L^2-norm plots.

5. The H^1 broken semi-norm of the error is presented in Figure 12.10. The definition

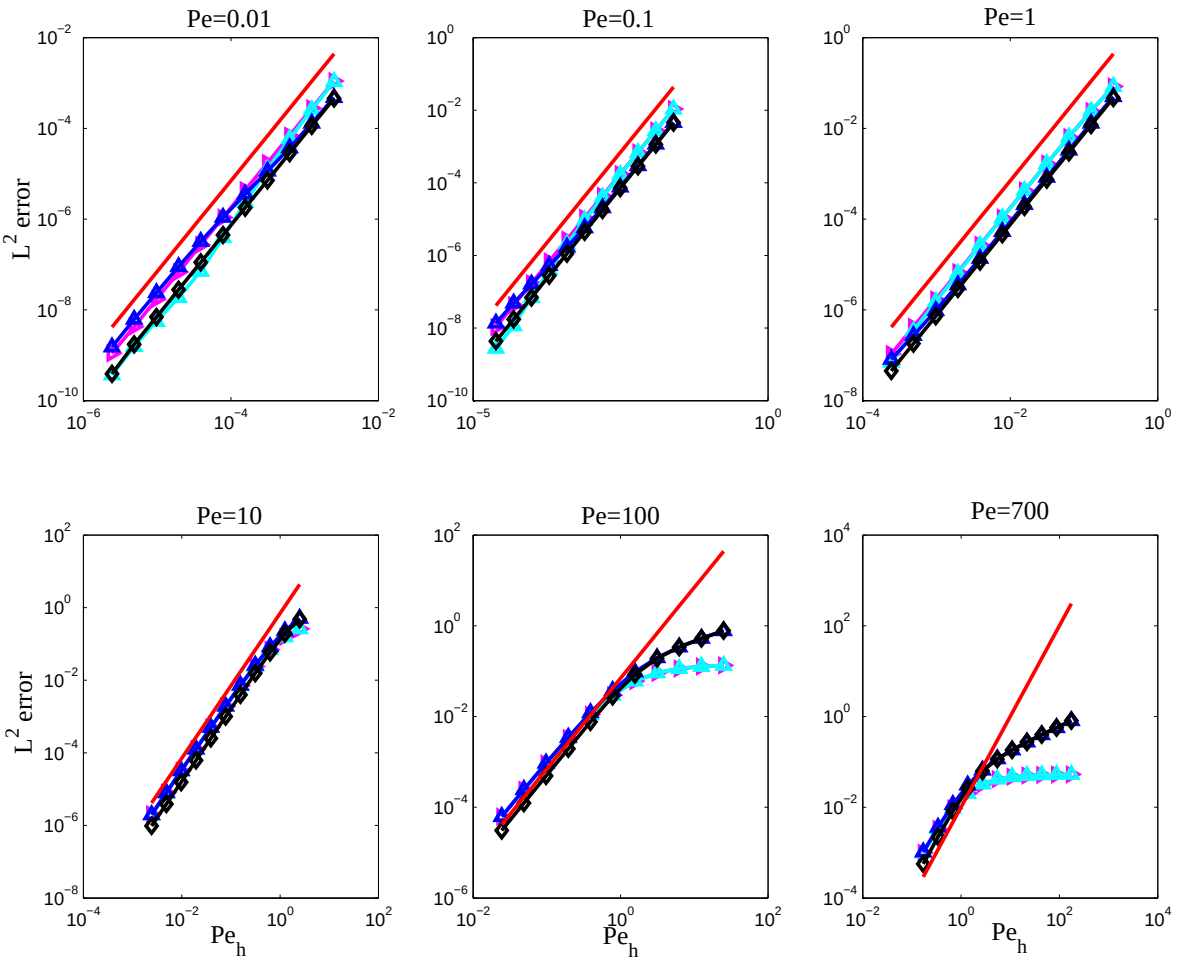

Figure 12.9: Convergence rates in the L^2-norm of the error, symmetric ($s = -1$) version, with body-force $f = 1$. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^2 slope; black, Galerkin SUPG-stabilized solution with optimal τ.

of broken semi-norm reads:

$$[\![\eta]\!]_1^2 \;=\; \sum_{T\in\mathcal{T}_h}\int_T |\nabla\eta|^2 = \sum_{T\in\mathcal{T}_h} |\eta|_{1,e}^2 \tag{12.23}$$

The continuous component of the solution $\overline{\phi}_h$ is very close in performance to the SUPG optimal solution. The discontinuous part is slightly poorer, but it still delivers optimal convergence rate.

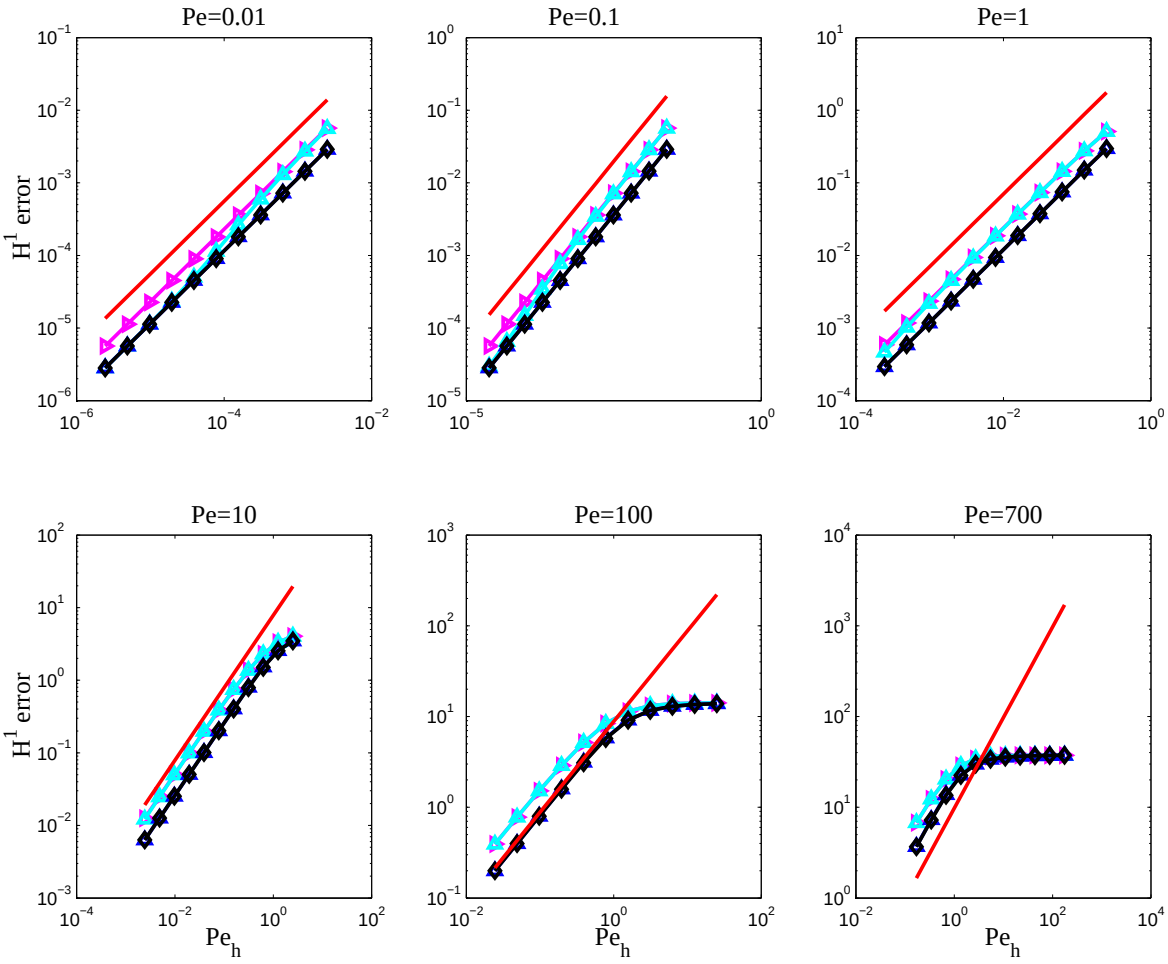

Figure 12.10: Convergence rates in the H^1 broken semi-norm of the error, symmetric ($s = -1$) version, with body-force $f = 1$. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^1 slope; black, Galerkin SUPG-stabilized solution with optimal τ.

12.2 Conclusions and Future Directions

A completely new approach, blending the advantages of the continuous and discontinuous Galerkin methods into a unique framework, has been proposed and preliminary numerical tests for the linear advection-diffusion problem showed a significant potential. The role of the local problem has been explored and its implications on the convergence rates and overall quality of the solution for the tests performed has proved of great help in understanding the nature of the method.

The possibility of computing discontinuous Galerkin approximations at the cost of the continuous Galerkin counterparts, if confirmed in more challenging problems, might have

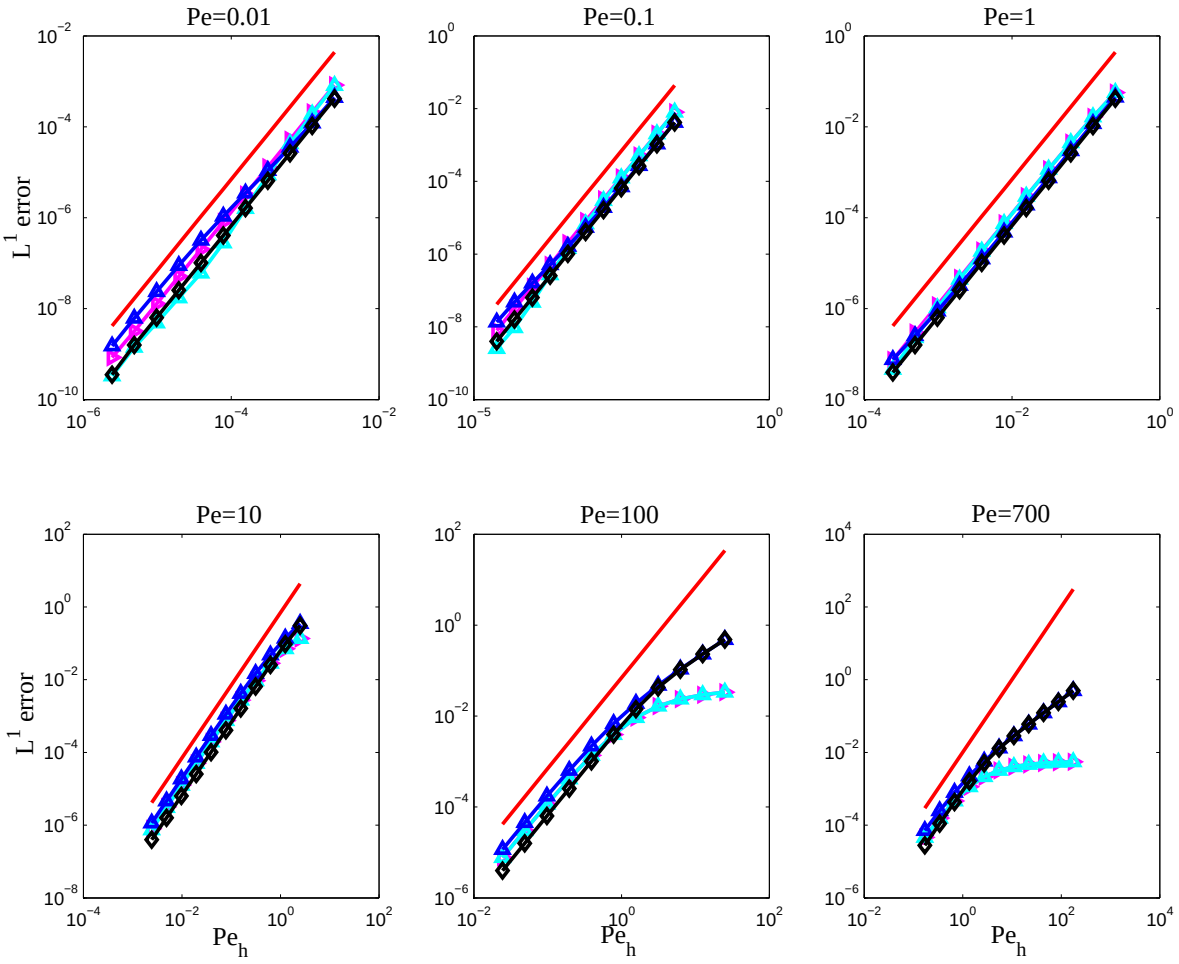

Figure 12.11: Convergence rates in the L^1-norm of the error, symmetric ($s = -1$) version, with body-force $f = 1$. Blue, $\overline{\phi}_h$; cyan (light blue), ϕ_h; magenta, global DG solution without local condensation; red, Pe_h^2 slope; black, Galerkin SUPG-stabilized solution with optimal τ.

a significant impact on the way DG methods will be designed in the future. It is natural therefore to look forward and outline as a future research pathway the generalization of the proposed method to more complex, multi-dimensional, nonlinear applications.

Little is known also in terms of how to analyze methods of this kind, and how to prove convergence in a more general setting. This constitutes another important avenue for future work.

As a note, the name *Magic Carpet* for the method has been chosen since the discontinuous correction of the local problem acts as a magic trick in allowing accurate and robust solutions, considerably less expensive than their full discontinuous Galerkin counterparts.

The discontinuous solutions to the local problems (i.e., the sum of the continuous parts and discontinuous corrections) *feel* at times convection (like *streamers* in the wind), at times diffusion, at times the forcing term: *They work like magic and look like flying carpets.*

Part III

Multiscale Methods in Turbulence

Chapter 13

A General Framework

The present chapter develops the general multiscale framework applied to the incompressible Navier-Stokes equations. A multiscale decomposition of the solution into coarse and fine scales is introduced. Combined with an asymptotic expansion, the decomposition generates a cascade of multiscale problems for the fine scale solution. The validity and applicability of the asymptotic expansion to problems of practical interest will be discussed.

13.1 Incompressible Navier-Stokes equations

The initial/boundary-value problem consists of solving the following equations for $\boldsymbol{u} : \overline{\mathsf{Q}} \to \mathbb{R}^d$, the velocity, and $p : \mathsf{Q} \to \mathbb{R}$, the pressure (divided by density),

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) + \boldsymbol{\nabla} p = \nu \Delta \boldsymbol{u} + \boldsymbol{f} \qquad \text{in } \mathsf{Q} \qquad (13.1)$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0 \qquad \text{in } \mathsf{Q} \qquad (13.2)$$

$$\boldsymbol{u} = \boldsymbol{0} \qquad \text{on } \mathsf{P} \qquad (13.3)$$

$$\boldsymbol{u}(0^+) = \boldsymbol{u}(0^-) \qquad \text{on } \Omega \qquad (13.4)$$

where $\boldsymbol{f} : \mathsf{Q} \to \mathbb{R}^d$ is the given body force (per unit volume); ν is the kinematic viscosity, assumed positive and constant; $\boldsymbol{u}(0^-) : \Omega \to \mathbb{R}^d$ is the given initial velocity; and $\otimes$ denotes the tensor product (e.g., in component notation $[\boldsymbol{u} \otimes \boldsymbol{v}]_{ij} = u_i v_j$).

Equations (13.1)-(13.4) are, respectively, the linear momentum balance, the incompressibility constraint, the no-slip boundary condition and the initial condition.

13.1.1 Space-time formulation

We consider a Galerkin space-time formulation with weakly imposed initial condition. Let $\mathcal{V} = \mathcal{V}(\mathsf{Q})$ denote the trial solution and weighting function spaces, which are assumed to be identical. We assume $\boldsymbol{U} = \{\boldsymbol{u}, p\} \in \mathcal{V}$ implies $\boldsymbol{u} = \boldsymbol{0}$ on P and $\int_\Omega p(t)\, d\Omega = 0$ for all $t \in]0, T[$. The variational formulation is stated as follows:

Find $\boldsymbol{U} \in \mathcal{V}$ *such that* $\forall \boldsymbol{W} = \{\boldsymbol{w}, q\} \in \mathcal{V}$*:*

$$B(\boldsymbol{W}, \boldsymbol{U}) = B_1(\boldsymbol{W}, \boldsymbol{U}) + B_2(\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{U}) = (\boldsymbol{W}, \boldsymbol{F}) \tag{13.5}$$

with

$$\begin{aligned} B_1(\boldsymbol{W}, \boldsymbol{U}) &= (\boldsymbol{w}(T^-), \boldsymbol{u}(T^-))_\Omega \; - \; \left(\frac{\partial \boldsymbol{w}}{\partial t}, \boldsymbol{u}\right)_{\mathsf{Q}} \\ &\quad + (q, \boldsymbol{\nabla} \cdot \boldsymbol{u})_{\mathsf{Q}} - (\boldsymbol{\nabla} \cdot \boldsymbol{w}, p)_{\mathsf{Q}} \; + \; (\boldsymbol{\nabla}^s \boldsymbol{w}, 2\nu \boldsymbol{\nabla}^s \boldsymbol{u})_{\mathsf{Q}} \qquad (13.6) \\ B_2(\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}) &= \; - \; (\boldsymbol{\nabla} \boldsymbol{w}, \boldsymbol{u} \otimes \boldsymbol{v})_{\mathsf{Q}} \qquad (13.7) \\ (\boldsymbol{W}, \boldsymbol{F}) &= (\boldsymbol{w}, \boldsymbol{f})_{\mathsf{Q}} + (\boldsymbol{w}(0^+), \boldsymbol{u}(0^-))_\Omega \qquad (13.8) \end{aligned}$$

where $\boldsymbol{V} = \{\boldsymbol{v}, r\}$. This formulation implies weak satisfaction of the momentum equations and incompressibility constraint, in addition to the initial condition. The boundary condition is built into the definition of $\mathcal{V}$.

Remarks

1. $\boldsymbol{u}(0^-)$ is viewed as known when computing the solution in Q.

2. The standard weak formulation corresponding to the discontinuous Galerkin method with respect to time is obtained by replacing $[0, T]$ by $[t_n, t_{n+1}]$, $n = 0, 1, 2, \ldots$ and summing over the space-time slabs

$$\mathsf{Q}_n = \Omega \times]t_n, t_{n+1}[\; . \tag{13.9}$$

In this case we view (13.5)-(13.8) as the variational equation for a typical slab.

3. The conditions $\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0$ on Q and $\boldsymbol{u} = \mathbf{0}$ on P imply

$$(\boldsymbol{\nabla} \boldsymbol{u}, \boldsymbol{u} \otimes \boldsymbol{u})_{\mathsf{Q}} = 0 \ . \tag{13.10}$$

In the discrete case this term may need to be altered to preserve this property. See Quarteroni and Valli [58], p.435.

13.2 Separation of scales

Let us decompose $\mathcal{V}$ as follows:

$$\mathcal{V} = \overline{\mathcal{V}} \oplus \mathcal{V}' \tag{13.11}$$

where $\overline{\mathcal{V}}$ is identified with a standard *finite dimensional space*. Various characterizations of $\mathcal{V}'$ are possible. Note that $\mathcal{V}'$ is ∞-dimensional. The over-bar and prime notations are used to connote large and small scales. All resolved scales in $\overline{\mathcal{V}}$ are viewed as coarse scales, obviating the issue of inefficiency *ab initio*. Furthermore, the decomposition into coarse and fine scales is considered exact. For example, in the spectral case, the coarse-scale space consists of all Fourier modes beneath some cut-off wave number (i.e. the highest Fourier coefficient computed in the simulation) and the fine-scale space consists of all remaining Fourier modes. Other interpretations are possible: in hierarchical finite element methods, $\mathcal{V}'$ can be represented by the levels of refinement (h or p-refinement), in stabilized methods $\mathcal{V}'$ can be the space of bubbles, etc.

The derivation of the coarse- and fine-scale equations proceeds, first, by substituting the decomposed solution into the Navier-Stokes equations, then, by projecting this equation into the coarse- and fine-scale subspaces. The projection into coarse scales yields a finite dimensional system for the coarse-scale component of the solution, which depends parametrically on the fine-scale component.

