A Discontinuous Galerkin Method for Diffusion Flames Embedded in a Low-Mach Solver Framework

Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
Vorgelegte Dissertation von Juan Francisco Gutiérrez Jorquera aus Santiago de Chile
Tag der Einreichung: 23. November 2022, Tag der Prüfung: 23. November 2022

Erstreferent: Prof. Dr.-Ing Martin Oberlack
Koreferent: Prof. Dr. rer. nat. habil. Amsini Sadiki
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Maschinenbau
Fachgebiet für Strömungsdynamik

A Discontinuous Galerkin Method for Diffusion Flames Embedded in a Low-Mach Solver Framework

Vorgelegte Dissertation von Juan Francisco Gutiérrez Jorquera

Tag der Einreichung: 23. November 2022
Tag der Prüfung: 23. November 2022

Darmstadt

Bitte zitieren Sie dieses Dokument als:
URN: urn:nbn:de:tuda-tuprints-1234
URL: http://tuprints.ulb.tu-darmstadt.de/12345
DOI: https://doi.org/10.25534/tuprints-1234

Dieses Dokument wird bereitgestellt von tuprints,
E-Publishing-Service der TU Darmstadt
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de

Dedication

Erklärungen laut Promotionsordnung

§ 8 Abs. 1 lit. c PromO

Ich versichere hiermit, dass die elektronische Version meiner Dissertation mit der schriftlichen Version übereinstimmt.

§ 8 Abs. 1 lit. d PromO

Ich versichere hiermit, dass zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde. In diesem Fall sind nähere Angaben über Zeitpunkt, Hochschule, Dissertationsthema und Ergebnis dieses Versuchs mitzuteilen.

§ 9 Abs. 1 PromO

Ich versichere hiermit, dass die vorliegende Dissertation selbstständig und nur unter Verwendung der angegebenen Quellen verfasst wurde.

§ 9 Abs. 2 PromO

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 23. November 2022

J. Gutiérrez-Jorquera

Abstract

Abstract

Zusammenfassung

Zusammenfassung

Acknowledgements

Contents

List of Figures

List of Tables

List of Abbreviations

BoSSS	Bounded Support Spectral Solver
DOFs	degrees of freedom
ENO	Essentially Non-Oscillatory methods
SIMPLE	Semi-Implicit Method for Pressure Linked Equation
TVM	Total Variation diminishing Method
XNSEC	eXtended Navier-Stokes Equations solver for Combustion

List of Symbols

Ω	Domain of interest, computational domain
$\mathfrak{I}$	Interface
$\mathbf{n}_{\mathfrak{I}}$	Interface normal vector
$\mathbf{n}$	Normal vector
$\mathcal{B}^1$	Boundary condition form of the continuity equation.
$\mathcal{B}^2$	Boundary condition forms of the momentum equations.
$\mathcal{B}^3$	Boundary condition form of the temperature equation.
$\mathcal{B}^4$	Boundary condition forms of the mass fraction equations.
$\mathbf{f}$	Flux vector
μ	Dynamic viscosity
ν	Kinematic viscosity
c_p	Specific heat capacity of the mixture
$c_{p,k}$	Specific heat capacity of species k
T	Temperature
ρ	Density
D	Diffusion coefficient
ν_k	Stoichiometric coefficient of species k
N_k	Number of local DOF
N	Number of total DOF
$\mathbf{Op}$	Operator matrix
k	Total degree of the polynomial space
ϕ	Basis function
$\mathbb{P}_k$	Broken polynomial space
K	Numerical cell
Γ	Edge
h	Numerical mesh size
$\mathfrak{K}_h$	Numerical mesh
$\mathbf{M}$	Mass matrix
$\mathbf{n}_\Gamma$	Edge normal field
$\hat{f}$	Numerical flux
ϑ	General test function
ER	Expansion ratio the backward-facing step
Fr	Froude number