Therefore, (13.11) enables us to decompose (13.5) into two sub-problems:

$$\begin{aligned} B(\overline{\boldsymbol{W}}, \overline{\boldsymbol{U}} + \boldsymbol{U}') &= B_1(\overline{\boldsymbol{W}}, \overline{\boldsymbol{U}} + \boldsymbol{U}') + B_2(\overline{\boldsymbol{W}}, \overline{\boldsymbol{U}} + \boldsymbol{U}', \overline{\boldsymbol{U}} + \boldsymbol{U}') \\ &= (\overline{\boldsymbol{W}}, \boldsymbol{F}) \qquad (13.12) \\ B(\boldsymbol{W}', \overline{\boldsymbol{U}} + \boldsymbol{U}') &= B_1(\boldsymbol{W}', \overline{\boldsymbol{U}} + \boldsymbol{U}') + B_2(\boldsymbol{W}', \overline{\boldsymbol{U}} + \boldsymbol{U}', \overline{\boldsymbol{U}} + \boldsymbol{U}') \\ &= (\boldsymbol{W}', \boldsymbol{F}) \qquad (13.13) \end{aligned}$$

in which

$$\begin{aligned} \boldsymbol{U} &= \overline{\boldsymbol{U}} + \boldsymbol{U}' \\ \boldsymbol{W} &= \overline{\boldsymbol{W}} + \boldsymbol{W}' \end{aligned} \qquad (13.14)$$

$\overline{\boldsymbol{U}}, \overline{\boldsymbol{W}} \in \overline{\mathcal{V}}$ and $\boldsymbol{U}', \boldsymbol{W}' \in \mathcal{V}'$. Consider equation (13.13) and expand the terms using linearity of the bilinear form $B_1(\cdot, \cdot)$ and trilinear form $B_2(\cdot, \cdot, \cdot)$:

$$B_{\overline{U}}(\boldsymbol{W}', \boldsymbol{U}') + B_2(\boldsymbol{W}', \boldsymbol{U}', \boldsymbol{U}') = (\boldsymbol{W}', \boldsymbol{Res}(\overline{\boldsymbol{U}})) \qquad (13.15)$$

where

$$\begin{aligned} B_{\overline{U}}(\boldsymbol{W}', \boldsymbol{U}') &= B_1(\boldsymbol{W}', \boldsymbol{U}') + B_2(\boldsymbol{W}', \boldsymbol{U}', \overline{\boldsymbol{U}}) + B_2(\boldsymbol{W}', \overline{\boldsymbol{U}}, \boldsymbol{U}') \qquad (13.16) \\ (\boldsymbol{W}', \boldsymbol{Res}(\overline{\boldsymbol{U}})) &= (\boldsymbol{W}', \boldsymbol{F}) - B_1(\boldsymbol{W}', \overline{\boldsymbol{U}}) - B_2(\boldsymbol{W}', \overline{\boldsymbol{U}}, \overline{\boldsymbol{U}}) \qquad (13.17) \end{aligned}$$

The term $B_2(\overline{\boldsymbol{W}}, \boldsymbol{U}', \boldsymbol{U}')$, according to classical LES notation, will be referred as the *Reynolds stress* term, and the sum $B_2(\overline{\boldsymbol{W}}, \boldsymbol{U}', \overline{\boldsymbol{U}}) + B_2(\overline{\boldsymbol{W}}, \overline{\boldsymbol{U}}, \boldsymbol{U}')$ will be referred as the *cross stress* term. In the spectral case, the coupling of coarse and large scales is due to the cross and Reynolds stress terms exclusively. In the case of non-orthogonal bases, even the linear terms give rise to coupling between coarse and fine scales. The coarse-scale component plays an analogous role to the filtered field in the classical approach, but has the advantage of avoiding all problems associated with homogeneity, commutativity, walls, compressibility, etc.

It will also be assumed that the cut-off wave number is sufficiently large that the philosophy of LES is appropriate. For example, if there is a well-defined inertial sub-range, then we assume the cut-off wave number resides somewhere within it. This assumption enables us to further assume that the energy content in the fine scales is small compared

with the coarse scales. This turns out to be *crucial* in our efforts to analytically represent the solution of the fine-scale equations.

The strategy is to obtain approximate analytical expressions for the fine scales then substitute them into the coarse-scale equations which are, in turn, solved numerically. If the scale decomposition is performed in space and time, the *only* approximation in the procedure is the representation of the fine-scale solution.

Remark

$\overline{\boldsymbol{U}}$ and $\boldsymbol{U}'$ may be considered as "projections" of $\boldsymbol{U}$ onto $\overline{\mathcal{V}}$ and $\mathcal{V}'$, respectively. The terminology "projections" is used loosely, because $\overline{\boldsymbol{U}}$ and $\boldsymbol{U}'$ are obtained from $\boldsymbol{U}$ by solving coupled nonlinear problems, viz.,

$$B(\overline{\boldsymbol{W}}, \overline{\boldsymbol{U}} + \boldsymbol{U}') = B(\overline{\boldsymbol{W}}, \boldsymbol{U}) \tag{13.18}$$

$$B(\boldsymbol{W}', \overline{\boldsymbol{U}} + \boldsymbol{U}') = B(\boldsymbol{W}', \boldsymbol{U}) \ . \tag{13.19}$$

$\overline{\boldsymbol{U}}$ represents the part of $\boldsymbol{U}$ which lives in $\overline{\mathcal{V}}$, and thus clearly is a large-scale representation of $\boldsymbol{U}$. Likewise, $\boldsymbol{U}'$ is a small-scale representation of $\boldsymbol{U}$.

13.3 An asymptotic expansion for the small scales

To provide a framework for the fine-scale approximation, we assume an infinite perturbation series expansion to treat the fine-scale nonlinear term in the fine-scale equation.

By virtue of the smallness of the fine scales, this expansion is expected to converge rapidly in many cases of practical interest. The remaining part of the fine-scale Navier-Stokes system is the *linearized* operator which is formally inverted through the use of a Green's function. The combination of a perturbation series and Green's function provides an exact formal solution of the fine-scale Navier-Stokes equations. The driving force in these equations is the Navier-Stokes system residual computed from the coarse scales.

This expresses the intuitively obvious fact that if the coarse scales constitute a good approximation to the solution of the problem, the coarse-scale residual will be small and the resulting fine-scale solution will be small as well, providing a rational basis for

assuming the perturbation series converges rapidly. Note that one cannot use such an argument on the original problem because in this case the perturbation series would almost definitely fail to converge (in fact this argument could be used, one would solve the Navier-Stokes equations analytically).

Let us assume then $\boldsymbol{U}'$ is small in some sense. From the multiscale analysis, it is well-known that $\boldsymbol{U}'$ is related to the residual of the large scale equation, namely $\boldsymbol{Res}(\overline{\boldsymbol{U}})$. Using this fact, if the residual of the large scale equations is assumed small, so is $\boldsymbol{U}'$, and it is possible to apply the following asymptotic expansion:

$$\boldsymbol{U}' = \varepsilon \boldsymbol{U}'_1 + \varepsilon^2 \boldsymbol{U}'_2 + \varepsilon^3 \boldsymbol{U}'_3 + \ldots = \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n \tag{13.20}$$

where we enforce $\varepsilon = \left\| \boldsymbol{Res}(\overline{\boldsymbol{U}}) \right\|$. Technically, $\|\cdot\|$ should be the dual norm, since this is the norm that makes sense for the residual. However, by means of *a posteriori* error-estimation techniques, it is possible to relate the mentioned dual norm to the energy norm, much easier to implement for practical purposes. In fact, by way of the *lifting* operator, (13.17), $\boldsymbol{Res}(\overline{\boldsymbol{U}}) \in (\mathcal{V}')^* \subset L^2$. Thus, in the following derivations, $\|\cdot\|$ will indicate an energy-type norm.

Let us therefore rewrite (13.15) in terms of the proposed expansion:

$$B_{\overline{U}} \left(\boldsymbol{W}', \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n \right) + B_2 \left(\boldsymbol{W}', \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n, \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n \right) = \varepsilon \left(\boldsymbol{W}', \widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}}) \right), \tag{13.21}$$

where

$$\widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}}) = \frac{\boldsymbol{Res}(\overline{\boldsymbol{U}})}{\left\| \boldsymbol{Res}(\overline{\boldsymbol{U}}) \right\|} \tag{13.22}$$

Notice that, by linearity, $B_{\overline{U}}(\cdot, \cdot)$ can be written as:

$$B_{\overline{U}} \left(\boldsymbol{W}', \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n \right) = \sum_{n=1}^{\infty} \varepsilon^n B_{\overline{U}} \left(\boldsymbol{W}', \boldsymbol{U}'_n \right), \tag{13.23}$$

while the second term requires further consideration.

We expand it as follows:

$$\begin{aligned} B_2(\boldsymbol{W}' \quad , \quad & \varepsilon\boldsymbol{U}'_1 + \varepsilon^2\boldsymbol{U}'_2 + \varepsilon^3\boldsymbol{U}'_3 + \dots \ , \quad \varepsilon\boldsymbol{U}'_1 + \varepsilon^2\boldsymbol{U}'_2 + \varepsilon^3\boldsymbol{U}'_3 + \dots \ ) \\ = \quad & \varepsilon^2 B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_1) \\ & + \varepsilon^3 \left[B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_2) + B_2(\boldsymbol{W}', \boldsymbol{U}'_2, \boldsymbol{U}'_1) \right] \\ & + \varepsilon^4 \left[B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_3) + B_2(\boldsymbol{W}', \boldsymbol{U}'_2, \boldsymbol{U}'_2) + B_2(\boldsymbol{W}', \boldsymbol{U}'_3, \boldsymbol{U}'_1) \right] \\ & + \dots \end{aligned} \tag{13.24}$$

A recurrence formula can be easily computed if we refer to the following scheme:

$$\begin{array}{lllll} \varepsilon^2 & \rightarrow & B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_1) & & \\ \varepsilon^3 & \rightarrow & B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_2) & B_2(\boldsymbol{W}', \boldsymbol{U}'_2, \boldsymbol{U}'_1) & \\ \varepsilon^4 & \rightarrow & B_2(\boldsymbol{W}', \boldsymbol{U}'_1, \boldsymbol{U}'_3) & B_2(\boldsymbol{W}', \boldsymbol{U}'_2, \boldsymbol{U}'_2) & B_2(\boldsymbol{W}', \boldsymbol{U}'_3, \boldsymbol{U}'_1) \\ \varepsilon^5 & \rightarrow & \dots & \dots & \dots \qquad \dots \end{array}$$

Hence:

$$B_2\left(\boldsymbol{W}', \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n, \sum_{n=1}^{\infty} \varepsilon^n \boldsymbol{U}'_n\right) = \sum_{n=2}^{\infty} \varepsilon^n \sum_{j=1}^{n-1} B_2\left(\boldsymbol{W}', \boldsymbol{U}'_j, \boldsymbol{U}'_{n-j}\right) \tag{13.25}$$

The full expansion of the equation can be compactly written as follows:

$$\sum_{n=1}^{\infty} \varepsilon^n B_{\overline{\boldsymbol{U}}}(\boldsymbol{W}', \boldsymbol{U}'_n) + \sum_{n=2}^{\infty} \varepsilon^n \sum_{j=1}^{n-1} B_2\left(\boldsymbol{W}', \boldsymbol{U}'_j, \boldsymbol{U}'_{n-j}\right) = \varepsilon\left(\boldsymbol{W}', \widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}})\right) \tag{13.26}$$

Equating like coefficients, we obtain a sequence of linear equations coupled through their right hand sides (RHS):

$$\text{For } n = 1 \qquad B_{\overline{\boldsymbol{U}}}(\boldsymbol{W}', \boldsymbol{U}'_1) = \left(\boldsymbol{W}', \widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}})\right) \tag{13.27}$$

$$\text{For } n \geq 2 \qquad B_{\overline{\boldsymbol{U}}}(\boldsymbol{W}', \boldsymbol{U}'_n) = -\sum_{j=1}^{n-1} B_2\left(\boldsymbol{W}', \boldsymbol{U}'_j, \boldsymbol{U}'_{n-j}\right) \tag{13.28}$$

The bilinear operator $B_{\overline{\boldsymbol{U}}}(\cdot, \cdot)$ is the same for all the equations in the asymptotic cascade, and can be inverted through Green's function techniques, or, since the analytic solution

is usually very hard to find, using an approximation of the Green's function. The Green's function concept can be introduced in an abstract sense through a *resolvent* operator:

$$M'_{\overline{U}}(\cdot): \quad \mathcal{V}'^{*} \rightarrow \mathcal{V}' \tag{13.29}$$

$$\mathcal{F}(\cdot) \mapsto \boldsymbol{V}' \tag{13.30}$$

such that

$$B_{\overline{U}}(\boldsymbol{W}', \boldsymbol{V}') = \mathcal{F}(\boldsymbol{W}') \tag{13.31}$$

If a sequence of operators $\mathcal{F}_j : \mathcal{V}' \rightarrow \mathbb{R}$ (i.e., $\mathcal{F}_j \in \mathcal{V}'^{*}$) is defined as:

$$\begin{aligned} \text{For } n = 1 \quad \mathcal{F}_1(\boldsymbol{W}') &= \mathcal{F}_1(\boldsymbol{W}'; \widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}})) \\ &= \left(\boldsymbol{W}', \widehat{\boldsymbol{R}}(\overline{\boldsymbol{U}})\right) \end{aligned} \tag{13.32}$$

$$\begin{aligned} \text{For } n \geq 2 \quad \mathcal{F}_n(\boldsymbol{W}') &= \mathcal{F}_n(\boldsymbol{W}'; \boldsymbol{U}'_1, \ldots, \boldsymbol{U}'_{n-1}) \\ &= -\sum_{j=1}^{n-1} B_2\left(\boldsymbol{W}', \boldsymbol{U}'_j, \boldsymbol{U}'_{n-j}\right) \end{aligned} \tag{13.33}$$

then it is possible to reformulate the sequence of problems (13.27)-(13.28) as:

$$\boldsymbol{U}'_n = M'_{\overline{U}}(\mathcal{F}_n), \qquad n = 1, 2, \ldots \tag{13.34}$$

Notice that in the cascade of problems (13.27)–(13.28) (or (13.32)–(13.33)) the level n correction in the asymptotic expansion depends on terms on the RHS which involve the residual of the coarse-scale component and levels of the asymptotic expansions from 1 to $n-1$. Therefore the asymptotic expansion provides a solution method to the nonlinear problem by means of a cascade of linear systems.

Finally, once $\boldsymbol{U}'$ is computed, it can be substituted into (13.12), in order to account for the effect of the small scales onto to the large ones. The substitution *closes* the equations; (13.12) can then be solved only in terms of the $\overline{\boldsymbol{U}}$ component.

The formal solution of the fine-scale equations suggests various approximations may be employed in practical problem solving. One would be tempted to use the word "modeling" because approximate analytical representations of the fine scales constitute the

only approximation and hence may be thought of as the "modeling" component of the present approach. It is important however to emphasize that it is very different from classical modeling ideas which are dominated by the *addition* of *ad hoc* eddy viscosities.

Numerical results will demonstrate eddy viscosities are unnecessary in some circumstances. There are two aspects to the approximation of the fine scales:

1. Approximation of the Green's function for the linearized Navier-Stokes system.

2. Approximation of the nonlinearities represented by the perturbation series.

The first and obvious thought for the latter aspect, nonlinearity, is to simply truncate the perturbation series. This idea is investigated, in conjunction with some simple approximations of the Green's function. It turns out there is considerable experience in local scaling approximations of the Green's function based on the theory of *stabilized methods* [32, 34, 37]. The Green's function is typically approximated by locally defined algebraic operators (i.e., the "τ's" of stabilized methods) multiplied by local values of the coarse-scale residual. With this approximation of the solution of the linearized operator, nonlinearities can be easily accounted for in perturbation series fashion.

Chapter 14

Numerical Experiments with the Burgers' Equation

A number of preliminary experiments with the Burgers equation will be presented in this chapter, devoted to highlight the nonlinear dynamics of the subgrid-scale component of the solution, with special emphasis on the analysis of the asymptotic expansion. The following discussion will form the background for a new modeling approach for large eddy simulation of the Navier-Stokes presented in chapter 15.

14.1 Strong form of the Burgers' equation

The initial/boundary-value problem (in non-dimensional form) consists of seeking $u : \mathsf{Q} \rightarrow \mathbb{R}$, the velocity, (where $\mathsf{Q} = \Omega \times [0, T]$, with $\Omega = \mathbb{T}_{[0,l]}$, the l-periodic 1-torus), such that

$$u_{,t} + \left(\frac{u^2}{2}\right)_{,x} = \frac{1}{Re} u_{,xx} + f \qquad \text{on } \mathsf{Q} \tag{14.1}$$

$$u(0^+) = u(0^-) \qquad \text{on } \Omega \tag{14.2}$$

Re is the Reynolds number (its precise definition is given in Appendix B). The length scale used to form Re is *not* the physical length $\tilde{l}$ of the computational domain (*tilde* denotes dimensional quantities), but is instead $\tilde{L} = 1/\tilde{k}_f$, the inverse of the wavenumber at which the Fourier spectrum of the stochastic forcing peaks. Hence, the non-dimensional

parameter l in the definition of the torus $\mathbb{T}_{[0,l]}$ is in general different from unity, and in the specific case of the following numerical tests, $l = 6$, that is, the forcing spectrum peaks at a wavelength equal to 1/6 of the domain length. $u(0^-) : \Omega \to \mathbb{R}$ is the given initial velocity, and the forcing $f : \mathsf{Q} \to \mathbb{R}$ is taken to be a Gaussian, white-in-time field with variance given by

$$< f(x,t)f(x',s) >= F(|x-x'|)\delta(t-s) \tag{14.3}$$

where $< \cdot >$, stands for an ensemble average, which can be replaced by an average in time due to stationarity of the stochastic process.

14.1.1 Space-time formulation

We consider a Galerkin space-time formulation with weakly imposed initial condition. Let $\mathcal{V} = \mathcal{V}(\mathsf{Q})$ denote the trial solution and weighting function spaces, which are assumed to be identical. The variational formulation is stated as follows:

Find $u \in \mathcal{V}$ such that $\forall w \in \mathcal{V}$

$$B(w,u) = B_1(w,u) + B_2(w,u,u) = (w,f) \tag{14.4}$$

where, for $v \in \mathcal{V}$

$$\begin{aligned} B_1(w,u) &= (w(T^-),u(T^-))_\Omega \; - \; (w_{,t},u)_\mathsf{Q} + \frac{1}{Re}\,(w_{,x},u_{,x})_\mathsf{Q} && (14.5)\\ B_2(w,u,v) &= \; - \; \Big(w_{,x},\frac{uv}{2}\Big)_\mathsf{Q} && (14.6)\\ (w,f) &= (w,f)_\mathsf{Q} + (w(0^+),u(0^-))_\Omega && (14.7) \end{aligned}$$

14.2 Separation of scales and asymptotics

As for the case of the Navier-Stokes equations in chapter 13, let $\mathcal{V} = \overline{\mathcal{V}} \oplus \mathcal{V}'$, so that:

$$\begin{aligned} B(\overline{w},\overline{u}+u') &= B_1(\overline{w},\overline{u}+u') + B_2(\overline{w},\overline{u}+u',\overline{u}+u') = (\overline{w},f) && (14.8)\\ B(w',\overline{u}+u') &= B_1(w',\overline{u}+u') + B_2(w',\overline{u}+u',\overline{u}+u') = (w',f) && (14.9) \end{aligned}$$

in which the following scale decomposition is adopted:

$$u = \overline{u} + u' \tag{14.10}$$

$$w = \overline{w} + w' \tag{14.11}$$

$\overline{u}, \overline{w} \in \overline{\mathcal{V}}$ and $u', w' \in \mathcal{V}'$. Consider equation (14.9) and expand the terms using linearity of the bilinear form $B_1(\cdot,\cdot)$ and trilinear form $B_2(\cdot,\cdot,\cdot)$:

$$B_{\overline{u}}(w', u') + B_2(w', u', u') = (w', Res(\overline{u})) \tag{14.12}$$

where

$$B_{\overline{u}}(w', u') = B_1(w', u') + B_2(w', u', \overline{u}) + B_2(w', \overline{u}, u') \tag{14.13}$$

$$(w', Res(\overline{u})) = (w', f) - B_1(w', \overline{u}) - B_2(w', \overline{u}, \overline{u}) \tag{14.14}$$

Assuming u' is small and expanding:

$$u' = \varepsilon u'_1 + \varepsilon^2 u'_2 + \varepsilon^3 u'_3 + \ldots = \sum_{n=1}^{\infty} \varepsilon^n u'_n \tag{14.15}$$

where $\varepsilon = \|Res(\overline{u})\|$. Carrying out the necessary derivations yields:

$$\sum_{n=1}^{\infty} \varepsilon^n B_{\overline{u}}\left(w', u'_n\right) + \sum_{n=2}^{\infty} \varepsilon^n \sum_{j=1}^{n-1} B_2\left(w', u'_j, u'_{n-j}\right) = \varepsilon\left(w', \widehat{R}(\overline{u})\right) \tag{14.16}$$

with $\widehat{R}(\overline{u}) = Res(\overline{u}) / \|Res(\overline{u})\|$. Equating like coefficients, we obtain a cascade of linear equation coupled through their right hand sides:

$$\text{For } n = 1 \qquad B_{\overline{u}}\left(w', u'_1\right) = \left(w', \widehat{R}(\overline{u})\right) \tag{14.17}$$

$$\text{For } n \geq 2 \qquad B_{\overline{u}}\left(w', u'_n\right) = -\sum_{j=1}^{n-1} B_2\left(w', u'_j, u'_{n-j}\right) \tag{14.18}$$

The bilinear operator $B_{\overline{u}}(\cdot,\cdot)$ can be inverted through a Green's function (or at least its discrete version), and the solution can be reconstructed from its asymptotic expansion.