$\mathbf{g}$	Gravity vector
h	Channel heigth of the backward-facing step
$\hat{h}$	Convective heat transfer coefficient
λ	Heat conductivity
h_k	Specific enthalpy of k species.
Y_k	Mass fraction of species k
W	Mean molecular weight of the mixture
W_k	Molecular weight of species k
N	Total number of species in the mixture
$\mathbf{Nu}$	Nusselt number
$\mathbf{Nu_{loc}}$	Nusselt number, local
ω_k	Net rate of production of species k
$\mathbf{Pr}$	Pradntl number
p	Pressure
$\mathcal{R}$	Universal gas constant
R	Residual from DG approximation
r	Error of the DG discretization
$\mathbf{Re}$	Reynolds number
S	Step heigth of the backward-facing step
$\boldsymbol{\tau}$	Viscous tensor
t	Time
u	General variable of the DG-method
$\mathbf{u}$	Velocity vector
D	Diffusion coefficient
$\mathbf{U}$	Diffusion velocity vector

1 Introduction

ch:introduction

1.1 Introduction and state of the art.

1.1.1 Combustion

Premixed Flames(???)

Non-Premixed Flames

Droplet(?)

High order discretization methods is a topic which has been gaining increasing attention in the last decades. An important exponent of them is the Discontinuous Galerkin (DG) method Cockburn et al. (2000). The DG method was initially developed and utilized for solving hyperbolic conservation laws, especially in the field of computational fluid dynamics (CFD), and has recently gained increased attention for incompressible CFD problems for structured and unstructured grids. Two main advantages stand out when compared with traditional methods such as the Finite Volume Method (FVM) or the Finite Difference Method (FDM): First, DG offers an arbitrary order of error convergence due to the polynomial local approximation of the solution field. A polynomial approximation of degree p provides a numerical discretization error of the order $\mathcal{O}(h^{p+1})$ for sufficiently smooth solutions, where h is a characteristic grid length. Secondly, regardless of the desired order of accuracy, any given cell of the grid only requires information from its immediate neighbours, allowing for efficient parallelization with minimal communication overhead. In contrast, more traditional schemes, such as the FVM, are usually limited to $\mathcal{O}(h^N)$ accuracy, with $N \leq 2$ for unstructured grids. Even for structured grids N is practically limited to low values due to the increasing stencil size for increasing N. Advantageously the DG method offers the locality of low-order schemes and the accuracy per degree of freedom of spectral schemes.

In the context of CFD solvers, an important distinction is that between pressure-based and density-based solvers. Historically, pressure-based solvers have been used for incompressible flows, i.e. flows with divergence-free velocity fields. On the other hand, density-based (also called fully compressible) solvers should, in theory, be able to solve flows in all Mach number ranges. However, in practice, as the Mach number approaches zero, density-based solvers experience efficiency and accuracy problems. These issues are mainly attributed to the acoustic effects in the flow, which tends to generate very stiff systems. None of these approaches are directly applicable to flows with varying density in the low-Mach limit. Hennink et al. (2021) It is possible, however, to extend existing incompressible and compressible codes so that they are capable of dealing with low Mach numbers. Keshtiban et al. (2003) In this work a extension from a incompressible solver is presented.

There are numerous works in which the DG method has been used within the context of incompressible flows. Shahbazi et al. (2007), Kummer (2012), Klein et al. (2013), and Rhebergen et al. (2013) However, there are not many publications in which the flow problem at low Mach numbers is addressed using a pressure-based solver. In the work by Klein et al. Klein et al. (2016) the low-Mach equations are solved in a DG Framework making use of a SIMPLE type scheme. The solution of a time-step involves an iterative process that requires multiple matrix assemblies and solutions. The obtained systems of equations are solved by means of fixed-point iterations, where relaxation factors are necessary to obtain convergence of the computations. In the work of Hennink et al. Hennink et al. (2021) a pressure-based solver for low-mach flows is presented. They solve the mass flux instead of the velocity as the primitive variable.

High order methods are very attractive for complex reacting fluid dynamical systems, where usually a high numerical resolution is necessary. Particularly for combustion problems, where large amounts of heat are released in rather small zones within the flow, the high number of elements required to resolve the resulting steep gradients could result in prohibitive calculation times even for simple problems. In particular the study of so-called diffusion flames – also known as non-premixed flames – requires special consideration. In a diffusion flame, the reactants are initially spatially separated. For this kind of system, mixing plays a crucial role because reactants need to be brought together to the flame zone in order to maintain combustion. Many practical applications of diffusion flames consider deflagration flames, Poinsot and Veynante (2005) which are characterized by a small characteristic velocity compared to the speed of sound. The low-Mach approximation of the Navier–Stokes equations is often chosen for describing this kind of system. This approximation allows for the calculation of non-constant density flows (such as temperature dependent density), while neglecting acoustic effects, thus dramatically reducing the required temporal resolution. Müller (1998)

In addition to the compressibility effects mentioned above, the need to accurately and efficiently represent the chemical reactions governing the combustion problem poses a major challenge. Generally speaking, to study the combustion process a detailed chemistry description is preferable. However, this is often impractical as it can be very intensive computationally speaking. Stauch et al. investigated systems with detailed mechanism for methanol combustion, where 23 chemical species and 166 elementary reactions are involved, Stauch et al. (2006) and for n-heptane with 62 chemical species and 572 elementary reactions, and with detailed transport processesStauch and Maas (2007). Because of their high complexity the mentioned works are restricted to simple one- or two-dimensional configurations, and to a small number of grid elements. If one is interested in more complex geometries or more complicated flow systems, the use of detailed kinetics can be prohibitive. Simplified kinetic models have been developed to overcome this difficulty. In the work of Westbrook et al. Westbrook and Dryer (1981) a one-step kinetic model is presented, where combustion is expressed as a single chemical reaction with a reaction rate given by an Arrhenius-type expression with constant parameters. Multi-step chemical reaction models have also been developed, such as the four-step mechanism for methane combustion by Peters, Peters (1985) or the three-step mechanism by Peters and Williams. Peters and Williams (1987) Furthermore, extensions of one-step models exist, such as the one presented by Fernandez-Tarrazo et al. Fernandez-Tarrazo et al. (2006) for hydrocarbon combustion with air, where kinetic parameters are correlated to the equivalence ratio in order to better describe characteristic flame properties for premixed and non-premixed flames.

In the last decades several numerical investigations related to diffusion flames have been carried out. Burke and Schumann were the first ones to investigate the structure of diffusion flames by studying the flame jet problem. Burke and Schumann (1928) By assuming an infinitely fast chemical reaction they managed to predict flame properties fairly accurately. In the work of Smooke et al. Smooke et al. (1986) a numerical simulation for a two-dimensional axisymmetric laminar diffusion jet with detailed chemistry was conducted and solved with Newton-type methods. In order to obtain adequate initial estimates for this problem, the solution of the problem for an infinitely fast chemistry was solved first. This idea was used in several works such as that by Keyes and Smooke Keyes and Smooke (1987) for a counter diffusion flame, by Smooke Smooke and Giovangigli (1992) for a Tsuji-counterflow configuration and in the work by Dobbins et al. Dobbins and Smooke (2010) for an axisymmetric laminar jet diffusion flame with time dependent boundary conditions. In the work of Paxion Paxion et al. (2001) unstructured multigrid solver for laminar flames with detailed chemistry is presented. A Krylov-Newton method was used for solving several flame configurations. A two-dimensional counter diffusion flame was calculated, and its results were compared with the one-dimensional self-similar solution of the equations.

The DG method has also been used for simulations of combustion, mainly within a fully compressible framework. In the work from Johnson et al. Johnson and Kercher (2020) the compressible Navier–Stokes equations are solved using a nodal DG scheme for combustion with complex chemistry and transport parameters. An hp-adaptive method is also presented, and shown to be useful for solving the ordinary differential equations used for describing the unsteady behaviour of the system. Similarly in the work from Lv and Ihme Lv and Ihme (2017) a DG solver for multi-component chemically reacting flows, which solves the fully compressible Navier-Stokes equation, is presented. A hybrid-flux formulation is used, where a conservative Riemann solver is used for shock treatment, and a double-flux formulation is used in smooth regions. They show its applicability for non-reacting and reacting flows, particularly for systems characterized by high Mach numbers. On the other hand, the solution of combustion problems using the DG method in an incompressible framework is a topic that has not received much attention in the literature. This fact is the motivation of the present paper.