Once u' is computed, it can be substituted into (14.8), in order to account for the effect of the small scales onto to the large ones. As in chapter 13, it is possible to abstract the Green's function concept by the *resolvent* operator:

$$M'_{\overline{u}}(\cdot): \quad \mathcal{V}'^{*} \rightarrow \mathcal{V}' \tag{14.19}$$

$$\mathcal{F}(\cdot) \mapsto v' \tag{14.20}$$

such that

$$B_{\overline{u}}(w', v') = \mathcal{F}(w') \tag{14.21}$$

As before, the sequence of operators $\mathcal{F}_j : \mathcal{V}' \rightarrow \mathbb{R}$ (i.e., $\mathcal{F}_j \in \mathcal{V}'^{*}$) is introduced:

$$\begin{aligned} \text{For } n = 1 \quad \mathcal{F}_1(w') &= \mathcal{F}_1(w'; \widehat{R}(\overline{u})) \\ &= \left(w', \widehat{R}(\overline{u})\right) \end{aligned} \tag{14.22}$$

$$\begin{aligned} \text{For } n \geq 2 \quad \mathcal{F}_n(w') &= \mathcal{F}_n(w'; u'_1, \ldots, u'_{n-1}) \\ &= -\sum_{j=1}^{n-1} B_2\left(w', u'_j, u'_{n-j}\right) \end{aligned} \tag{14.23}$$

Therefore, (14.17)–(14.18) reduce to the following cascade of equations is obtained:

$$u'_n = M'_{\overline{u}}(\mathcal{F}_n), \qquad n = 1, 2, \ldots \tag{14.24}$$

14.3 Small-scale dynamics

The simplicity of the Burgers equation allows the solution of (14.8) and (14.17)–(14.18), to study the nonlinear subgrid-scale dynamics of the asymptotic expansion. In practical implementations, the cascade of equations (14.17)–(14.18) is solved for only approximately, but in this particular case, the brute force solution can provide some insight on the energy redistribution patterns among the terms in the asymptotic expansion.

To solve (14.17)–(14.18), in essence, we just need to compute a certain number of times the solution to a system of the form:

$$B_{\overline{u}}(w', u'_n) = (w', \mathcal{F}_n) \tag{14.25}$$

in which $\mathcal{F}_n$ represent the right hand side term in equations (14.17)–(14.18), corresponding to the component u'_n. In order to simplify the notation in what follows, the subscript n will be dropped from $\mathcal{F}_n$ and u'_n.

We proceed introducing a basis for $\mathcal{V}'$. If we consider that $B_{\overline{u}} : \mathcal{V}' \times \mathcal{V}' \to \mathbb{R}$, we can easily see that the correct expressions for both the test function w' and the solution u' in a classical Galerkin formulation are given by:

$$w'(x,t) = \sum_{A=1}^{n'_{\rm np}} N'_A w'_A \tag{14.26}$$

$$u'(x,t) = \sum_{A=1}^{n'_{\rm np}} N'_A u'_A \tag{14.27}$$

where $n'_{\rm np}$ (possibly infinite) is the number of degrees-of-freedom characterizing the space $\mathcal{V}'$ of the solution, and N'_A is the shape function associated with the degree-of-freedom u'_A. The discretization adopted leads to a linear matrix problem:

$$\sum_{B=1}^{n'_{\rm np}} B_{\overline{u}}(N'_A, N'_B)\ u'_B = (N'_A, \mathcal{F}),\ A = 1, 2,\ \ldots\ , n'_{\rm np} \tag{14.28}$$

or

$$\sum_{B=1}^{n'_{\rm np}} [\boldsymbol{K}_{\ \overline{u}}]_{AB}\ \boldsymbol{U}'_B = \mathcal{F}_A, \qquad A = 1, 2,\ \ldots\ , n'_{\rm np} \tag{14.29}$$

where we have used the linearity in the second argument of $B_{\overline{u}}(\cdot, \cdot)$, and

$$[\boldsymbol{K}_{\ \overline{u}}]_{AB} = B_{\overline{u}}(N'_A, N'_B) \quad \in \mathbb{R}^{n'_{\rm np} \times n'_{\rm np}}. \tag{14.30}$$

Therefore we have:

$$u'_B = \sum_{A=1}^{n'_{\rm np}} [\boldsymbol{K}^{-1}_{\overline{u}}]_{BA} \mathcal{F}_A, \qquad B = 1, 2, \ \ldots \ , n'_{\rm np} \tag{14.31}$$

$$\begin{aligned} u'(x,t) &= \sum_{B=1}^{n'_{\rm np}} N'_B(x,t) \boldsymbol{U}'_B \\ &= \sum_{A,B=1}^{n'_{\rm np}} N'_B(x,t) (\boldsymbol{K}^{-1}_{\overline{u}})_{BA} \mathcal{F}_A \\ &= \int_{\tilde{\Omega}} \sum_{A,B=1}^{n'_{\rm np}} N'_B(x,t) (\boldsymbol{K}^{-1}_{\overline{u}})_{BA} N'_A(\tilde{x},\tilde{t}) \mathcal{F}(\tilde{x},\tilde{t}) d\tilde{\Omega} \\ &= \int_{\tilde{\Omega}} G_{\overline{u}}(x,t;\tilde{x},\tilde{t}) \mathcal{F}(\tilde{x},\tilde{t}) d\tilde{\Omega} \ . \end{aligned} \tag{14.32}$$

where

$$G_{\overline{u}}(x,t;\tilde{x},\tilde{t}) = \sum_{A,B=1}^{n'_{\rm np}} N'_B(x,t) (\boldsymbol{K}^{-1}_{\overline{u}})_{BA} N'_A(\tilde{x},\tilde{t}) \tag{14.33}$$

is the Green's function for the linear problem, and in particular $G_{\overline{u}} \in \mathbb{R}$. Appendix B.3 contains the details of the implementation of the numerical Green's function concept, which can also be understood in terms of a generalized Shur's complement formula. At the numerical level, $n'_{\rm np}$ is taken large enough to incorporate all significant dynamics of the Burgers' equation with the set of coarse and fine scales.

14.4 Numerical testing.

The numerical implementation of the Burgers' equation simulation is documented in Appendix B. The algorithm is a multiscale variant of a classical Bubnov-Galerkin formulation with piecewise continuous interpolation on a uniform mesh. The large-scale equation is solved at each time-step using a (nonlinear) Newton iterative solver, and a triadiagonal solver is adopted for the cascade of linearized fine-scale problems.

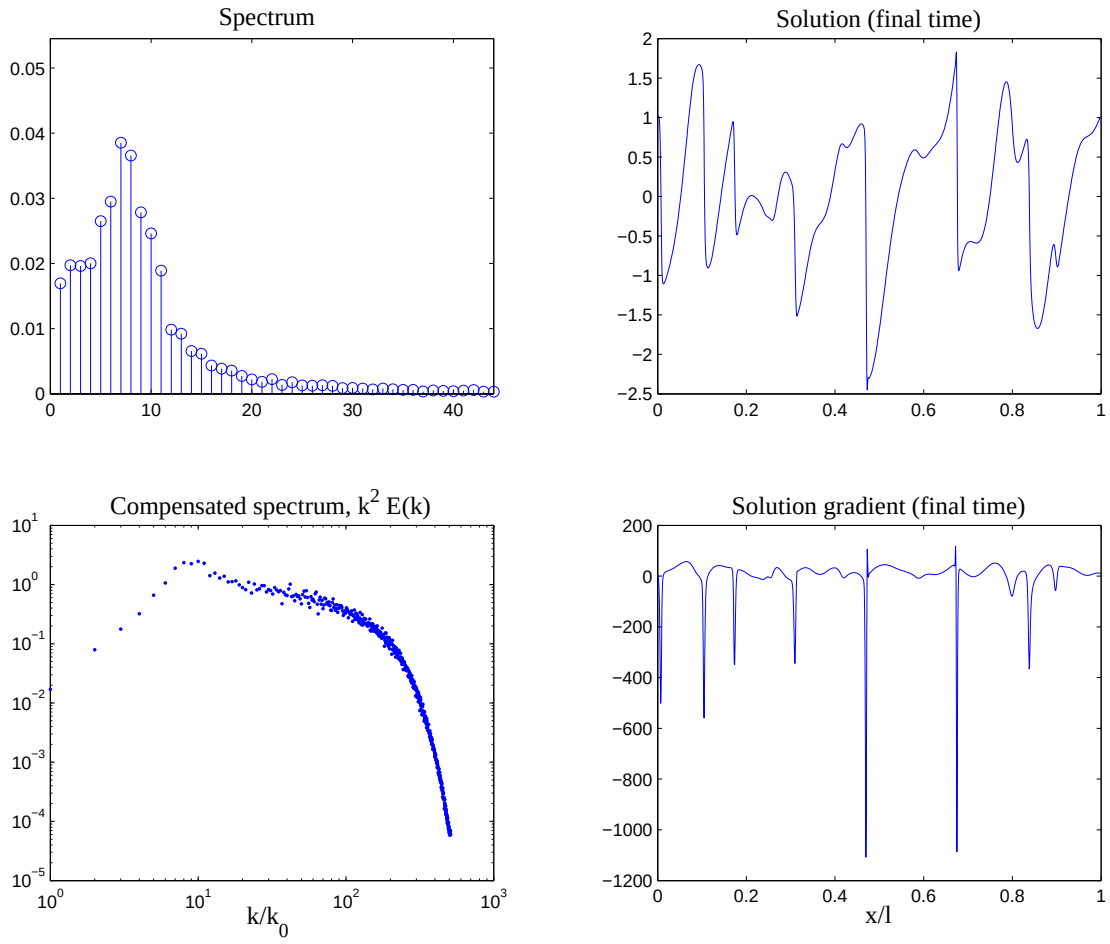

Figure 14.1: $Re = 1000$, 1024 elements, $\Delta t = 0.001$. Statistics for Pope's forcing (see Appendix B.5.2 for details). The shocks are producing very pronounced negative cusps in the gradient of the solution (bottom right plot). The negative gradient of the solution scales with the Reynolds number: the magnitude of the negative cusps in the solution gradient, observable in the bottom right plot, is approximately equal to the value of the Reynolds number.

The dynamics of the Burgers' equation can shed some light on how energy redistributes among the various terms of the asymptotic expansion and the resulting general considerations can be applied to the much more complex Navier-Stokes equations. The nonlinearities in Burgers' equation are mainly due to shocks, that cannot be represented accurately on coarse meshes . At the same time, a shock can be thought of as a boundary layer of extreme sharpness.

In this regard, the representation of sharp gradients with coarse meshes becomes an issue that involves the mathematical and numerical aspects of function representation. The proposed analysis tends to direct the attention on such numerical representation

problems, that are common to more complex systems (e.g., the Navier-Stokes equations). Some of the commonalities will be outlined in the next chapter on bypass transition simulations.

14.4.1 Preliminary calculations.

Preliminary studies have been devoted to outline the basic chaotic dynamics of the randomly forced Burgers' equation. Figure 14.1 shows the result of a simulation at $Re = 1000$, with 1024 elements, and $\Delta t = .001$ using S.B. Pope's implementation of the stochastic forcing (see Appendix B.5 and [56] for more details).

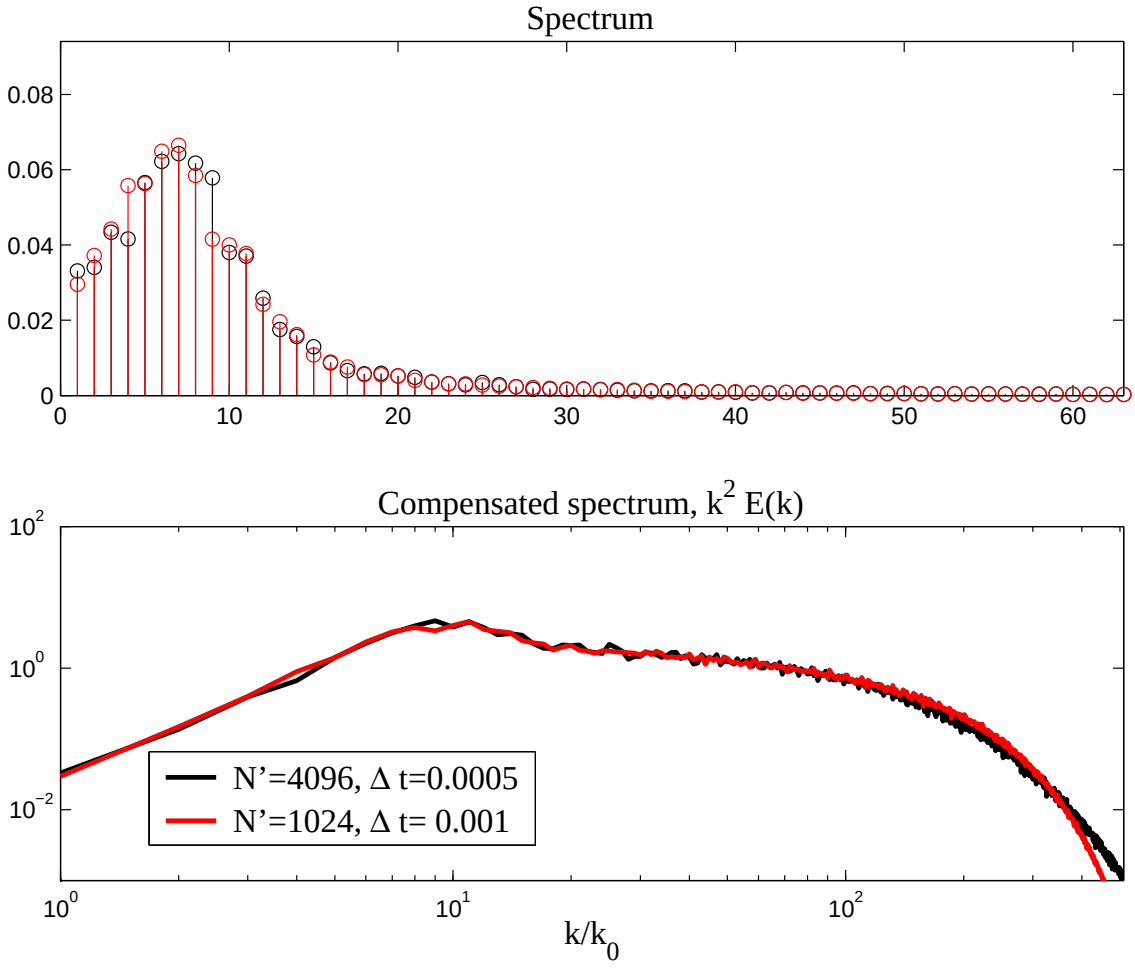

Figure 14.2: Comparison of spectra for Re=1000. Black: 4096 elements, $\Delta t = 0.0005$; red: 1024 elements, $\Delta t = 0.001$. It can be appreciated that the range of low/medium wavenumbers ($k_j = 2\pi j/L$) is not influenced by the increase in resolution. On the contrary, high wavenumbers are sensitive to a change in resolution.

The solution has evolved for 50 non-dimensional time units from a randomly generated initial condition and sampling has been performed over the subsequent $T_s = 50$ time units. As it can be clearly seen, pronounced negative cusps in the gradient of the solution are forming due to shock occurrences. The plots in Figure 14.1 are presented just for illustration purposes, since, as will be clear from subsequent analysis, the resolution in this case is not adequate to characterize the dynamics of highest wavenumbers in the Fourier spectrum of the solution.

In fact, Figure 14.2 shows a mesh sensitivity analysis at Re=1000, for which it is noticeable that the representation of the high-wavenumber range for the energy spectrum of the solution is inaccurate (recall that the definition of the wavenumber k_j is $k_j = 2\pi j/L$). Reducing the accuracy is not beneficial for the correct representation of high wavenumbers, while low wavenumbers (corresponding to the coarsest scales in the simulation) are practically unaffected. Since the small-scale dynamics needs to be characterized properly, the analysis confirms that the finer mesh should be used.

Re	number of elements	Δt	T_s (Sampling Duration)
100	1024	0.001	100
1000	4096	0.0005	100
2000	4096	0.0005	100

Table 14.1: Effect of the Reynolds number. All the simulations have been performed without the asymptotic expansion, retaining only the large scales, and using a resolution providing adequate accuracy with respect to the corresponding Reynolds number.

14.4.2 Effect of the Reynolds number.

Table 14.1 summarizes the tests performed, with adequate meshes and increments of the time integration. 4096 elements have been used for the higher Reynolds numbers, to accurately compute the spectrum for high wavenumbers.

In this case, no asymptotic expansion has been applied, and all the scales in the simulation are coarse scales, according to the notation previously introduced, although the resolution of the mesh ensures that there is virtually no energy in the modes past the cutoff of the Nyquist wavenumber $k_{cutoff} = \frac{\pi}{h}$ (the highest wavenumber *sampled* by the mesh of element spacing h).

.

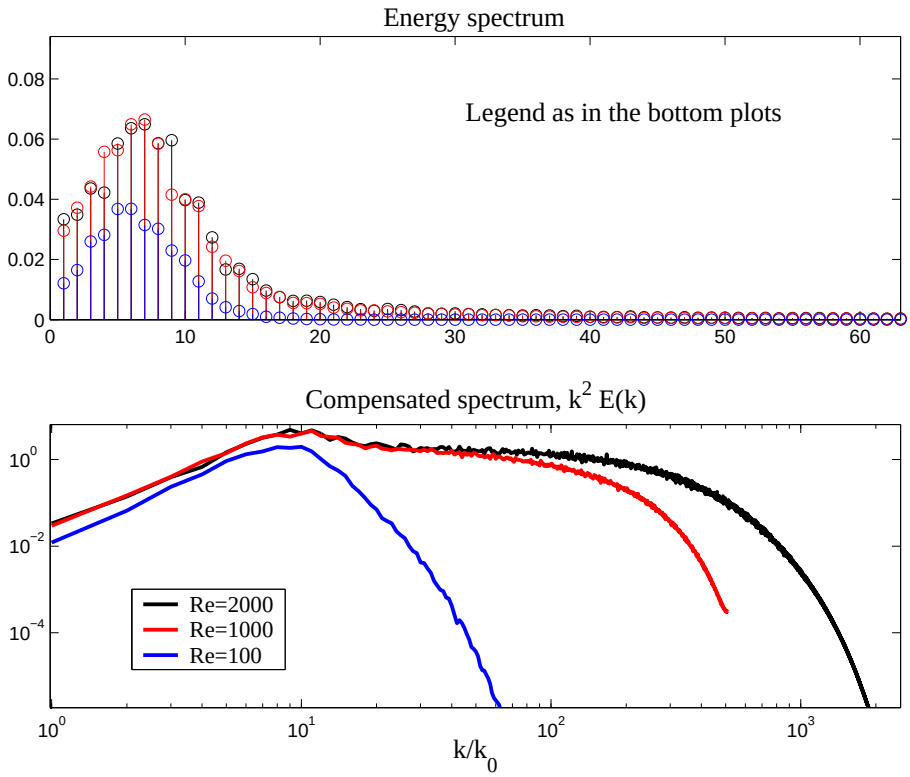

Figure 14.3: Comparison of spectra for the three simulations outlined in Table 14.1. Notice the formation of a *plateau* as the Reynolds number increases.

A comparison of spectra is presented in Figure 14.3, for a random forcing applied as suggested in Appendix B. As $Re \to \infty$, a *plateau* forms in the compensated spectrum, corresponding to a k^{-2} decay in the energy spectrum. Such decay is originated by the formation of shocks in the solution, whose Fourier coefficients decay as k^{-1}.

The plateau in the spectrum is fairly pronounced in the case of $Re = 2000$, given the very steep gradients produced (see Fig. 14.4). Notice that in the case $Re = 100$, the k^{-2} range is not present, since viscosity does not allow the shocks to be sharp enough.

14.4.3 Multiscale dynamics.

For the case with the highest Reynolds number of Table 14.1, corresponding to $Re = 2000$, the multiscale approach of section 14.3 was subsequently applied. Section 14.3 is sufficient for the comprehension of the basic approach, but the interested reader can find all the details of the implementation in section B.3 of the Appendix.