The system of equations obtained from the discretization of highly nonlinear systems can be very difficult to solve. Fixed-point iteration schemes have been used as a linearization strategy, as done by Klein et al. Klein et al. (2016) A major drawback of this approach is the highly problem dependent choice of under-relaxation factors. A much more robust strategy is the use of Newton type methods. Shadid et al. (1997) and Pawlowski et al. (2006) Here, a problem-dependent factor is not needed. There is however a trade-off, because the Jacobian matrix has to be calculated, which can be computationally expensive. This approach has been used for combustion problems in numerous works. Karaa et al. Karaa et al. (2003) studied the axisymmetric laminar jet diffusion flame and investigated the behaviour of a multi-grid solver when using different pre-conditioners with a damped Newton algorithm. Shen et al. Shen et al. (2006) investigated the use and efficiency of a Newton method coupled with the Bi-CGSTAB method for an axisymmetric laminar jet flame. They concluded that, in terms of computational cost, the steady-state solution is more efficiently obtained by directly solving the steady formulation of the equations, than by solving the transient Navier-Stokes equations until the steady state is reached.

It is well known that the Newton method shows the property of having quadratic convergence

sufficiently close to the solutionDeuflhard (2011). However, if the initial solution guess is not adequate, the method converge to a solution slowly, or may not even converge at all. For some highly nonlinear problems this is a significant issue. The so-called globalized Newton methods are used to overcome this problem by effectively increasing the area of convergence of the method. A popular globalized Newton method is the Newton-Dogleg method, Pawlowski et al. (2008) which is based on a trust-region technique.

In the present work, a steady low-Mach pressure-based solver for the simulation of temperature dependent non-reactive and reactive flows using the Discontinuous Galerkin method is presented. To the best of our knowledge, this is the first time that a pressure-based solver is used together with the DG method for solving the reactive low-Mach equations using the Newton-Dogleg method in a fully coupled manner. We focus in this study on two-dimensional configurations, but the ideas presented could be extended to three-dimensional systems. The one-step combustion model presented by Fernandez-Tarrazo et al. Fernandez-Tarrazo et al. (2006) is used for describing the chemical reactions. In the present work we consider only methane combustion, but the one-step model could be used for other hydrocarbons as well. We choose this chemical model to obtain physically correct results for a wide range of applications, while avoiding the use of complex chemistry models. The discrete system of equations is solved by a globalized Newton method, by means of the Dogleg approach. In addition to the Newton-Dogleg method, a homotopy strategy is presented, which was found to be useful for obtaining solutions of steady state calculations of highly nonlinear problems. In order to find appropriate initial values for Newton's method in combustion applications, the concept of flame sheet estimates (i.e. the solution for infinitely fast chemistry) is used. Several benchmark cases are presented that allow us to validate our implementation. First we solve the differentially heated cavity problem, with which we intend to validate our implementation of the low-Mach solver for non-constant density flows using the fully coupled solver. Later two flame configurations are calculated, namely the counter-diffusion flame and the chambered diffusion flame. (Matalon et al., 1980)

In the following, we consider a two-dimensional system. However, the methods shown in this work could be also used for three-dimensional problems.

They were first derived by Rehm and Baum (Rehm and Baum, 1978). A rigorous extension to combustion problems was done by Majda and Sethian (Majda and Sethian, 1985).

Pitsch and Peters (1998) Habla sobre pposibles formas de escribir las ecuaciones para un flamelet de tal forma que sea posible contabilizar la diffusion diferencial (non unity lewis numbers.)

1.1.2 The BoSSS code

The presented solver is embedded in the *BoSSS* (Bounded Support Spectral Solver) code, which is under active development at the chair of fluid dynamics of the Technical University of Darmstadt, and is available under https://github.com/FDYdarmstadt/BoSSS.

BoSSS is a general framework for the discretization of conservation laws using the DG method and uses a modal DG approach with orthonormal Legendre polynomials as basis functions. The *BoSSS* code features a variety of applications in the context of computational fluid dynamics, such as a solver for multiphase flows with a sharp interface approach, (Kummer, 2017) an incompressible Immersed Boundary Method solver for particle laden flows,(Krause and Kummer, 2017) a solver for viscoelastic fluid flows,(Kikker et al., 2020) and a solver for