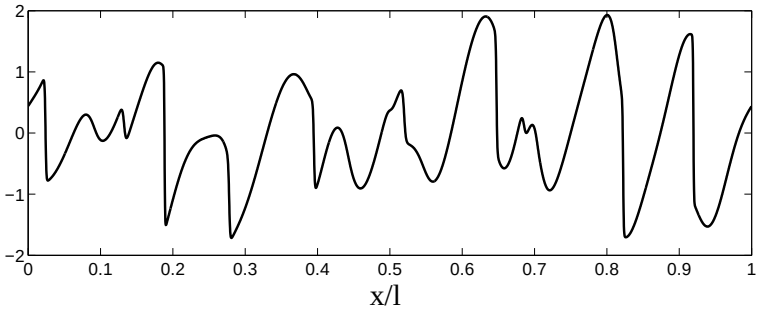

Figure 14.4: Solution of the stochastically forced Burgers equation for $Re = 2000$, corresponding to the parameters on the third row of table 14.1.

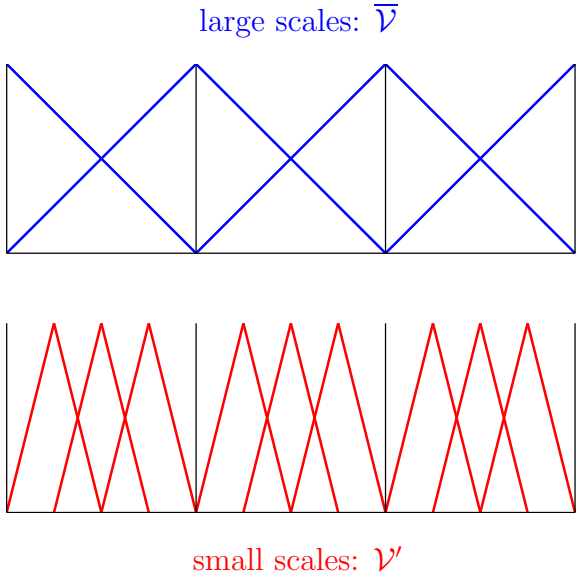

Figure 14.5: A sketch of the refinement strategy: coarse scales are in blue and small scales in red.

In this case the total number of 4096 degrees-of-freedom has been split into 512 degrees of freedom associated to a coarser mesh of spacing $L/512$ (L is the measure of the computational domain) and $n'_{\rm np} = 4096 - 512$ degrees-of-freedom, associated with a nested refinement. A sketch of the refinement strategy is depicted in Figure 14.5. The time duration for the statistical sample was $T_s = 400$ non-dimensional time units (8×10^5 time steps), to achieve convergence for the spectra of the small scale components. As in the previous simulation at $Re = 2000$, the time-step was fixed as $\Delta t = 5 \times 10^{-4}$. In particular, Figure 14.6 shows the results for respectively $n_{ae} = 1, 2, 3, 7$ terms in the asymptotic expansion.

The top plot shows the decay in the square-root of the spectrum, $E_{\overline{u}}^{1/2}$, for the coarse

scale solution $\overline{u}$. The plot immediately below shows the square-root of the spectrum, $E^{1/2}_{u'_1 u'_1}$, of the first component of the asymptotic expansions and so forth for the higher terms in the asymptotic expansion, plotted in descending order.

Several curves are plotted in color on each of the three lower plots of Figure 14.6. Each color corresponds to terms of the expansion truncated at the particular term n_{ae}. For example, red corresponds to terms of the asymptotic expansion truncated at $n_{ae} = 7$, green to a truncation at $n_{ae} = 3$ and so forth. Consequently, the very bottom plot of Figure 14.6 shows the term u'_3 in the asymptotic expansion for a truncation at the third ($n_{ae} = 3$, green) and seventh ($n_{ae} = 7$, red) term (curves for a truncation at the first and second term are obviously not present). A few observations can be made:

1. The order of magnitude of the peak of $E^{1/2}_{u'_1 u'_1}$ is about 1% of the peak of $E^{1/2}_{\overline{u}\,\overline{u}}$, and for successive corrections, the magnitude of the peaks for the $E^{1/2}_{u'_j u'_j}$'s (with $j \geq 2$) is about 0.1% of the one of $E^{1/2}_{\overline{u}\,\overline{u}}$. Therefore, the hypothesis of smallness of u' with respect to $\overline{u}$ is confirmed. Furthermore, it appears clear that if the separation of scales between coarse and fine scales in not extreme, the first term of the the expansion is sufficient to capture the bulk effects of the *subgrid-scale* space, in terms of the spectrum of u and its integral in wave-space, the variance.

2. Notice that when taking Fourier transforms, a "signature" of the finite element discretization appears in the plots for $E^{1/2}_{u'_1 u'_1}$ and $E^{1/2}_{u'_2 u'_2}$ (second and third from the top), appearing as bubble-like bumps in the energy distribution for the high wavenumbers. However, one should realize that the energy content of the mentioned bumps is fairly small.

3. The graph of $E^{1/2}_{u'_1 u'_1}$ is almost unaffected by changes in the truncation threshold n_{ae}, as the curves for $n_{ae} = 1, 2, 3, 7$ overlap. This fact is of crucial importance, since it tells that the *linear* correction u'_1 is very well characterized by just the expansion truncated at the first term. Notice also that the first term, once the numerical Green's function is appropriately approximated, corresponds to an enhanced SUPG-type approach in which the small-scale component of the solution is substituted in the full cross-stress sum as well as in the Reynolds-stresses (in the case of classical stabilized methods, the subgrid component is substituted into only

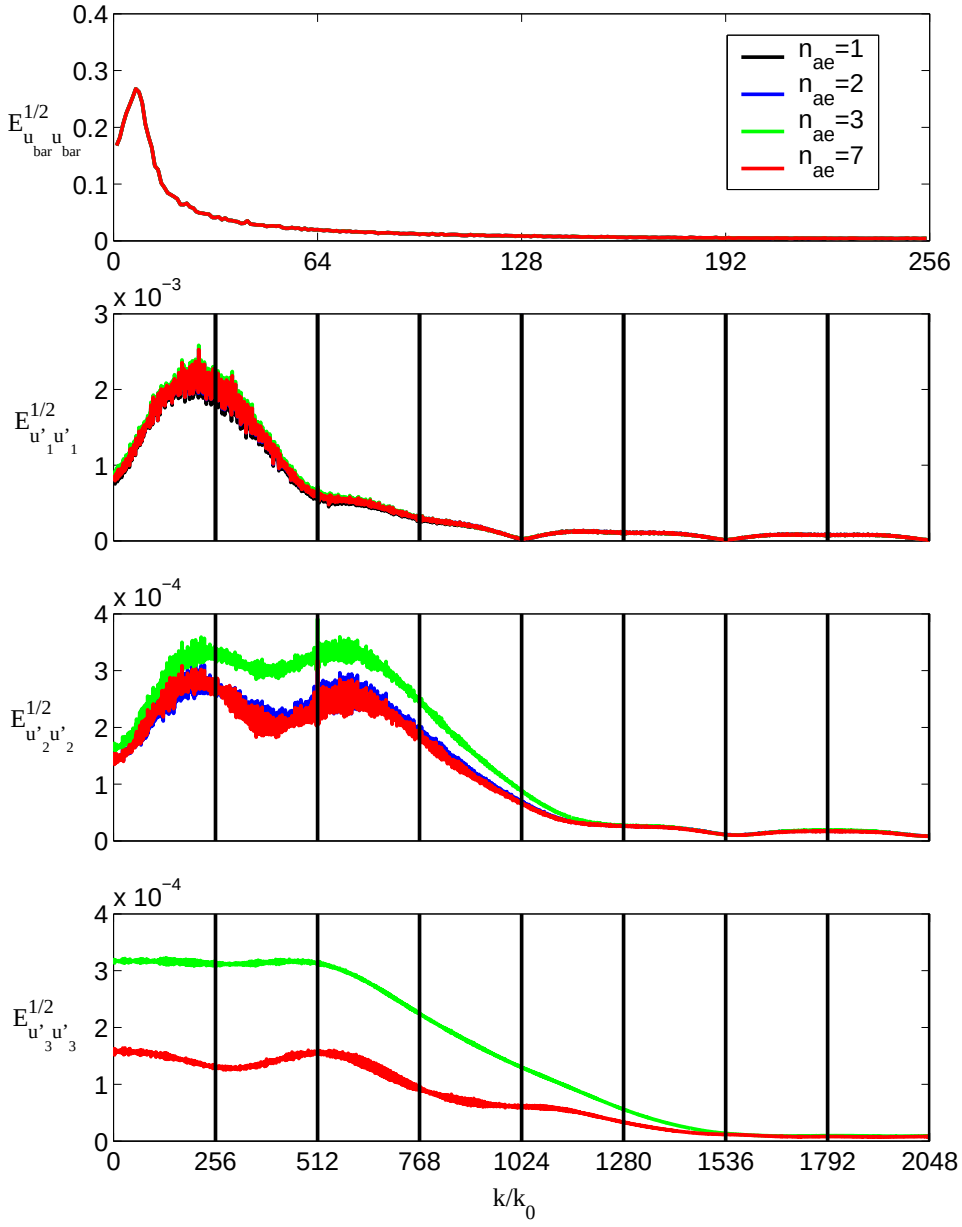

Figure 14.6: Square root of the energy spectrum. Top plot: $\overline{u}$. Second plot from top: u'_1 term of the asymptotic expansion for the small scale component of the solution. Third plot from top: u'_2. Fourth plot from top: u'_3. The range of values of normalized wavenumbers on the abscissa is from 1 to 256 for the top plot, while the lower plots range from 1 to 2048 (the black vertical separators represent intervals of 256 wave numbers). Also, for the plots of the terms in the asymptotic expansion, black refers to an expansion truncated at the first term ($n_{ae} = 1$), blue refers to a truncation at the second term ($n_{ae} = 2$), green to a truncation at the third ($n_{ae} = 3$), and red is for a truncation at the seventh term ($n_{ae} = 7$).

one of the two terms in the cross-stress sum, either $B_2(w', u', \overline{u})$ or $B_2(w', \overline{u}, u'))$.

4. For higher terms u'_j ($j \geq 2$), energy *does* redistribute as the truncation in the asymptotic expansion is changed. However, it is clear from the picture that a fairly small amount of energy is contained in these terms.

5. Note also that for higher terms in the asymptotic expansion, the spectrum is spread across a broader band of wavenumbers. Most of the energy in the case of u'_1 is concentrated in a band four times larger than the coarse scale band.

The conclusion from this analysis is that in the limit of applicability of the asymptotic expansion, most of the subgrid energy resides only in the first term of the expansion, so that new approaches to modeling can leverage this fact.

Chapter 15

Applications to Bypass Transition Phenomena and Conclusions

A new variational model for turbulence simulations will be proposed in what follows, and numerical results for bypass transition in boundary layers will be presented. The model hinges upon a generalization of the discussion outlined in the previous chapter regarding Burgers' equation experiments.

15.1 Bridging the gap with turbulence

It is important to recapitulate the results at the end of the previous chapter. If the mesh is not too coarse, then the subgrid scale component of the solution is fairly small compared with the coarse scale solution and the asymptotic expansion can be applied. Furthermore, it was observed that only the first term of the expansion, the *linear contribution*, was significantly larger than all *nonlinear contributions* coming from higher terms in the asymptotic expansion. This fact is not just a feature of the Burgers' equation, but also of more complex systems. For example, let us consider Figure 15.1, relative to a multiscale analysis applied to the Navier-Stokes equations in the case of homogeneous isotropic turbulence. The computations are courtesy of Alan Wray [69], and were performed on a fairly resolved mesh in spectral (Fourier) space, for a Reynolds number $Re_\lambda = 90$. For decaying homogeneous turbulence, the Reynolds number Re_λ is computed with the Taylor microscale (the autocorrelation characteristic length-scale of the turbulent signal),

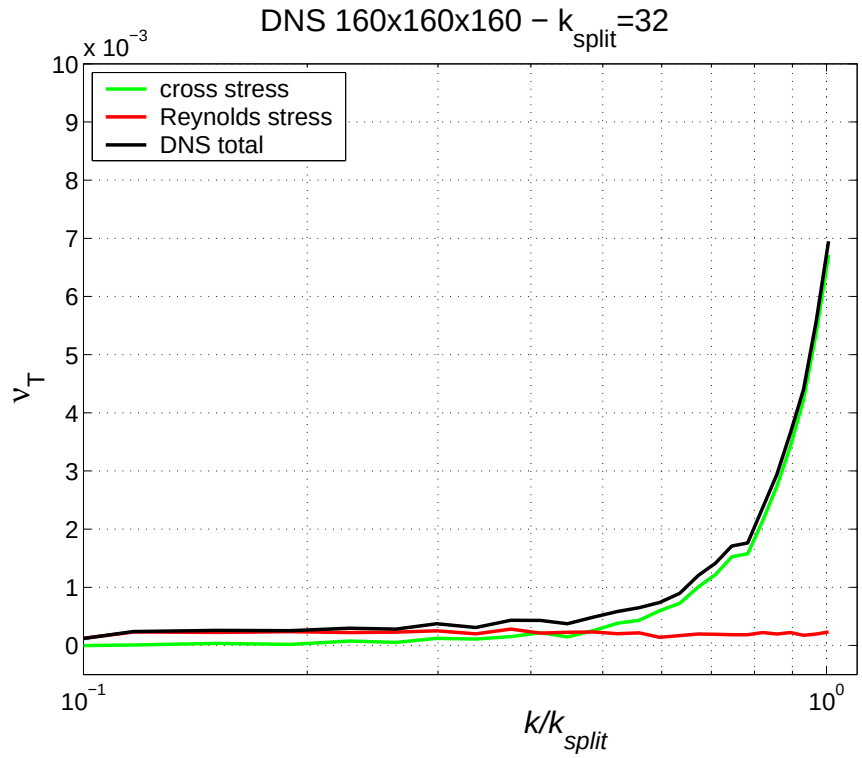

Figure 15.1: Spectral eddy viscosity for a splitting of the scales in which Fourier modes in a spectral ball of radius $k_{split} = 32$ are considered as coarse scales and the remaining complement to a total of 160^3 modes – equivalent to 79^3 wavenumbers – represent the fine scales. The total transfer of energy is in black, green represent the cross-stress contribution to the total transfer, red depicts the Reynolds stress contribution. Simulation performed by A.A. Wray.

and decays to zero as the simulation progresses. The initial condition was generated by a synthetic turbulent flow with prescribed spectrum and random phases, but the statistical sampling was started only later on, to allow the Navier-Stokes dynamics to develop adequately. The value of Re_λ refers then to the time at which the statistical sampling began. Figure 15.1 presents the so-called *spectral eddy viscosity*, a measure of the energy transfer from the coarse scales to the fine scales. The spectral eddy viscosity relative to a certain wavenumber magnitude $|k|$ quantifies how much energy is transferred from the coarse scales on the sphere of radius k (defined by the wavenumbers k_x, k_y, k_z such that $\sqrt{k_x^2 + k_y^2 + k_z^2} = |k|$) to the fine scales past the cutoff k_{split}. For a detailed explanation about energy transfers in homogeneous isotropic turbulence, the reader can refer to [29]. In particular, Figure 15.1 shows how much energy is transferred from each k-sphere of wavenumbers in the ball of radius $k_{split} = 32$ to wavenumbers outside that

ball (the fine scales, i.e., the complement of the ball to the total of 160^3 modes computed, equivalent to 79^3 wavenumbers).

It is very noticeable that most of the energy transfer from the coarse scales is due to cross-stress terms, which are actually represented by the linear part of the turbulent stresses. Turbulent flows are dominated by the Kolmogorov cascade, an inertial transfer of energy from coarse to fine scales: as the Reynolds number increases, the cascade spans a broader and broader range of scales.

If the space of coarse scales is reduced, the effect of the Reynolds stresses becomes more evident, in the sense that the *plateau* in red rises to higher values, until the profile of the total transfer almost coincides with it. In particular we can outline to trends for the energy transfer, as we vary the size of the set of coarse scales in a direct numerical simulation (DNS):

1. If the coarse space is much smaller than the total space of modes, the energy transfer is dominated by the highly nonlinear Reynolds stress dynamics.

2. If the coarse space is covering a non-negligible portion of the total amount of modes computed, the energy transfer is dominated by the linear dynamics of the cross stresses.

Thus, it appears clear that a linearization approach for the cross stresses can be applied in the second case, performing a large eddy simulation without incorporating a model of the Reynolds stresses. It should also be observed that in general, the space of coarse scale has to be very small to start observing strong nonlinear dynamics, and for many large eddy simulations reported in the literature, the turbulent flows are typically in the second situation rather then the first, or somewhat in between the two.

15.2 A new multiscale approach

In practice, only the first term in the asymptotic expansion (13.20) will be accounted for at the modeling level, and it appears clear that the first subgrid equation (13.27) in the expansion corresponds to a local linearization of the Navier-Stokes equations.

SUPG approaches can be interpreted as ways of approximately solving the mentioned equation, by means of the substitution:

$$\boldsymbol{U}' \approx \boldsymbol{U}'_1 \approx -\boldsymbol{\tau}\boldsymbol{Res}(\overline{\boldsymbol{U}}) \tag{15.1}$$

where (15.1) has to be intended in a variational/distributional sense. $\boldsymbol{Res}(\overline{\boldsymbol{U}})$ is the residual of the coarse scale equation, and $\boldsymbol{\tau}$ can be thought of as an approximation of the subgrid Green's integral operator (for an extensive discussion on how to approximate the Green's integral operator and a complete list of references, refer to the chapter in the Encyclopedia of Computational Mechanics by Hughes, Scovazzi and Franca [37]). Substitution of the proposed approximation for $\boldsymbol{U}'$ into (13.12) yields a subgrid-scale model. This procedure can be generalized to all the terms in the asymptotic cascade, since the right hand side of the equation for $\boldsymbol{U}'_2$ is a function of $\boldsymbol{U}'_1$, so that, recursively, additional contributions can be added to the approximation for $\boldsymbol{U}'$.

Notice that the local linearization is an approach substantially different from *Rapid Distortion Theory* (RDT, see [39] for a comprehensive review), since RDT is usually applied to the theoretical analysis of flow instabilities and turbulent shear/boundary layers under restrictive conditions, and, furthermore, the linearization is about the statistical mean flow, and not the coarse scale solution. Finally, the coarse-scale dynamics is fully nonlinear while RDT is, as a matter of fact, a linearization of the complete Navier-Stokes equations.

15.3 Applications to bypass transition studies

Bypass transition is a phenomenon that occurs whenever the free-stream of a laminar base flow is injected with sufficiently energetic *germs* of turbulence. Under such conditions, the classical Orr-Sommerfeld mechanism for instability and transition to turbulence is bypassed and the flow undergoes an early transition, due to the formation of streaky structures, also termed Klebanoff modes. Figure 15.3 shows the typical patterns of the flow. In particular, the following simulations refer to a case study where the turbulent intensity is about 3% of the free-stream velocity U_0, analogously to the experiments performed by Roach and Brierley [62].

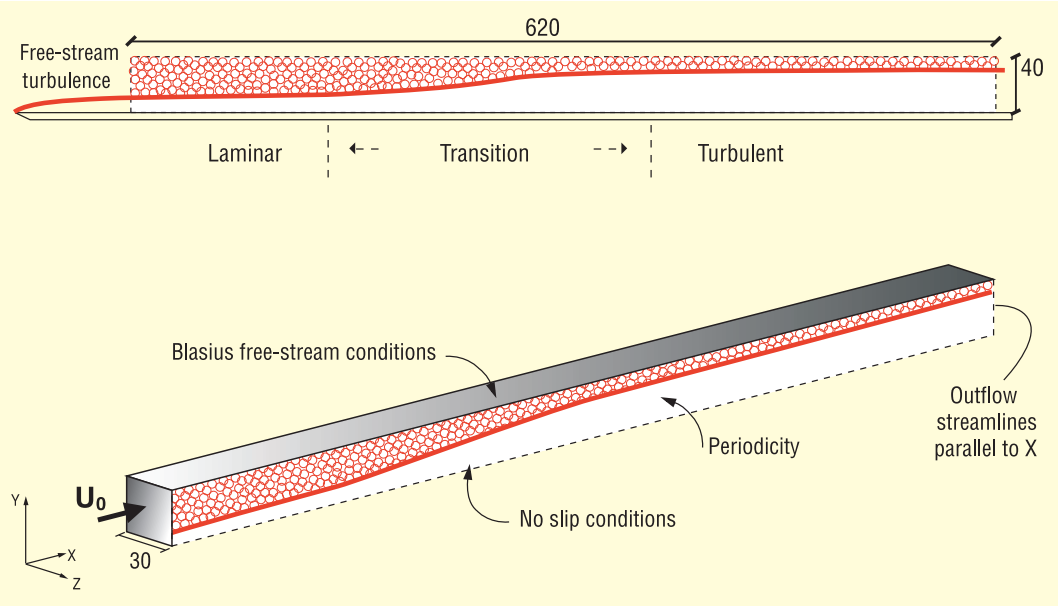

Figure 15.2: Sketch of the computational domain for the large eddy simulation studies of bypass transition. The domain is a parallelepiped of dimensions $L_x \times L_y \times L_z = 620\ \delta_0 \times 40\ \delta_0 \times 30\ \delta_0$. δ_0 is the inflow boundary layer displacement thickness (laminar flow), and the inflow momentum thickness is $\theta_0 = 0.1336\ \delta_0$. (Courtesy of V.M. Calo.)

The reader is encouraged to read the Ph.D. thesis of V.M. Calo [7], who performed the simulations using a modification of the finite-volume/finite-difference implementation of the problem and kindly provided the pictures presented herein. The thesis of V.M. Calo includes an exhaustive discussion about the computational burdens successfully overcome in implementing the problem correctly from a numerical and physical perspective. The reader should also consult [40, 41], for an earlier direct numerical simulation aimed at analyzing the physics of bypass transition.

Simulation parameters

L_x	$620\ \delta_0$
L_y	$40\ \delta_0$
L_z	$30\ \delta_0$
θ_0	$0.1336\ \delta_0$
Re_{δ_0}	795

Resolutions studied

Test	N_x	N_y	N_z	mesh ratio
Full DNS	2048	180	192	1
LES 1/2	1024	90	96	$(1/2)^3$
LES 1/4	512	48	48	$(1/4)^3$
LES 1/8	256	32	32	$(1/8)^3$

Table 15.1: Several bypass transition simulations were performed. The results presented herein are relative to the LES 1/4 denomination, corresponding to a coarsening of the mesh by a factor 1/4 in each of the spatial directions. The mesh is stretched at the wall using an hyperbolic tangent map.

The numerical simulations have been performed using a variant of a classical second-order, fully-staggered, central difference scheme (see [55], and references therein for an account of the original scheme and subsequent development of the numerical architecture). The mesh was logically structured and the resolution used in the simulations is detailed in Table 15.1.

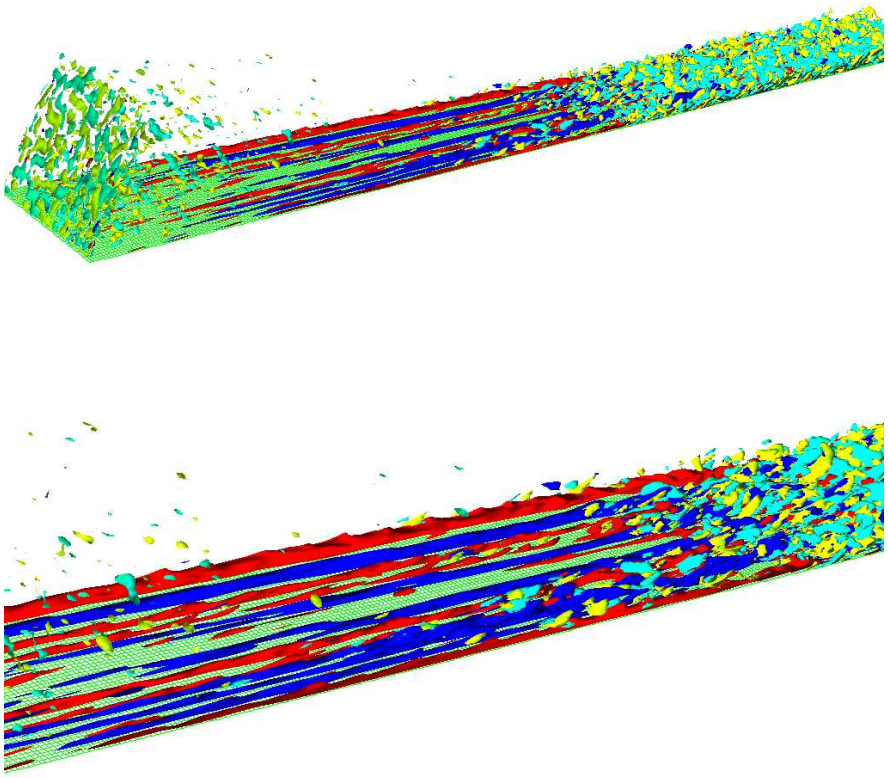

Figure 15.3: Full Direct Numerical Simulation. Visualization of the streamwise velocity fluctuations (blue/red are positive/negative fluctuations) generated by germs of turbulence at the inflow in the free-stream (positive/negative vorticity is plotted in cyan/yellow). The decay of free-stream turbulence as the flow is advected downstream of the inflow is visible in the upper plot. Streaky structures form at the bottom of the boundary layer (see the zoomed view on the lower visualization), and develop instabilities that eventually transition to turbulence. (Plots courtesy of V.M. Calo.)

Velocity and pressure equations are decoupled by means of an approximate factorization technique, leading to a Poisson problem for the pressures. The scheme was proved energy-conservative on uniform meshes. The boundary conditions are outlined in the

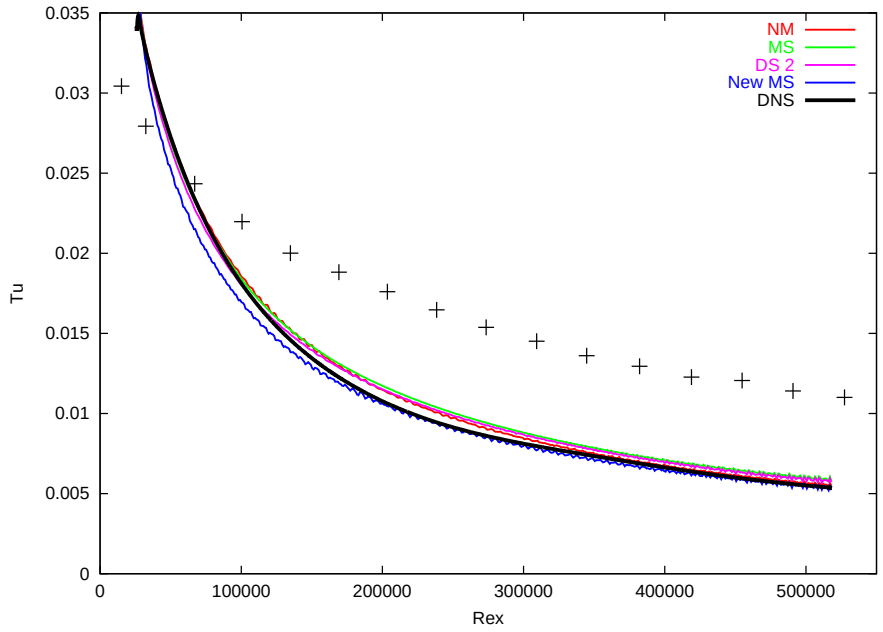

Figure 15.4: Turbulent intensity decay as a function of $Re_x = U_\infty x/\nu$, for DNS and various LES (corresponding to the LES 1/4 mesh denomination, differing for the type of model used). Red: no model; green: multiscale model as in Hughes et al. [30]; blue: new model based on SUPG stabilization and asymptotic expansions; magenta: dynamic model of Germano et. al[21]; black: DNS of Jacobs and Durbin [40, 41]; +: experimental measurements from [62]. It was found of crucial importance to match the free-stream turbulence decay, in order to appropriately characterize the transition dynamics, for all methods and all mesh refinements. This is a very interesting example of close coupling between numerics and physics. (Plot courtesy of V.M. Calo.)

sketch in Figure 15.2. The computational domain is a parallelepiped of dimensions $L_x \times L_y \times L_z = 620\ \delta_0 \times 40\ \delta_0 \times 30\ \delta_0$. δ_0 is the inflow boundary layer displacement thickness, defined as the distance from the wall at which the streamwise velocity u has regained 99% of the free-stream value. The inflow momentum thickness is $\theta_0 = 0.1336\ \delta_0$, according to the theory of laminar boundary layers.

Special care was devoted to the inflow conditions, since it was observed that the free-stream turbulent intensity was spuriously damped by the numerics across the layer of inflow volumes (the layer of volumes which have one of their faces laying on the inflow boundary). Matching the turbulent decay with respect to a reference direct numerical simulation (DNS) [40, 41] rather than the inflow turbulent intensity produced a dramatic improvement in the results, also in terms of trends in mesh refinement studies.

A plot of the decay of turbulent intensity from the inflow to downstream is presented in Figure 15.4, in the case of a the mesh for the LES 1/4 simulations. It is felt important to say a few words clarifying the purpose of the simulations performed. As one can notice in Figure 15.4, the experimental curve for the decay of the free-stream turbulence is far apart from the corresponding DNS curve. Hence, there are two separate issues to be considered here: The consistency of DNS with experimental conditions and the effect of LES modeling. The first of the two mentioned issues will *not* be addressed here, since it investigates the meaning of closeness between experimental and numerical conditions in turbulence, a field of research in itself. The second is instead the matter of the following discussion, and one should therefore compare results to the DNS output, for which the simulation conditions can be monitored and matched more closely. Therefore, the reader should keep in mind that although experimental results are presented for the sake of completeness, only the DNS results should be used as a benchmark.

In order to implement the proposed model, a reconstruction of the residual was needed, a non-trivial aspect in the case of a finite-volume/finite-difference implementation, for which a standard variational formulation is usually not available. By interpreting the finite volume formulation as a variant of a finite element method *à la* Raviart-Thomas, the residual was reconstructed as a cell-centered quantity. Although the reconstruction may not be as precise as in the case of a standard finite element formulation, the numerical performance of the implementation proves its effectiveness.

15.3.1 Analysis of the results.

Figure 15.5 is for the large eddy simulation denoted "LES 1/4", for which the mesh is coarsened by a factor of four in each Cartesian direction. It shows the non-dimensional skin friction coefficient defined as:

$$C_f = \frac{\tau_w}{1/2\rho U_0^2} \tag{15.2}$$

where τ_w is the shear stress at the wall, ρ is the density of the fluid, assumed constant in the present case, and U_0 is the free-stream velocity. It was found that in the case of the test termed LES 1/8, the numerics were incapable of representing either the laminar

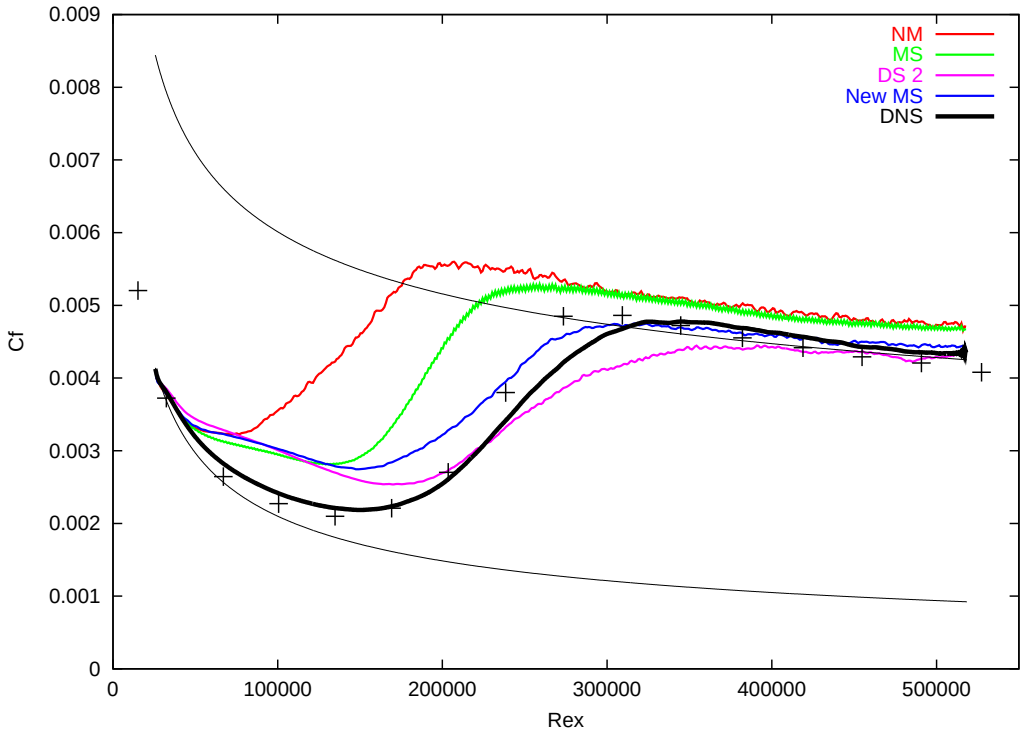

Figure 15.5: Skin friction coefficient as a function of $Re_x = U_\infty x/\nu$. Red: no model; green: multiscale model as in Hughes et al. [30]; blue: new model based on SUPG stabilization and asymptotic expansions; magenta: dynamic model of Germano et. al[21]; black: DNS of Jacobs and Durbin [40, 41]; +: experimental measurements from [62]. (Plot courtesy of V.M. Calo.)

region of the boundary layer or the free-stream turbulence decay due to too few points in the wall normal direction. This was not an issue of LES modeling, rather, a matter of turbulence sustainability in the free-stream flow.

The dynamic model produced excessive smearing on the skin friction coefficient c_f in the transitional region, and the numerical results were also found very dependent on the definition of the mesh characteristic length scale (needed to set up the least square procedure to determine the dynamic coefficient). In addition, there was pronounced dependence of the results on mesh refinement [7]. The combination of the three aspects proved the dynamic model less robust than the newly proposed method, which exhibited correct trends across all meshes.

15.4 Conclusions

A new method, based on ideas inherited from SUPG stabilization and Green's function concepts, was developed and tested on the challenging problem of bypass transition, and superior results were noted. In particular, the new concept bridges the gap between subgrid-scale stabilization and turbulence modeling.

The potential for a new pathway in LES modeling was confirmed in numerical studies, although more testing is needed, on a wider variety of physical flows and numerical frameworks (e.g., spectral, finite difference and finite element methods). In particular, the enormous (and often underestimated) impact that the numerical discretization method has on the results was experienced.

Finally, although the model is based upon the assumption that the subgrid scales are effectively small with respect to the mesh resolved scales, the tests performed could not assess the magnitude of "small". It is expected that as the mesh is successively coarsened, the linearized framework would be less and less appropriate, but, due to issues related to free-stream turbulence sustainability, it was not possible to achieve this limit in the present computations. This is in fact a good sign for the effectiveness of the model, and probably an indicator of its flexibility in more challenging conditions.

Further research will therefore be oriented to the implementation of the method under a wider variety of conditions, either from the physical, numerical, or combined perspectives.

Appendix A

An Algorithm for the Square Root of a 3×3 Positive Definite Matrix

The present algorithm is presented in detail by Franca [16].

Step 1: Compute invariants $I_{\boldsymbol{B}}$, $II_{\boldsymbol{B}}$ and $III_{\boldsymbol{B}}$ of $\boldsymbol{B}$:

$$I_{\boldsymbol{B}} = \mathrm{tr}\boldsymbol{B} \tag{A.1}$$

$$II_{\boldsymbol{B}} = \frac{1}{2}\left(I_{\boldsymbol{B}}^2 - \mathrm{tr}(\boldsymbol{B}^2)\right) \tag{A.2}$$

$$III_{\boldsymbol{B}} = \det\boldsymbol{B} \tag{A.3}$$

$$k = I_{\boldsymbol{B}}^2 - 3II_{\boldsymbol{B}} \tag{A.4}$$

Step 2: Check isotropy of the tensor $\boldsymbol{B}$, if $k \leq$ toll, with toll close to machine accuracy, go to Step 3. Otherwise skip Step 3 and go to Step 4.

Step 3: *Isotropic case.* Compute $\boldsymbol{B}^{1/2}$ and $\boldsymbol{B}^{-1/2}$:

$$\lambda = \left(\frac{I_{\boldsymbol{B}}}{3}\right)^{1/2} \tag{A.5}$$

$$\boldsymbol{B}^{1/2} = \lambda\, \boldsymbol{I}_{3\times 3} \tag{A.6}$$

$$\boldsymbol{B}^{-1/2} = \lambda^{-1}\, \boldsymbol{I}_{3\times 3} \tag{A.7}$$

End the computation.

Step 4: *Non-isotropic case.* Compute the largest eigenvalue (Cartan method):

$$l = I_{\boldsymbol{B}}\left(I_{\boldsymbol{B}}^2 - \frac{9}{2}II_{\boldsymbol{B}}\right) + \frac{27}{2}III_{\boldsymbol{B}} \tag{A.8}$$

$$\phi = \arccos\left(\frac{l}{k^{3/2}}\right) \tag{A.9}$$

$$\lambda^2 = \frac{1}{3}\left(I_{\boldsymbol{B}} + 2k^{1/2}\cos\left(\frac{\phi}{3}\right)\right) \tag{A.10}$$

Step 5: *Non-isotropic case.* Compute invariants of $\boldsymbol{U} = \boldsymbol{B}^{1/2}$ (λ is the positive root of λ^2 since $\boldsymbol{B}^{1/2} \geq \mathbf{0}$, by definition):

$$III_{\boldsymbol{U}} = III_{\boldsymbol{B}}^{1/2} \tag{A.11}$$

$$I_{\boldsymbol{U}} = \lambda + \left(-\lambda^2 + I_{\boldsymbol{B}} + \frac{2\ III_{\boldsymbol{U}}}{\lambda}\right)^{1/2} \tag{A.12}$$

$$II_{\boldsymbol{U}} = \frac{I_{\boldsymbol{U}}^2 - I_{\boldsymbol{B}}}{2} \tag{A.13}$$

Step 6: *Non-isotropic case.* Compute $\boldsymbol{U} = \boldsymbol{B}^{-1/2}$:

$$\boldsymbol{B}^{1/2} = \frac{1}{I_{\boldsymbol{U}}\ II_{\boldsymbol{U}} - III_{\boldsymbol{U}}}\left(I_{\boldsymbol{U}}\ III_{\boldsymbol{U}}\ \boldsymbol{I}_{3\times3} + (I_{\boldsymbol{U}}^2 - II_{\boldsymbol{U}})\boldsymbol{B} - \boldsymbol{B}^2\right) \tag{A.14}$$

$$\boldsymbol{B}^{-1/2} = \frac{1}{III_{\boldsymbol{U}}}\left(II_{\boldsymbol{U}}\ \boldsymbol{I}_{3\times3} - I_{\boldsymbol{U}}\boldsymbol{B}^{1/2} + \boldsymbol{B}\right) \tag{A.15}$$

End computation.

Appendix B

Numerical Implementation of the Multiscale Analysis for the Burgers' Equation

In this section of the appendix, the details of the numerical testing performed on the nonlinear Burgers' equation are presented. The numerical time integrator used is a generalized trapezoidal rule combined with a Newton solver. Piece-wise linear continuous interpolation has been used for the discretization of the solution.

B.1 The dimensional problem.

The dimensional Burgers' equation reads:

$$\tilde{u}_{,\tilde{t}} + \left(\frac{\tilde{u}^2}{2}\right)_{,\tilde{x}} = \nu \tilde{u}_{,\tilde{x}\tilde{x}} + \tilde{f} \tag{B.1}$$

The forcing $\tilde{f}$ is taken to be a Gaussian, white-in-time field with variance given by

$$< f(\tilde{x},\tilde{t}) f(\tilde{x}',\tilde{s})) >= F(|\tilde{x}-\tilde{x}'|)\delta(\tilde{t}-\tilde{s}) \tag{B.2}$$

where $< \cdot >$, stands for an ensemble average, equivalent to an average in time due to the statistical stationarity of the problem.

The spectrum of the forcing must also respect the following constraints:

$$\int_0^\infty \tilde{k}^2 \tilde{F}(\tilde{k}) = \tilde{\xi}_f^3 \tag{B.3}$$

$$|\hat{\tilde{f}}|^2 \propto \tilde{F} \sim \tilde{k}^4 e^{-\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} \tag{B.4}$$

To be clear regarding the notation used, here " $\tilde{\cdot}$ " means a dimensional quantity, $\tilde{k}_f$ is the (dimensional) wavenumber about which the spectrum $\tilde{\hat{F}}$ peaks, $\tilde{\xi}_f$ is a scale for the velocity gradient of the solution $\tilde{u}$.

B.2 The non-dimensional problem.