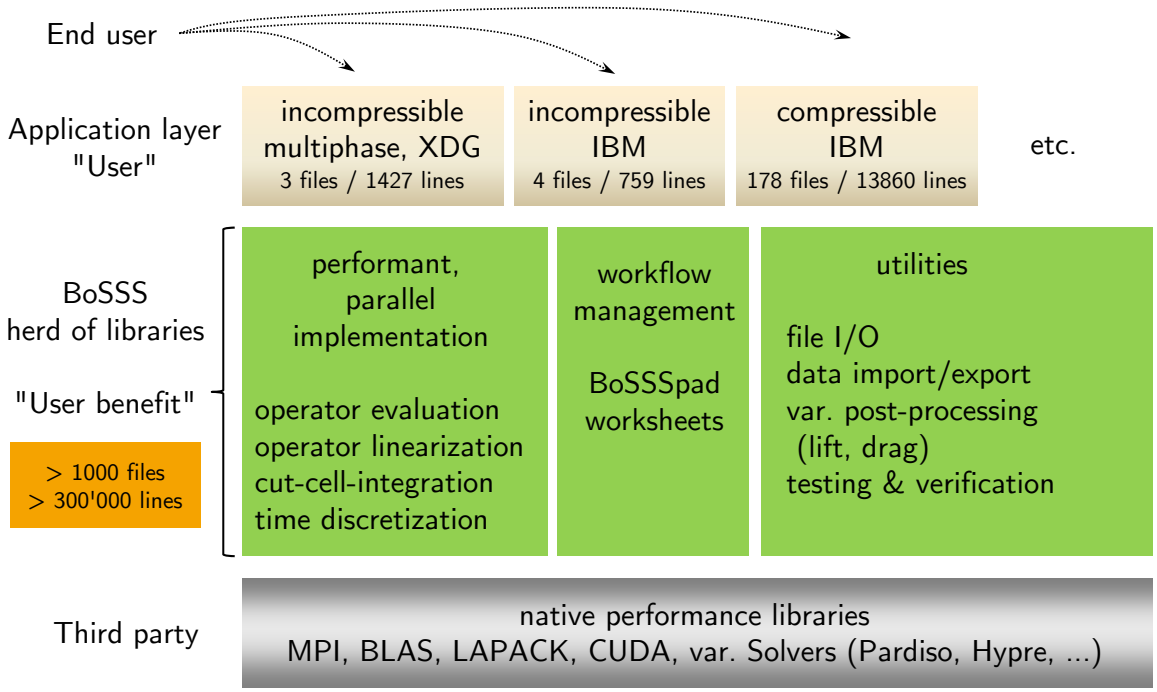


Figure 1.1: Schematic representation of the estructure of the BoSSS solver from the BoSSS handbook (Kummer et al., 2020).

Fig:BoSSS

compressible flows, (Geisenhofer et al., 2019) among others.

From this point on, the solver presented in this work will be called XNSEC (e**X**tended **N**avier-**S**tokes for **C**ombustion). The term *extended* refers to the framework on which the solver is built, which focuses on applications for multi-phase flows using a sharp interface approach using a level-set method. This point will be briefly discussed in the section

1.1.3 Motivation and objectives of this work

two-dimensional

2 Governing equations

ch:gov_eqs

The objective of this work is to present a methodology that allows the simulation of reactive fluids - with emphasis on combustion systems - making use of the low-Mach equations. In this chapter, it is intended to show the governing equations used in this work to give some remarks regarding their derivation, as well the assumptions made within the present framework.

First, in Section 2.1, the governing equations that allow the description of reactive fluids are presented, in addition to other thermodynamic relationships, expressions for the transport parameters and the one-step chemical model. Subsequently, in Section 2.1.2 the nondimensionalization of the equations is presented, as well as the low-Mach limit of the governing equations.

Finally, in Section 2.2 the governing equations for an irreversible chemical model with an infinite reaction rate are presented. These equations are used as part of the algorithm to solve the finite reaction rate case.

2.1 The low-Mach number equations for reactive flows

sec:GovEqLowMach

Combustion processes can be modeled by a system of non-linear partial differential equations, namely the balance equations for the total mass, momentum, energy, and mass of individual species (usually expressed in terms of mass fractions). This system needs to be solved together with an equation of state and expressions for the transport properties as well as for the chemical reaction rates. The derivation of the governing equations for a reacting flow system can be found in the literature. For more information see for example the works from Kee et al. (2003) and Poinsot and Veynante (2005). In the following pages, the main ideas regarding the derivation of equations are presented. For a more detailed explanation, the interested reader is referred to the cited works and the references therein.