We now would like to rephrase the Burgers' equation in non-dimensional form, the only one that make sense from a physical point of view, since it highlights the only non-dimensional parameters of importance for the system. We therefore introduce scales for time ($\tilde{T}$), length ($\tilde{L}$), velocity ($\tilde{U}$), whose magnitude will be specified later, when intuition about the dominant features of the system in question will be clearer. For the moment we will just fix:

$$\tilde{u} = \tilde{U}u \tag{B.5}$$

$$\tilde{x} = \tilde{L}x \tag{B.6}$$

$$\tilde{t} = \tilde{T}t = \tilde{L}/\tilde{U}t \tag{B.7}$$

$$\tilde{k} = k/\tilde{L} \tag{B.8}$$

$$\tilde{f} = \tilde{f}_0 f \tag{B.9}$$

Notice that in the normalization used for the time-scale, we are implying that the time scale is dominated by convection effects. We can therefore write the Burgers' equation as follows:

$$\frac{\tilde{U}^2}{\tilde{L}} u_{,t} + \frac{\tilde{U}^2}{\tilde{L}} \left(\frac{u^2}{2}\right)_{,x} = \nu \frac{\tilde{U}}{\tilde{L}^2} u_{,xx} + \tilde{f}_0 f \tag{B.10}$$

Dividing (B.10) by $\frac{\tilde{U}^2}{\tilde{L}}$, we get:

$$u_{,t} + \left(\frac{u^2}{2}\right)_{,x} = \frac{1}{Re} u_{,xx} + \frac{\tilde{f}_0 \tilde{L}}{\tilde{U}^2} f \tag{B.11}$$

where $Re = \frac{\tilde{U}\tilde{L}}{\nu}$, is the Reynolds number. We still need to define the scales with which we are normalizing the equation, and that the term $G = \frac{\tilde{f}_0 \tilde{L}}{\tilde{U}^2}$ represents the ratio between the forcing and the convection effects in the system. In our particular case the motion is generated by the forcing and we would expect

$$G = O(1). \tag{B.12}$$

Let us now specify the scales involved in the system dynamics. Since the forcing acts with a spectrum peaking at $\tilde{k}_f$, it is natural to use $\tilde{L} = 1/\tilde{k}_f$. Also, due to (B.3) and the fact that $\tilde{\xi}_f$ has the dimension of a velocity gradient, we would have:

$$\tilde{U} = \tilde{\xi}_f/\tilde{k}_f \tag{B.13}$$

$$\tilde{L} = 1/\tilde{k}_f \tag{B.14}$$

$$\tilde{T} = \tilde{\xi}_f \tag{B.15}$$

and consequently $\tilde{f}_0 \sim \tilde{\xi}_f^2/\tilde{k}_f$ (so far, by "$\sim$" we mean *"has the same dimension of"*, with no implication regarding the order of magnitude).

B.3 Semi-discrete Formulation

Let u_k represent the algorithmic solution at time t_k. The time step is denoted $\Delta t = t_{k+1} - t_k$. It proves convenient to employ the jump operator, viz.,

$$[v] = v^{k+1} - v^k \tag{B.16}$$

$$\langle v \rangle = \frac{v^{k+1} + v^k}{2} \tag{B.17}$$

$$\langle v \rangle_\alpha = \alpha v^{k+1} + (1-\alpha) v^k \qquad \alpha \in [0,1] \tag{B.18}$$

$$(\langle v \rangle_\alpha)_{,x} = \langle v_{,x} \rangle_\alpha \tag{B.19}$$

The algorithm implemented is a semi-discrete variational formulation of (14.4), given by the well-known generalized trapezoidal rule:

$$\frac{1}{\Delta t}\left(w,[u]\right)_\Omega - \left(w_{,x},\frac{\langle u\rangle_\alpha^2}{2}\right)_\Omega + \frac{1}{Re}\left(w_{,x},(\langle u\rangle_\alpha)_{,x}\right)_\Omega = \left(w,f^{k+1/2}\right)_\Omega \tag{B.20}$$

B.3.1 Large scale equation

Separation of Scales

The usual decomposition into coarse an fine scales yields:

$$\begin{aligned}
[\overline{u}+u'] &= [\overline{u}]+[u'] = \overline{u}^{k+1}-\overline{u}^{k}+[u'] && \text{(B.21)}\\
\langle \overline{u}+u'\rangle_\alpha &= \alpha\overline{u}^{k+1}+(1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha && \text{(B.22)}\\
\langle \overline{u}+u'\rangle_\alpha^2 &= \left(\alpha\overline{u}^{k+1}+(1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha\right)^2\\
&= \alpha^2(\overline{u}^{k+1})^2+2\alpha\left((1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha\right)\overline{u}^{k+1}\\
&\quad +\left((1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha\right)^2 && \text{(B.23)}
\end{aligned}$$

We then substitute (B.22), (B.23) and (B.21) into (B.20):

$$\begin{aligned}
&\frac{1}{\Delta t}\left(w,\overline{u}^{k+1}-\overline{u}^{k}+[u']\right)_\Omega + \frac{1}{Re}\left(w_{,x},\alpha\overline{u}^{k+1}_{,x}+(1-\alpha)\overline{u}^{k}_{,x}+\langle u'_{,x}\rangle_\alpha\right)_\Omega\\
-\;&\frac{1}{2}\left(w_{,x},\alpha^2(\overline{u}^{k+1})^2+2\alpha\left((1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha\right)\overline{u}^{k+1}\right)_\Omega\\
-\;&\frac{1}{2}\left(w_{,x},\left((1-\alpha)\overline{u}^{k}+\langle u'\rangle_\alpha\right)^2\right)_\Omega\\
=\;&\left(\overline{w},\overline{f}^{k+1/2}\right)_\Omega+\left(\overline{w},f'^{k+1/2}\right)_\Omega
\end{aligned} \tag{B.24}$$

The proposed time discretization is second order in time for the Burger's equation problem in the case $\alpha = 1/2$. In fact, we can easily check this claim by a Taylor expansion of the solution around the point $t^{k+\frac{1}{2}}$. All the linear terms are delivering second-order accuracy, the only term require special care is the nonlinear term.

$$\begin{aligned}
u^{k+1} &= u^{k+1/2}+u^{k+1/2}_{,t}\frac{\Delta t}{2}+u^{k+1/2}_{,tt}\frac{(\Delta t/2)^2}{2!}+O(\Delta t^3) && \text{(B.25)}\\
u^{k} &= u^{k+1/2}-u^{k+1/2}_{,t}\frac{\Delta t}{2}+u^{k+1/2}_{,tt}\frac{(\Delta t/2)^2}{2!}+O(\Delta t^3) && \text{(B.26)}
\end{aligned}$$

from which, collecting like terms and simplifying,

$$u^k u^{k+1} = \left(u^{k+1/2}\right)^2 + O(\Delta t^2) \tag{B.27}$$

The numerical tests and experiments performed were *always* run at $\alpha = 1/2 + \Delta t$, for which second-order accuracy is still preserved and stability is achieved (the stable range of values for α is $\alpha \in (1/2, 1]$).

Assembly of the large scale system

If we decompose the space of the test functions, according to (14.11), and we assume $\overline{w}$ and w' linearly independent, we obtain a formula for the large scale equation:

$$B_{LIN}(\overline{w}, \overline{u}^{k+1}; \langle u' \rangle_\alpha, \overline{u}^k) + B_{NL}(\overline{w}, \overline{u}^{k+1}) = F(\overline{w}; \langle u' \rangle_\alpha, \overline{u}^k) \tag{B.28}$$

where

$$\begin{aligned} B_{LIN}(\overline{w}, \overline{u}^{k+1}; \langle u' \rangle_\alpha, \overline{u}^k) &= \frac{1}{\Delta t}\left(\overline{w}, \overline{u}^{k+1}\right)_\Omega + \frac{\alpha}{Re}\left(\overline{w}_{,x}, \overline{u}^{k+1}_{,x}\right)_\Omega \\ &\quad -\alpha(\overline{w}_{,x}, \{\langle u' \rangle_\alpha + (1-\alpha)\overline{u}^k\}\ \overline{u}^{k+1})_\Omega \end{aligned} \tag{B.29}$$

$$B_{NL}(\overline{w}, \overline{u}^{k+1}) = -\frac{\alpha^2}{2}\left(\overline{w}, \left(\overline{u}^{k+1}\right)^2\right)_\Omega \tag{B.30}$$

$$\begin{aligned} F(\overline{w}; \langle u' \rangle_\alpha, \overline{u}^k) &= \frac{1}{\Delta t}(\overline{w}, \overline{u}^k - [u'])_\Omega - \frac{1}{Re}(\overline{w}_{,x}, (1-\alpha)\overline{u}^k_{,x} + \langle u'_{,x} \rangle_\alpha)_\Omega \\ &\quad + \left(\overline{w}_{,x}, \frac{\langle u' \rangle_\alpha^2}{2} + \frac{(1-\alpha)^2}{2}(\overline{u}^k)^2\right)_\Omega \\ &\quad + \left(\overline{w}_{,x}, (1-\alpha)\langle u' \rangle_\alpha \overline{u}^k\right)_\Omega \\ &\quad + \left(\overline{w}, \overline{f}^{k+1/2}\right)_\Omega + \left(\overline{w}, f'^{k+1/2}\right)_\Omega \end{aligned} \tag{B.31}$$

We can now introduce two bases functions, for the spaces $\overline{\mathcal{V}}$ and $\mathcal{V}'$ respectively:

$$\overline{w}(x,t) = \sum_{\overline{A}=1}^{\overline{n}_{\rm np}} \overline{N})_{\overline{A}}(x)\overline{w}_{\overline{A}}(t) = \overline{\boldsymbol{N}}\ \overline{\boldsymbol{W}} \tag{B.32}$$

$$\overline{u}(x,t) = \sum_{\overline{A}=1}^{\overline{n}_{\rm np}} \overline{N}_{\overline{A}}(x)\overline{w}_{\overline{A}}(t) = \overline{\boldsymbol{N}}\ \overline{\boldsymbol{U}} \tag{B.33}$$

$$w'(x,t) = \sum_{A'=1}^{n'_{\rm np}} N'_{A'}(x) w'_{A'}(t) = \boldsymbol{N'W'} \tag{B.34}$$

$$u'(x,t) = \sum_{A'=1}^{n'_{\rm np}} N'_{A'}(x) u'_{A'}(t) = \boldsymbol{N'U'} \tag{B.35}$$

Therefore (B.29), (B.30), and (B.31) can be rewritten in matrix form as follows:

$$\left\{ \boldsymbol{A}^{LIN}_{\bar{A}\bar{B}} \left[\langle \boldsymbol{U'} \rangle_\alpha, \overline{\boldsymbol{U}}^k \right] + \boldsymbol{A}^{NL}_{\bar{A}\bar{B}} \left[\overline{\boldsymbol{U}}^{k+1} \right] \right\} \overline{\boldsymbol{U}}^{k+1}_{\bar{B}} = \boldsymbol{F}_{\bar{A}} \left[\langle \boldsymbol{U'} \rangle_\alpha, \overline{\boldsymbol{U}}^k \right] \tag{B.36}$$

where

$$\begin{aligned} \boldsymbol{F}_{\bar{A}} \left[\langle \boldsymbol{U'} \rangle_\alpha, \overline{\boldsymbol{U}}^k \right] = & \left\{ \boldsymbol{K}^{\Delta t} - (1-\alpha) \left\{ \boldsymbol{K}^{Re} + \frac{(1-\alpha)}{2} \boldsymbol{K}^{\overline{u}^k} + \boldsymbol{K}^{\langle u' \rangle_\alpha} \right\} \right\}_{\bar{A}\bar{B}} \overline{\boldsymbol{U}}^k_{\bar{B}} \\ & - \boldsymbol{K}^{\Delta t}_{\bar{A}B'} [\boldsymbol{U'}_{B'}] - \left\{ \boldsymbol{K}^{Re} + \frac{1}{2} \boldsymbol{K}^{\langle u' \rangle_\alpha} \right\}_{\bar{A}B'} \langle \boldsymbol{U'}_{B'} \rangle_\alpha \\ & + \boldsymbol{M}_{\bar{A}\bar{B}} \overline{\boldsymbol{F}}^{k+1/2}_{\bar{B}} + \boldsymbol{M}_{\bar{A}B'} \boldsymbol{F'}^{k+1/2}_{B'} \end{aligned} \tag{B.37}$$

$$\boldsymbol{A}^{LIN}_{\bar{A}\bar{B}} \left[\langle \boldsymbol{U'} \rangle_\alpha, \overline{\boldsymbol{U}}^k \right] = \left\{ \boldsymbol{K}^{\Delta t} + \alpha \boldsymbol{K}^{Re} + \alpha \boldsymbol{K}^{\{\langle u' \rangle_\alpha + (1-\alpha)\overline{u}^k\}} \right\}_{\bar{A}\bar{B}} \tag{B.38}$$

$$\boldsymbol{A}^{NL}_{\bar{A}\bar{B}} \left[\overline{\boldsymbol{U}}^{k+1} \right] = \frac{\alpha^2}{2} \boldsymbol{K}^{\overline{u}^{k+1}}_{\bar{A}\bar{B}} \tag{B.39}$$

with

$$\boldsymbol{K}^{\Delta t}_{AB} = \frac{1}{\Delta t} \boldsymbol{M}_{AB} = \left(N_A, \frac{N_B}{\Delta t} \right)_\Omega \tag{B.40}$$

$$\boldsymbol{K}^{Re}_{AB} = \frac{1}{Re} \left(N_{A,x}, N_{B,x} \right)_\Omega \tag{B.41}$$

$$\boldsymbol{K}^{v}_{AB} = - \left(N_{A,x}, v N_B \right)_\Omega \tag{B.42}$$

$$\boldsymbol{M}_{AB} = (N_A, N_B)_\Omega \tag{B.43}$$

Newton iteration

We implement for the large scale equation a Newton iteration nonlinear solver, in order to preserve the nature of the nonlinear interaction. Let us set:

$$\boldsymbol{G}^{\nu} = \left\{ \boldsymbol{A}^{LIN}_{\bar{A}\bar{B}} \left[\langle \boldsymbol{U}' \rangle_{\alpha}, \overline{\boldsymbol{U}}^{k} \right] + \boldsymbol{A}^{NL}_{\bar{A}\bar{B}} \left[\overline{\boldsymbol{U}}^{\nu} \right] \right\} \overline{\boldsymbol{U}}^{\nu}_{\bar{B}} - F_{\bar{A}} \left[\langle \boldsymbol{U}' \rangle_{\alpha}, \overline{\boldsymbol{U}}^{k} \right] \tag{B.44}$$

The Newton iteration reads:

$$\left\{ \begin{array}{rcl} \delta\overline{\boldsymbol{U}} & = & -\left(\dfrac{\partial \boldsymbol{G}}{\partial \overline{\boldsymbol{U}}} \Big|_{\nu} \right)^{-1} \boldsymbol{G}^{\nu} \\ \overline{\boldsymbol{U}}^{\nu+1} & = & \overline{\boldsymbol{U}}^{\nu} + \delta\overline{\boldsymbol{U}} \end{array} \right. \tag{B.45}$$

where

$$\left. \frac{\partial \boldsymbol{G}}{\partial \overline{\boldsymbol{U}}} \right|_{\nu} = \left\{ \boldsymbol{A}^{LIN}_{\bar{A}\bar{B}} \left[\langle \boldsymbol{U}' \rangle_{\alpha}, \overline{\boldsymbol{U}}^{k} \right] + \alpha^2 \boldsymbol{K}^{\overline{u}^{\nu}} \right\}_{\bar{A}\bar{B}} \tag{B.46}$$

is the Jacobian matrix generated by differentiating with respect to $\overline{\boldsymbol{U}}^{\nu}$ the vector $\boldsymbol{G}$. Note that the Newton iteration requires the values of $\langle \boldsymbol{U}' \rangle_{\alpha}, \overline{\boldsymbol{U}}^{k}$ as data. Therefore at each step of the iteration the term $\langle \boldsymbol{U}' \rangle_{\alpha}$ must be updated through the small scale equation.

B.3.2 Small scale equation

For the small scale equation, the use an asymptotic expansion method (a linearization procedure) bypasses the Newton iteration.

Separation of scales

In the case of the small scale equations it will prove more convenient to decompose the terms as follows:

$$\begin{aligned} [\overline{u} + u'] &= [\overline{u}] + [u'] = [\overline{u}] + u'^{k+} - u'^{k+} && \text{(B.47)} \\ \langle \overline{u} + u' \rangle_{\alpha} &= \langle \overline{u} \rangle_{\alpha} + \alpha u'^{k+1} + (1-\alpha) u'^{k} && \text{(B.48)} \\ \langle \overline{u} + u' \rangle^2_{\alpha} &= \left(\alpha u'^{k+1} + (1-\alpha) u'^{k} + \langle \overline{u} \rangle_{\alpha} \right)^2 \\ &= \alpha^2 (u'^{k+1})^2 + 2\alpha \left((1-\alpha) u'^{k} + \langle \overline{u} \rangle_{\alpha} \right) u'^{k+1} \\ &\quad + \left((1-\alpha) u'^{k} + \langle \overline{u} \rangle_{\alpha} \right)^2 && \text{(B.49)} \end{aligned}$$

We then substitute (B.48), (B.49) and (B.47) into (B.20), in which we replace w bay w', since we are interested in the small scales:

$$\begin{aligned}
&\frac{1}{\Delta t}\left(w', u'^{k+1} - u'^{k} + [\overline{u}]\right)_\Omega + \frac{1}{Re}\left(w'_{,x}, \alpha u'^{k+1}_{,x} + (1-\alpha)u'^{k}_{,x} + < \overline{u}_{,x} >_\alpha\right)_\Omega \\
- \;& \frac{1}{2}\left(w'_{,x}, \alpha^2 (u'^{k+1})^2 + 2\alpha\left((1-\alpha)u'^{k} + \langle\overline{u}\rangle_\alpha\right) u'^{k+1}\right)_\Omega \\
- \;& \frac{1}{2}\left(w'_{,x}, \left((1-\alpha)u'^{k} + \langle\overline{u}\rangle_\alpha\right)^2\right)_\Omega \\
= \;& \left(w', \overline{f}^{k+1/2}\right)_\Omega + \left(w', f'^{k+1/2}\right)_\Omega
\end{aligned} \tag{B.50}$$

Asymptotic expansion of the nonlinearities

If we decompose the space of the test functions, according to (14.11), and we assume $\overline{w}$ and w' linearly independent, we obtain a formula for the large scale equation:

$$\begin{aligned}
B_{LIN}(w', u'^{k+1}; \langle\overline{u}\rangle_\alpha, u'^{k}) + B_{NL}(w', u'^{k+1}, u'^{k}) \;&=\; < Res\,\overline{u}, w' >_{\mathcal{V}'^*, \mathcal{V}'} \\
&\quad + F(w'; \langle\overline{u}\rangle_\alpha, u'^{k})
\end{aligned} \tag{B.51}$$

where

$$\begin{aligned}
B_{LIN}(w', u'^{k+1}; \langle\overline{u}\rangle_\alpha, u'^{k}) \;&=\; \frac{1}{\Delta t}\left(w', u'^{k+1}\right)_\Omega + \frac{\alpha}{Re}\left(w'_{,x}, u'^{k+1}_{,x}\right)_\Omega \\
&\quad -\alpha(w'_{,x}, \langle\overline{u}\rangle_\alpha \; u'^{k+1})_\Omega
\end{aligned} \tag{B.52}$$

$$\begin{aligned}
B_{NL}(w', u'^{k+1}, u'^{k}) \;&=\; -\frac{1}{2}\left(w'_{,x}, \alpha^2\left(u'^{k+1}\right)^2 + 2\alpha(1-\alpha)u'^{k+1}u'^{k}\right)_\Omega \\
&\quad -\frac{1}{2}\left(w'_{,x}, (1-\alpha)^2 (u'^{k})^2\right)_\Omega
\end{aligned} \tag{B.53}$$

$$\begin{aligned}
F(w'; \langle\overline{u}\rangle_\alpha, u'^{k}) \;&=\; \frac{1}{\Delta t}(w', u'^{k})_\Omega - \frac{1}{Re}(w'_{,x}, (1-\alpha)u'^{k}_{,x})_\Omega \\
&\quad + \left(w'_{,x}, (1-\alpha)\langle\overline{u}\rangle_\alpha u'^{k}\right)_\Omega + \left(w', f'^{k+1/2}\right)_\Omega
\end{aligned} \tag{B.54}$$

$$\begin{aligned}
< Res\,\overline{u}, w' >_{\mathcal{V}'^*, \mathcal{V}'} \;&=\; \left(w', \overline{f}^{k+1/2}\right)_\Omega - \frac{1}{\Delta t}(w', [\overline{u}])_\Omega \\
&\quad -\frac{1}{Re}(w'_{,x}, < \overline{u}_{,x} >_\alpha)_\Omega + \left(w'_{,x}, \frac{\langle\overline{u}\rangle^2_\alpha}{2}\right)_\Omega
\end{aligned} \tag{B.55}$$