2.1.1 The reactive Navier–Stokes equations

ssect:lowMachModel

Throughout this work, variables with a hat sign, for example $\hat{\rho}$, represent dimensional variables, while those without it are nondimensional. The derivation of the low-Mach number equations starts with the Navier–Stokes equations, the energy equation (in its temperature form), and the species transport equations. Consider a reacting fluid mixture composed of N species. Let $\hat{\mathbf{x}} = (\hat{x}, \hat{y}, \hat{z})$ and $\hat{t}$ be the spatial vector and time. The primitive variables are the velocity field $\hat{\mathbf{u}} = (\hat{u}, \hat{v}, \hat{w})$, the pressure $\hat{p}$, the temperature $\hat{T}$, and the mass fractions Y_k of the N total species. The set of governing equations to be solved is

$$\frac{\partial \hat{\rho}}{\partial \hat{t}} + \hat{\nabla} \cdot (\hat{\rho}\hat{\mathbf{u}}) = 0, \tag{2.1a}$$ {eq:NS_Conti}

$$\frac{\partial \hat{\rho}\hat{\mathbf{u}}}{\partial \hat{t}} + \hat{\nabla} \cdot (\hat{\rho}\hat{\mathbf{u}} \otimes \hat{\mathbf{u}}) = -\hat{\nabla}\hat{p} - \hat{\nabla} \cdot \hat{\boldsymbol{\tau}} - \hat{\rho}\hat{\mathbf{g}}, \tag{2.1b}$$ {eq:NS_Momentum}

$$\hat{\rho}\hat{c}_p\frac{\partial \hat{T}}{\partial \hat{t}} + \hat{\rho}\hat{c}_p\hat{\mathbf{u}}\cdot\hat{\nabla}\hat{T} = \frac{\mathrm{D}\hat{p}}{\mathrm{D}t} - \hat{\nabla}\cdot\hat{\mathbf{q}} - \left(\hat{\rho}\sum_{k=1}^{N}\hat{c}_{p,k}Y_k\hat{\mathbf{U}}_k\right)\cdot\hat{\nabla}\hat{T} + \hat{\boldsymbol{\tau}}:\hat{\nabla}\hat{\mathbf{u}} + \hat{\omega}_T, \tag{2.1c}$$ {eq:NS_energy}

$$\frac{\partial \hat{\rho}Y_k}{\partial \hat{t}} + \hat{\nabla}\cdot(\hat{\rho}\hat{\mathbf{u}}Y_k) = -\hat{\nabla}\cdot\hat{\mathbf{j}}_k + \hat{\omega}_k \quad (k = 1, \ldots, N) \tag{2.1d}$$ {eq:NS_SpeciesBalance}

eq:NS-eq

In the general case this results in $N+5$ differential equations to be solved. In these equations, $\hat{\rho}$ is the density of the mixture and $\hat{\mathbf{g}}$ is the acceleration of gravity. $\hat{\boldsymbol{\tau}}$, $\hat{\mathbf{q}}$ and $\hat{\mathbf{j}}_k$ are the viscous tensor, the heat flux vector and the molecular mass flux vector of species k, respectively. Additionally, $\hat{\omega}_T$ is the heat release from combustion and $\hat{\omega}_k$ is the reaction rate of species k. To close the system, expressions must be defined to link these variables with the primitive variables. They will be briefly shown and commented upon in the following paragraphs.

Equation of state

Assuming that the fluid behaves ideally, the density can be calculated as

$$\hat{\rho} = \frac{\hat{p}\hat{W}_{\text{avg}}}{\mathcal{R}\hat{T}}. \tag{2.2}$$ {eq:IdealGassDimensional}

Here, $\hat{\mathcal{R}}$ is the universal gas constant of gases, and $\hat{W}_{\text{avg}}$ is the average molecular weight of the fluid, defined as

$$\hat{W}_{\text{avg}} = \left(\sum_{k=1}^{N}\frac{Y_k}{\hat{W}_k}\right)^{-1} \tag{2.3}$$

with $\hat{W}_k$ as the molecular weight of species k. For an ideal mixture, the specific heat capacity can be calculated as a weighted average of the specific heats of the species

$$\hat{c}_p = \sum_{k=1}^{N} Y_k\hat{c}_{p,k}, \tag{2.4}$$

where $\hat{c}_{p,k}$ corresponds to the specific heat capacity of the component k. The temperature dependence of $\hat{c}_{p,k}$ can be accounted for by using NASA polynomials (Mcbride et al., 1993)

$$\hat{c}_{p,k} = \left(\hat{a}_1 + \hat{a}_2\hat{T} + \hat{a}_3\hat{T}^2 + \hat{a}_4\hat{T}^3 + \hat{a}_5\hat{T}^4\right)\frac{\hat{\mathcal{R}}}{\hat{W}_k}, \tag{2.5}$$

where $\hat{a}_1$, $\hat{a}_2$, $\hat{a}_3$, $\hat{a}_4$ and $\hat{a}_5$ are numerical coefficients supplied by the NASA database.