We are now ready to perform an asymptotic expansion of the nonlinear terms. Let us set as usual:

$$u' = \sum_{j=1}^{\infty} \varepsilon^j \tilde{u}_j \tag{B.56}$$

As it easily understood, there is no problem involved in the linear terms, since it trivially verified that, for a linear operator $\mathcal{L}$:

$$\mathcal{L}(\sum_{j=1}^{\infty} \varepsilon^j \tilde{u}_j) = \sum_{j=1}^{\infty} \varepsilon^j \mathcal{L}(\tilde{u}_j) \tag{B.57}$$

It is also assumed that $< Res\, \overline{u}, w' >_{\mathcal{V}'^*,\mathcal{V}'} = O(\varepsilon)$. The only term we have to take care of, is $B_{NL}(w', u'^{k+1}, u'^k)$. It is easily seen that if $a = \sum_{j=1}^{\infty} \varepsilon^j a_j$, and $b = \sum_{j=1}^{\infty} \varepsilon^j b_j$, then $ab = \sum_{n=2}^{\infty} \varepsilon^j \sum_{j=1}^{n-1} a_j b_{n-j}$. Therefore:

$$\begin{aligned} B_{NL}(w', \sum_{j=1}^{\infty} \varepsilon^j u_j^{k+1}, \sum_{j=1}^{\infty} \varepsilon^j u_j^k) &= -\frac{1}{2} \sum_{n=2}^{\infty} \varepsilon^n \sum_{j=1}^{n-1} \big(w', \alpha^2 \tilde{u}_j^{k+1} \tilde{u}_{n-j}^{k+1} \\ &\qquad +2\alpha(1-\alpha) \tilde{u}_j^{k+1} \tilde{u}_{n-j}^k \\ &\qquad +(1-\alpha)^2 \tilde{u}_j \tilde{u}_{n-j}^k)^2\big)_\Omega \\ &= -\sum_{n=2}^{\infty} \varepsilon^n F_{AE}(w', \{\tilde{u}_j^{k+1}\}_1^{n-1}) \end{aligned} \tag{B.58}$$

where we have set:

$$\begin{aligned} F_{AE}(w', \{\tilde{u}_j^{k+1}\}_1^{n-1}) &= \frac{1}{2} \sum_{j=1}^{n-1} \big(w', \alpha \tilde{u}_j^{k+1} < \tilde{u}_{n-j} >_\alpha \\ &\qquad +(1-\alpha) < \tilde{u}_j >_\alpha \tilde{u}_{n-j}^k\big)_\Omega \end{aligned} \tag{B.59}$$

We can summarize the asymptotic expansion procedure as follows (for convenience, we will set $u_j = \varepsilon^j \tilde{u}_j$, for $j = 1, 2, \ldots, \infty$):

$$\begin{aligned} \text{For } n = 1\text{: } B_{LIN}(w', u_1^{k+1}; \langle \overline{u} \rangle_\alpha, u_1^k) &= < Res\, \overline{u}, w' >_{\mathcal{V}'^*,\mathcal{V}'} \\ &\quad +F(w'; \langle \overline{u} \rangle_\alpha, u_1^k) \end{aligned} \tag{B.60}$$

$$\begin{aligned} \text{For } n \geq 2\text{: } B_{LIN}(w', u_n^{k+1}; \langle \overline{u} \rangle_\alpha, u_n^k) &= F(w'; \langle \overline{u} \rangle_\alpha, u_n^k) \\ &\quad +F_{AE}(w', \{u_j^{k+1}\}_1^{n-1}) \end{aligned} \tag{B.61}$$

Assembly of the small scale system

Recalling (B.32–B.35) and (B.40–B.43), we have:

$$\begin{aligned}
\text{For } j=1: \quad \boldsymbol{A}^{LIN}_{A'B'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha\right]\boldsymbol{U}^{k+1}_{1_{B'}} &= \boldsymbol{F}_{A'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha, \boldsymbol{U}^k_1\right] \\
&\quad +\boldsymbol{R}^{\overline{u}}_{A'} \qquad \text{(B.62)} \\
\text{For } n\geq 2: \quad \boldsymbol{A}^{LIN}_{A'B'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha\right]\boldsymbol{U}^{k+1}_{n_{B'}} &= \boldsymbol{F}_{A'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha, \boldsymbol{U}^k_n\right] \\
&\quad +\boldsymbol{F}^{AE}_{A'}\left[\{\boldsymbol{U}^{k+1}_{j_{B'}}\}^{n-1}_1\right] \qquad \text{(B.63)}
\end{aligned}$$

with

$$\begin{aligned}
\boldsymbol{A}^{LIN}_{A'B'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha\right] &= \left\{\boldsymbol{K}^{\Delta t}+\alpha\boldsymbol{K}^{Re}+\alpha\boldsymbol{K}^{\langle\overline{u}\rangle_\alpha}\right\}_{A'B'} \qquad \text{(B.64)} \\
\boldsymbol{R}^{\overline{u}}_{A'} &= \boldsymbol{M}_{A'\bar{B}}\overline{\boldsymbol{F}}^{k+1/2}_{\bar{B}} \\
&\quad -\left\{\boldsymbol{K}^{\Delta t}_{A'\bar{B}}[\overline{\boldsymbol{U}}_{\bar{B}}]+\left\{\boldsymbol{K}^{Re}+\frac{1}{2}\boldsymbol{K}^{\langle\overline{u}\rangle_\alpha}\right\}_{A'\bar{B}}\langle\overline{\boldsymbol{U}}_{\bar{B}}\rangle_\alpha\right\} \qquad \text{(B.65)} \\
\boldsymbol{F}_{A'}\left[\langle\overline{\boldsymbol{U}}\rangle_\alpha, \boldsymbol{U}^k_n\right] &= \left\{\boldsymbol{K}^{\Delta t}-(1-\alpha)\left\{\boldsymbol{K}^{Re}+\boldsymbol{K}^{\langle\overline{u}\rangle_\alpha}\right\}\right\}_{A'B'}\boldsymbol{U}^k_{n_{B'}} \\
&\quad +\boldsymbol{M}_{A'B'}\boldsymbol{F}^{k+1/2}_{n_{B'}} \qquad \text{(B.66)} \\
\boldsymbol{F}^{AE}_{A'}\left[\{\boldsymbol{U}^{k+1}_{j_{B'}}\}^{n-1}_1\right] &= -\frac{1}{2}\sum_{j=1}^{n-1}\left\{\alpha\boldsymbol{K}^{\langle u_{n-j}\rangle_\alpha}_{A'B'}\boldsymbol{U}^{k+1}_{j_{B'}}+(1-\alpha)\boldsymbol{K}^{\langle u_j\rangle_\alpha}_{A'B'}\boldsymbol{U}^k_{n-j_{B'}}\right\} \qquad \text{(B.67)}
\end{aligned}$$

A summary of the algorithm: multiscale Newton iteration

1. Do one iteration of the Newton solver for the Large Scale equation (assume initially $\boldsymbol{U}'^{\nu}=\boldsymbol{U}'^{k}$, since a guess for $\boldsymbol{U}'^{\nu+1}$ is not available).

2. Use $\overline{\boldsymbol{U}}^{\nu+1}$ to form $\langle\overline{\boldsymbol{U}}\rangle_\alpha$ to solve a cascade of linear systems for $\boldsymbol{U}^{\nu+1}_n$, $n=1,2,\ldots$.

3. Do another iteration of the Newton solver for the Large Scale equation (use $\boldsymbol{U}'^{\nu+1}=\sum_n\boldsymbol{U}^{\nu+1}_n$ computed in 2 to form $\langle\boldsymbol{U}'\rangle_\alpha$).

4. Iterate from 2 until $\|\delta^\nu\|=\dfrac{\|\overline{\boldsymbol{U}}^{\nu+1}-\overline{\boldsymbol{U}}^{\nu}\|}{\|\overline{\boldsymbol{U}}\|}\leq\beta$ (β is a predefined threshold).

B.4 Preliminary Testing for the unforced case

In Figure B.1, a comparison between exact solution and numerical solution without asymptotic expansion (large scales only) is shown for an initial condition given by a sine wave. In this case the Reynolds number is formed using the length $\tilde{L} = 1$ of the torus, the magnitude of the initial velocity $\tilde{u}_0$, and the viscosity ν, as $Re = \frac{\tilde{u}_0 \tilde{L}}{\nu}$. For this case 2048 elements were used, with $\Delta t = 0.001$ and $Re = 100$. The numerical solution overlaps the exact solution almost everywhere in the 1-periodic torus $\mathbb{T}$.

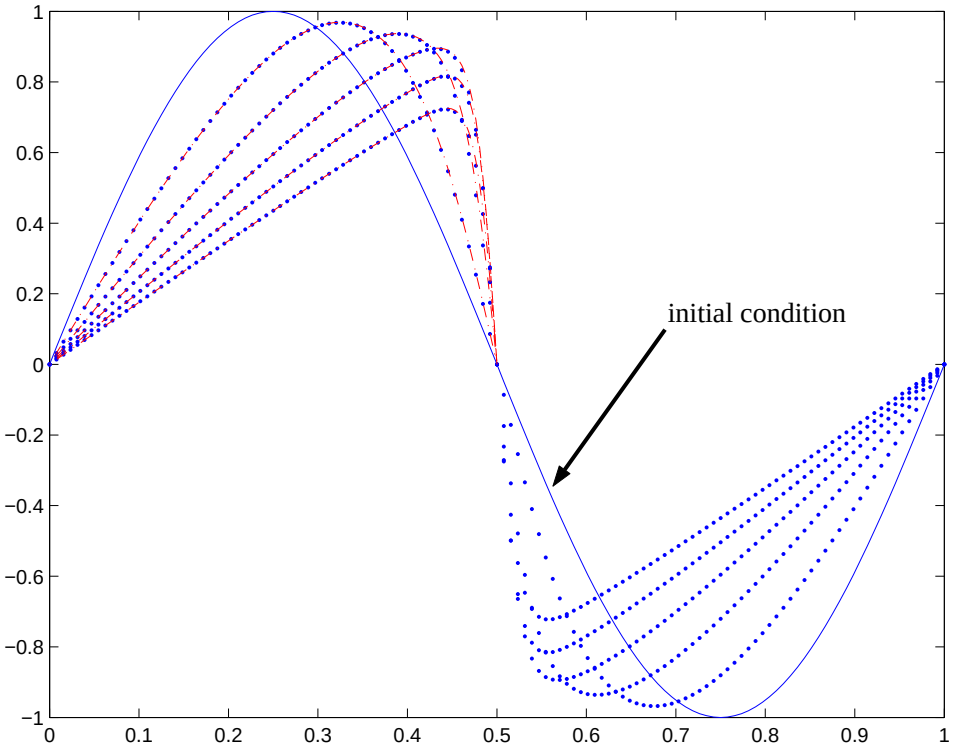

Figure B.1: $Re = 100$, 2048 elements, and $\Delta t = 0.001$. The initial solution is represented by the continuous blue line, the exact solution is represented by the dash/dotted red line, the numerical solution is represented by the blue dots.

B.5 Generation of random forcing.

Two techniques for the generation of a random forcing term in the stochastically forced Burgers equation will be presented and compared. As it will be clear from the following discussion, the two approaches, although different, lead to very similar numerical results.

B.5.1 Generation of the forcing according to T. Gotoh.

Following the approach by Gotoh [22, 23], a stochastic forcing which is white-in-time and with Gaussian Fourier spectrum can be generated (in wavenumber space) as follows:

1. For each wavenumber $\tilde{k}$, two Standard Normal $N(0, d\tilde{t})$ (i.e. Gaussian of mean zero and variance $d\tilde{t}$) Random Variables are generated: $a(\tilde{k})$ and $b(\tilde{k})$.

2. Recalling that $a(\tilde{k}), b(\tilde{k}) = N(0, d\tilde{t})$ (normal distributions), the spectrum is defined as

$$\begin{aligned} A(\tilde{k})^2 &= \tilde{k}^4 e^{-\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} && \text{(B.68)} \\ \tilde{F}(\tilde{k}) &= 2 < |A(\tilde{k})^2[a(\tilde{k}) + ib(\tilde{k})]|^2 > d\tilde{k} \\ &= 2A(\tilde{k})^2[< a(\tilde{k})^2 > + < b(\tilde{k})^2 >]d\tilde{k} \qquad (a(\tilde{k}), b(\tilde{k}) = N(0, d\tilde{t})) \\ &= 4A(\tilde{k})^2 d\tilde{k} d\tilde{t} && \text{(B.69)} \end{aligned}$$

   Here $d\tilde{k} = \frac{2\pi}{\tilde{l}} = \frac{1}{6}$, ($\tilde{l} = 12\pi$ in our case, is the length of the computational domain). The factor $d\tilde{t}$ comes into play because $\tilde{F}(\tilde{k})$ is the Fourier transform of the autocorrelation function (an ensemble average), or, by stationarity of the signal, the integral in time of the spectra. The 2 factor in the first row of (B.68) is due to the fact that to each wavenumber $\tilde{k}$ there corresponds a wavenumber $-\tilde{k}$, which contributes to the same wavelength.

3. The single Fourier coefficient of the spectrum is expressed as:

$$\hat{\tilde{f}} = C_1 \sqrt{\frac{\tilde{F}(\tilde{k})}{2d\tilde{t}d\tilde{k}}} \, [a(\tilde{k}) + ib(\tilde{k})] = \sqrt{2} C_1 A(\tilde{k})[a(\tilde{k}) + ib(\tilde{k})] \qquad \text{(B.70)}$$

   where C_1 is a normalization constant to be determined.

4. The spectrum is then normalized so that (B.3) is satisfied. If we introduce the normalization constant $B = \tilde{\xi}_f^3$, then

$$\begin{aligned} B = \tilde{\xi}_f^3 &= \sum_{\overline{k}=1}^{N/2-1} 2\tilde{k}^2 2C_1^2 A(\tilde{k})^2 |[a(\tilde{k}) + ib(\tilde{k})]|^2 d\tilde{k}^2 d\tilde{t} \\ &= \sum_{\overline{k}=1}^{N/2-1} 4C_1^2 \tilde{k}^2 A(\tilde{k})^2 [a(\tilde{k})^2 + b(\tilde{k})^2] d\tilde{k}^2 d\tilde{t} \end{aligned} \tag{B.71}$$

Here the sum is over the integer $\overline{k} = \tilde{k}\tilde{l}/(2\pi) \in \mathbb{N}$. Thus:

$$\begin{aligned} C_1 &= \sqrt{\frac{B}{\sum_{\overline{k}=1}^{N/2-1} 4\tilde{k}^2 A(\tilde{k})^2 [a(\tilde{k})^2 + b(\tilde{k})^2] d\tilde{k}^2 d\tilde{t}}} \\ &= \sqrt{\frac{B}{\sum_{\overline{k}=1}^{N/2-1} 4\tilde{k}^6 e^{-\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} [a(\tilde{k})^2 + b(\tilde{k})^2] d\tilde{k}^2 d\tilde{t}}} \end{aligned} \tag{B.72}$$

5. Using the fact that $d\tilde{t} = \frac{dt}{\tilde{\xi}_f}$, $k = \frac{\tilde{k}}{\tilde{k}_f}$, we finally have, for $\overline{k} = 1, 2, \ldots, N/2 - 1$:

$$\begin{aligned} \hat{\tilde{f}} &= \frac{\sqrt{2B}\tilde{k}^2 e^{-\frac{1}{2}\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} [a(\tilde{k}) + ib(\tilde{k})]}{\sqrt{\sum_{\overline{k}=1}^{N/2-1} 4\tilde{k}^6 e^{-\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} [a(\tilde{k})^2 + b(\tilde{k})^2] d\tilde{k}^2 d\tilde{t}}} \\ &= \frac{\tilde{\xi}_f^{\frac{3}{2}} \tilde{k}_f^2}{\sqrt{d\tilde{t}}\tilde{k}_f^4} \frac{\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} [a(\tilde{k}) + ib(\tilde{k})]}{\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^6 e^{-\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2} [a(\tilde{k})^2 + b(\tilde{k})^2] d\left(\frac{\tilde{k}}{\tilde{k}_f}\right)^2}} \\ &= \frac{\tilde{\xi}_f^{\frac{3}{2}} \tilde{k}_f^2}{\sqrt{1/\tilde{\xi}_f}\tilde{k}_f^4} \frac{k^2 e^{-\frac{k^2}{2}} [a(k) + ib(k)]}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2k^6 e^{-k^2} [a(k)^2 + b(k)^2] dk^2}} \\ &= \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2} \frac{k^2 e^{-\frac{k^2}{2}} [a(k) + ib(k)]}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2k^6 e^{-k^2} [a(k)^2 + b(k)^2] dk^2}} \\ &= \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2} \hat{f}(\overline{k}) \end{aligned} \tag{B.73}$$

Therefore, for $\overline{k} = 1, 2, \ldots, N/2 - 1$:

$$\hat{f}_{\text{Gotoh}} = \frac{\left(\frac{\overline{k}}{\overline{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} [a(\overline{k}) + ib(\overline{k})]}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} \frac{2}{\overline{k}_f^2} \left(\frac{\overline{k}}{\overline{k}_f}\right)^6 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} [a(\overline{k})^2 + b(\overline{k})^2]}} \tag{B.74}$$

where the expression for $\hat{f}$ is non-dimensional. Here we have used the definition $\overline{k}_f = \tilde{l}/(2\pi)\tilde{k}_f$, so that $\frac{\tilde{k}}{\tilde{k}_f} = \frac{\overline{k}}{\overline{k}_f}$, $d\overline{k} = 1$, and $d\left(\frac{\overline{k}}{\overline{k}_f}\right) = \frac{1}{\overline{k}_f}$. In the simulations we have set $\overline{k}_f = 6$. Notice that we have obtained $\hat{\tilde{f}} \sim \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2}$, which yields $\tilde{f}_0 \sim \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2}\tilde{k}_f = \frac{\tilde{\xi}_f^2}{\tilde{k}_f}$. In particular we readily see that $G = \frac{\tilde{f}_0 \tilde{L}}{\tilde{U}^2} = \frac{\tilde{f}_0 \tilde{k}_f}{\tilde{\xi}_f^2} = 1$, as expected in (B.12).

B.5.2 Generation of the forcing according to S.B. Pope.

In the approach proposed by S.B. Pope in [56, 57], the normalization is slightly different: $\hat{\tilde{f}}$ is normalized by the sum of the squares of the Fourier coefficients, and involves an extra factor 0.4. Also, the forcing is not Gaussian, but instead the coefficients $a(\overline{k})$ and $b(\overline{k})$ are obtained from a random phase uniformly distributed in the interval $[0, 2\pi)$.

Here are the explicit steps:

1. For each k, a Uniform Random Variable $\theta = 2\pi U([0, 1))$ is generated.

2. The spectrum is defined as

$$A(\overline{k})^2 = \left(\frac{\overline{k}}{\overline{k}_f}\right)^4 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} \tag{B.75}$$

$$F(\overline{k}) = A(\overline{k})^2 \tag{B.76}$$

   Notice that there is no factor 4 in front of the RHS in (B.76).

3. The single Fourier coefficient of the spectrum is expressed as:

$$\hat{\tilde{f}} = C_2 C_1 \sqrt{F(\overline{k})}\, e^{i\theta} = C_2 C_1 A(\overline{k}) e^{i\theta} \tag{B.77}$$

   where C_1 and C_2 are normalization constant to be determined.

4. The spectrum is then normalized imposing a unitary sum of the squares of the Fourier coefficients:

$$C_1 = \frac{1}{\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2A(\overline{k})^2 \frac{1}{\overline{k}_f^2}}} = \frac{1}{\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2\left(\frac{\overline{k}}{\overline{k}_f}\right)^4 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} \frac{1}{\overline{k}_f^2}}} \tag{B.78}$$

5. The final step involves multiplication by an appropriate dimensional constant C_2 that makes the forcing dimensionally consistent. Setting

$$C_2 = \sqrt{\frac{0.4B}{\tilde{k}_f^2 d\tilde{t}}} = \frac{\sqrt{0.4}\tilde{\xi}_f^{\frac{3}{2}}}{\tilde{k}_f^2\sqrt{dt}\tilde{\xi}_f^{-\frac{1}{2}}} = \frac{\sqrt{0.4}}{\sqrt{dt}}\frac{\tilde{\xi}_f^2}{\tilde{k}_f^2} \tag{B.79}$$

and finally

$$\hat{\tilde{f}} = C_2 C_1 A(\overline{k}) e^{i\theta} = \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2}\hat{f} \tag{B.80}$$

with, for $\overline{k} = 1, 2, \ldots, N/2-1$

$$\hat{f}_{\text{Pope}} = \frac{\sqrt{0.4}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} e^{i\theta}}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2\left(\frac{\overline{k}}{\overline{k}_f}\right)^4 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} \frac{1}{\overline{k}_f^2}}} \quad , \tag{B.81}$$

where the expression for $\hat{f}$ is non-dimensional. Once more, we have obtained $\hat{\tilde{f}} \sim \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2}$, which yields $\tilde{f}_0 \sim \frac{\tilde{\xi}_f^2}{\tilde{k}_f^2}\tilde{k}_f = \frac{\tilde{\xi}_f^2}{\tilde{k}_f}$, and $G = \frac{\tilde{f}_0 \tilde{L}}{\tilde{U}^2} = \frac{\tilde{f}_0 \tilde{k}_f}{\tilde{\xi}_f^2} = 1$.

The Pope forcing is closely related to the Gotoh's forcing, in fact:

$$\int_0^\infty x^4 e^{-x^2} dx = 0.4 \int_0^\infty x^6 e^{-x^2} dx \tag{B.82}$$

so that, for $\overline{k} = 1, 2, \ldots, N/2-1$, (B.81) can also be written as:

$$\hat{f}_{\text{Pope}} = \frac{\left(\frac{\overline{k}}{\overline{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} e^{i\theta}}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} 2\left(\frac{\overline{k}}{\overline{k}_f}\right)^6 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} \frac{1}{\overline{k}_f^2}}} \tag{B.83}$$

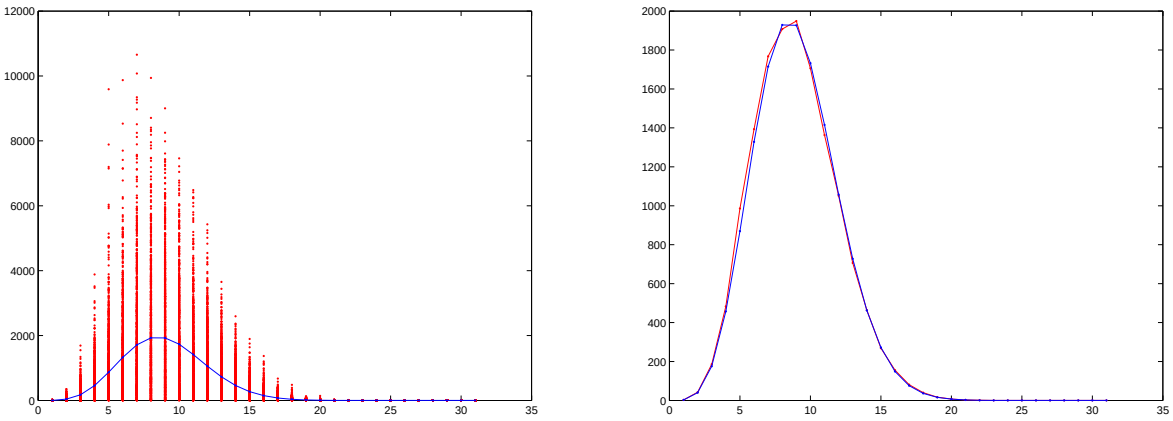

Figure B.2: $Re = 1000$, $N = 1024$, $\Delta t = 0.001$. For the two pictures, red refers to Gotoh's implementation, blue refers to Pope's implementation. Left: 1000 realizations of the spectrum for Gotoh's forcing are compared with Pope's forcing. Right: The average of the Gotoh's spectrum over the previous 1000 realizations is compared to Pope's spectrum. As we can see, the average of the Gotoh's forcing converges to Pope's forcing, but instantaneous behaviors may be very different.

B.5.3 Comparison of forcing techniques

The non dimensional Burgers' equation reads then:

$$u_{,t} + \left(\frac{u^2}{2}\right)_{,x} = \frac{1}{Re} u_{,xx} + f \tag{B.84}$$

where, for $\overline{k} = 1, 2, \ldots, N/2 - 1$,

$$\hat{f}_{\text{Gotoh}} = \frac{\left(\frac{\overline{k}}{\overline{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} [a(\overline{k}) + ib(\overline{k})]}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} \frac{2}{\overline{k}_f^2} \left(\frac{\overline{k}}{\overline{k}_f}\right)^6 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} [a(\overline{k})^2 + b(\overline{k})^2]}} \tag{B.85}$$

$$\hat{f}_{\text{Pope}} = \frac{\left(\frac{\overline{k}}{\overline{k}_f}\right)^2 e^{-\frac{1}{2}\left(\frac{\overline{k}}{\overline{k}_f}\right)^2} e^{i\theta}}{\sqrt{dt}\sqrt{\sum_{\overline{k}=1}^{N/2-1} \frac{2}{\overline{k}_f^2} \left(\frac{\overline{k}}{\overline{k}_f}\right)^6 e^{-\left(\frac{\overline{k}}{\overline{k}_f}\right)^2}}} \tag{B.86}$$

The role of dt is due to the stochastic nature of the forcing, which makes the Burgers' ODE an SDE, i.e. a Stochastic Differential Equation, and the typical time scale of the forcing is correctly $1/\sqrt{dt}$. As we can see, the forcing terms have a very different nature, the forcing by Gotoh follows a Gaussian distribution, and computes the value of the

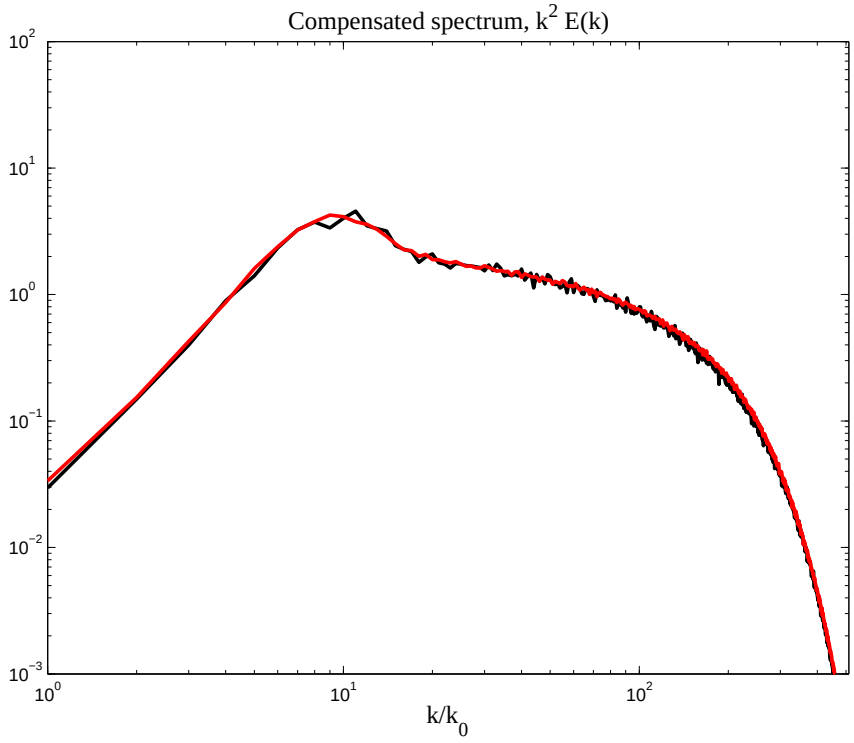

Figure B.3: $Re = 1000$, 1024 elements, $\Delta t = 0.001$. Red refers to Gotoh's implementation of the forcing (duration $T_s = 50$), black refers to Pope's implementation (duration $T_s = 200$).

Fourier coefficients of the forcing directly using the random generator, which implies that, occasionally, the values of some coefficient may be very high or very low (while mean and variance are unaffected). On the other hand, the forcing by Pope, prescribes the amplitude of each coefficient, and randomness affects the phase content only. As we can see in Figure B.2, the two forcing terms seem to have the same effect on average, but the instant realizations are substantially different.

Comparing the effect on the statistics of the solution u in Figure B.3, very little difference is found in the spectrum of u (and therefore the variance of the solution in time). The Gotoh's forcing produces a smoother energy spectrum, even though the solution in the case of Pope's forcing is sampled for a duration four times longer.

Bibliography

[1] D. N. Arnold, F. Brezzi, B. Cockburn, and L.D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, **39**(5):1749–1779, 2002.

[2] A. K. Aziz and P. Monk. Continuous finite elements in space and time for the heat equation. *Mathematics of Computation*, **144**:70–97, 1998.

[3] D. J. Benson. Computational methods in Lagrangian and Eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, **99**, 1992.

[4] F. Brezzi, G. Manzini, D. Marini, P. Pietra, and A. Russo. Discontinuous Galerkin approximations for elliptic problems. *Numerical Methods for Partial Differential Equations*, **16**(4):365–378, 2000.

[5] F. Brezzi and L.D. Marini. A nonconforming element for the Reissner-Mindlin plate. *Computers and Structures*, **81**(8-11):515–522, 2003.

[6] Dawson C. The P^{k+1}-S^k local discontinuous Galerkin method for elliptic equations. *SIAM Journal on Numerical Analysis*, **40**(6):2151–2170, 2003.

[7] V.M. Calo. *A new multiscale paradigm for les: residual-based turbulence modeling. Finite volume simulations of bypass transition.* PhD thesis, Civil and Environmental Engineering Department, Stanford Universty, 2004.

[8] E. J. Caramana, M. J. Shashkov, and P. P. Whalen. Formulations of artificial viscosity for multi-dimensional shock wave computations. *Journal of Computational Physics*, **144**:70–97, 1998.

[9] R. B. Christiansen. Godunov methods on a staggered mesh – an improved artificial viscosity. Technical report, UCLR-JC-105269, Lawrence Livermore National Laboratory, 1991.

[10] P. G. Ciarlet. *Mathematical Elasticity: Volume I: Three-Dimensional Elasticity*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[11] B. Cockburn, G.E. Karniadakis, and C.-W. Shu (eds.). *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Lecture Notes in Computational Science and Engineering, 11. Springer-Verlag, 2000.

[12] S. S. Collis. The DG/VMS method for unified turbulence simulation. AIAA 2002-3124. AIAA 32nd Fluid Dynamics Conference, St. Louis, Missouri, June 24-27, 2002.

[13] G. Engel, K. Garikipati, T.J.R. Hughes, M.G. Larson, L. Mazzei, and R.L. Taylor. Continuous/discontinuous finite element approximations of fourth-order elliptic problems in structural and continuum mechanics with applications to thin beams and plates, and strain gradient elasticity. *Computer Methods in Applied Mechanics and Engineering*, **191**(34):3669–3750, 2002.

[14] D. Estep and D. A. French. Global error control for the continuous Galerkin finite element method for ordinary differential equations. *RAIRO*, **28**:815–852, 1994.

[15] C. Farhat and B. Koobus. Finite volume discretization on unstructured meshes of the multiscale formulation of large eddy simulations. In *Proceedings of the Fifth World Congress on Computational Mechanics (WCCM V)*, Vienna University of Technology, Austria, July 7-12, 2002.

[16] L. P. Franca. An algorithm to compute the square root of a 3×3 positive definite matrix. *Computers and Mathematics with Applications*, **18**:459–466, 1989.

[17] D. A. French. A space-time finite elements method for the wave equation. *Computer Methods in Applied Mechanics and Engineering*, **107**:145–157, 1993.

[18] D. A. French. Continuous Galerkin finite element methods for a forward-backward heat equation. *Numerical Methods for Partial Differential Equations*, **15**:491–506, 1999.

[19] D. A. French and S. Jensen. Long time behaviour of arbitrary order continuous time Galerkin schemes for some one-dimensional phase transition problems. *IMA Journal on Numerical Analysis*, **14**:421–442, 1994.

[20] D. A. French and T. E. Peterson. A continuous space-time finite element method for the wave equation. *Mathematics of Computation*, **65**:491–506, 1996.

[21] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot. A dynamic subgrid-scale model. *Physics of Fluids A*, **3**:1760–1765, 1991.

[22] T. Gotoh. Probability density functions in steady-state Burgers turbulence. *Physics of Fluids*, 11(8):2143–2148, 1999.

[23] T. Gotoh. Private communication, January 2003.

[24] G. Hauke and T. J. R. Hughes. A unified approach to compressible and incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, **113**:389–396, 1994.

[25] G. Hauke and T. J. R. Hughes. A comparative study of different sets of variables for solving compressible and incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, **153**:1–44, 1998.

[26] J. Holmen, T.J.R. Hughes, A.A.Oberai, and G.N.Wells. Sensitivity of the scale partition for variational multiscale les of channele flow. *Physics of Fluids*, **16**(3):824–827, 2004.

[27] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[28] T. J. R. Hughes, L. Mazzei, and K. E. Jansen. Large eddy simulation and the variational multiscale method. *Computing and Visualization in Science*, **3**(47):147–162, 2000.

[29] T. J. R. Hughes, L. Mazzei, A. A. Oberai, and A.A. Wray. The multiscale formulation of large eddy simulation: decay of homogenous isotropic turbulence. *Physics of Fluids*, **13**:505–512, 2001.

[30] T. J. R. Hughes, A. A. Oberai, and L. Mazzei. Large eddy simulation of turbulent channel flows by the variational multiscale method. *Physics of Fluids*, **13**(6):1784–1799, 2001.

[31] T. J. R. Hughes, G. N. Wells, and A. A. Wray. Energy transfers and spectral eddy viscosity in large eddy simulations of homogeneous isotropic turbulence. Technical Report, Institute for Computational Engineering and Sciences, University of Texas at Austin, 2004. Available at `http://www.utexas.edu/reports/2004/0420.pdf`. To appear in *Physics of Fluids*.

[32] T.J.R. Hughes. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid-scale models, bubbles and the origin of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, **127**:387–401, 1995.

[33] T.J.R. Hughes, G. Engel, L. Mazzei, and M.G. Larson. A comparison of discontinuous and continuous Galerkin methods based on error estimates, conservation, robustness and efficiency. In B. Cockburn, G.E. Karniadakis, and C.-W. Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Springer-Verlag, 2000.

[34] T.J.R. Hughes, G.R. Feijóo, L. Mazzei, and J.-B. Quincy. The Variational Multiscale Method — a paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, **166**:3–24, 1998.

[35] T.J.R. Hughes, H.M. Hilber, and R.L. Taylor. Implicit-explicit finite elements in nonlinear transient analysis. *Computer Methods in Applied Mechanics and Engineering*, **17-18**:159–182, 1979.

[36] T.J.R. Hughes, A. Masud, and J. Wan. A stabilized mixed discontinuous Galerkin method for Darcy flow. (in preparation), 2004.

[37] T.J.R. Hughes, G. Scovazzi, and Franca L.P. Multiscale and stabilized methods. In R. Stein E., de Borst and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*. John Wiley & Sons, 2004.

[38] B. L. Hulme. Discrete Galerkin and related one-step methods for ordinary differential equations. *Mathematics of Computation*, **26**-120:881–891, 1972.

[39] J.C.R. Hunt and D.J.B. Carruthers. Rapid distortion theory and the problems of turbulence. *Journal of Fluid Mechanics*, **212**:497–532, 1990.

[40] R. G. Jacobs and P. A. Durbin. Bypass transition phenomena studied by computer simulation. Technical report, TF-77, Flow Physics and Computation Division, Department of Mechanical Engineering, Stanford University, 2000.

[41] R. G. Jacobs and P. A. Durbin. Simulations of bypass transition. *Journal of Fluid Mechanics*, **212**:428–185, 2001.

[42] P. Jamet. Stability and convergence of a generalized Crank-Nicolson scheme on a variable mesh for the heat equation. *SIAM Journal on Numerical Analysis*, **17**(4):530–539, 1980.

[43] H. Jeanmart and G.S. Winckelmans. Comparison of recent dynamic subgrid-scale models in the case of the turbulent channel flow. In *Proceedings Summer Program 2002*, Center for Turbulence Research, NASA Ames / Stanford University, 2002.

[44] C. Johnson, U. Nävert, and J. Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, **45**:285–312, 1984.

[45] O. Klaas, A. Miniatty, and M.S. Shepard. A stabilized mixed finite element method for finite elasticity. Formulation for linear displacement and pressure interpolation. *Computer Methods in Applied Mechanics and Engineering*, **180**:65–79, 1999.

[46] B. Koobus and C. Farhat. Finite volume/element discretization on unstructured meshes of the multiscale formulation of the large eddy simulation method and application to vortex shedding. *Computer Methods in Applied Mechanics and Engineering*, **193**(15-16):: 1367–1383, 2004.

[47] V.F. Kuropatenko. On difference methods for the equations of hydrodynamics. In N.N. Janenko, editor, *Difference methods for solutions of problems of mathematical physics, I.* American Mathematical Society, Providence, R.I., 1967.

[48] P. Lesaint and P. A. Raviart. On a finite element method for solving the neutron transport equation. In de Boor C., editor, *Mathematical Aspects of Finite Elements in Partial Differential Equations*. Academic Press, New York, 1974.

[49] R. Liska and B. Wendroff. Comparison of several difference schemes on 1D and 2D test problems for the Euler equations. *SIAM Journal on Scientific Computing*, **25**(3):995–1017, 2003.

[50] A. Miniatty and Y. Liu. Stabilized finite element method for viscoplastic flow: formulation with state variable evolution. *International Journal of Numerical Methods in Engineering*, **56**:185–209, 2003.

[51] A. Miniatty, Y. Liu, O. Klaas, and M.S. Shepard. Stabilized finite element method for viscoplastic flow: formulation and a simple progressive solution strategy. *Computer Methods in Applied Mechanics and Engineering*, **190**:4609–4625, 2001.

[52] A. Miniatty, Y. Liu, O. Klaas, and M.S. Shepard. Higher order stabilized finite element method for hyperelastic finite deformation. *Computer Methods in Applied Mechanics and Engineering*, **191**:1491–1503, 2002.

[53] W. F. Noh. Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux. *Journal of Computational Physics*, **72**:78–120, 1987.

[54] A. A. Oberai and T. J. R. Hughes. The variational multiscale formulation of LES: Channel flow at $Re_\tau = 590$. 40th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 14-17, 2002.

[55] C. D. Pierce and P. Moin. Progress-variable approach for large eddy simulation of turbulent combustion. Technical report, TF-80, Flow Physics and Computation Division, Department of Mechanical Engineering, Stanford University, 2001. Available at `http://ctr.stanford.edu/Pierce/thesis.pdf`.

[56] S. B. Pope. Large-eddy simulation using projection onto local basis functions. In J. L. Lumley, editor, *Fluid Mechanics and the Environment: Dynamical Approaches*. Springer, 2000.

[57] S.B. Pope. Private communication, January 2003.

[58] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer-Verlag, Berlin, 1994.

[59] S. Ramakrishnan and S. S. Collis. Turbulence control simulation using the variational multiscale method. *AIAA Journal*, **42**(4), 2004.

[60] S. Ramakrishnan and S. S. Collis. Variational multiscale modeling for turbulence control. AIAA 2002-3280. AIAA 1st Flow Control Conference, St. Louis, Missouri, June 24-26, 2002.

[61] W.H. Reed and T.R. Hill. Triangular mesh methods for the neutron transport equation. Technical report, LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[62] P.E. Roach and D.H. Brierley. The influence of a turbulent free-stream on a zero-pressure gradient transitional boundary layer development, part I: Test cases T3A and T3B. In *Numerical Simulation of Unsteady Flows and Transition to Turbulence*. ERCOFTAC, 1992.

[63] F. Shakib and T. J. R. Hughes. A new finite element formulation for computational fluid dynamics: IX. Fourier analysis of space-time Galerkin/least-squares algorithms. *Computer Methods in Applied Mechanics and Engineering*, **87**:35–58, 1991.

[64] F. Shakib, T. J. R. Hughes, and Z. Johan. A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, **89**:141–219, 1991.

[65] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer-Verlag, Berlin-Heidelberg, 2nd edition, 1999.

[66] J. Von Neumann and R.D. Richtmyer. A method for the numerical computation of hydrodynamic shocks. *Journal of Applied Physics*, **21**:232–237, 1950.

[67] M.L. Wilkins. Use of artificial viscosity in multidimensional shock wave problems. *Journal of Computational Physics*, **36**:281–303, 1979.

[68] G.S. Winckelmans, H. Jeanmart, and D. Carati. On the comparison of turbulence intensities from large-eddy simulation with those from experiment or direct numerical simulation. *Physics of Fluids*, **14**(5), 2002.

[69] A.A. Wray. Private communication, 2002.