Transport models

The viscous tensor $\hat{\boldsymbol{\tau}}$ is defined for a Newtonian fluid as

$$\hat{\boldsymbol{\tau}} = -\hat{\mu}\left(\hat{\nabla}\hat{\mathbf{u}} + (\hat{\nabla}\hat{\mathbf{u}})^T\right) + \left(\frac{2}{3}\hat{\mu} - \hat{\kappa}\right)(\hat{\nabla}\cdot\hat{\mathbf{u}})\boldsymbol{I}. \tag{2.6}$$

Here, $\hat{\mu}$ is the dynamic viscosity of the fluid, which is generally specific to the fluids and depends on its temperature and pressure. Furthermore, $\hat{\kappa}$ corresponds to the bulk viscosity,

which is usually negligible for fluids at low pressures (Bird et al., 1960). In the rest of this work $\hat{\kappa}$ will be taken to be equal to zero.

The heat flux vector $\hat{\mathbf{q}}$ is given by Fourier's law of heat conduction,

{eq:fourierLaw}
$$\hat{\mathbf{q}} = \hat{\lambda}\hat{\nabla}\hat{T}. \tag{2.7}$$

Here, $\hat{\lambda}$ corresponds to the thermal conductivity, which, similar to viscosity, depends on the particular fluid under study as well as its temperature and pressure. Soret and Dufour effects are not considered in the present work.

The molecular mass flux vector $\hat{\mathbf{j}}_k$ of species k is defined as $\hat{\mathbf{j}}_k = \hat{\rho}\hat{\mathbf{U}}_k$, where $\hat{\mathbf{U}}_k$ is the diffusion velocity of the component k. In general, $\hat{\mathbf{U}}_k$ can be obtained by solving the Maxwell-Stefan equations

{MaxwellStefan}
$$\nabla X_p = \sum_{k=1}^{N} \frac{X_p X_k}{D_{pk}}(\mathbf{U}_k - \mathbf{U}_p), \qquad p = 1, \ldots, N. \tag{2.8}$$

Here $D_{pk} = D_{kp}$ is the binary mass diffusion coefficient of species p into species k. X_p is the mole fraction of species k and is related to the mass fraction of k as $X_k = Y_k\hat{W}/\hat{W}_k$. The solution of the system (2.8) is often a difficult and costly task (Williams, 2000; Poinsot and Veynante, 2005), and often simplifications are made. It can be shown that for binary mixtures ($N = 2$), and for mixtures containing multiple species ($N > 2$), where all diffusion coefficients are equal, the system (2.8) reduces exactly to the well-known Fick's law

$$\hat{\mathbf{j}}_k = -\rho\hat{D}_k\nabla Y_k. \tag{2.9}$$

eq:FickLaw

This expression is exact only in the cases mentioned above. The variable $\hat{D}_k$ corresponds in this case to the diffusion coefficient of species k in the mixture. An issue related to global mass conservation can be noted here. Recall that by definition, the sum of the mass fractions must always be one, namely $\sum_{k=1}^{N} Y_k = 1$. This is only true for the solution of Equation (2.1) if exact expressions for diffusion velocities are used (Poinsot and Veynante, 2005). If some inexact expression is used (as, for example, Fick's law), the constraint for the sum of the mass fractions will not be fulfilled. This problem can be solved by solving the global mass conservation (continuity) equation Equation (2.1a) and only the equations for the first $N - 1$ species Equation (2.1d). Therefore, all inconsistencies originating from not using an exact species diffusion model are absorbed by Y_N. As pointed out in Poinsot and Veynante (2005), this simplification should only be used if all $N - 1$ species are strongly diluted in species N, such as the case of a flame in air, where the mass fraction of nitrogen is large. It can also be noted that this approach reduces the number of differential equations needed to be solved by one.

The temperature dependence of viscosity is modeled by Sutherland's law (Sutherland, 1893)

{eq:DimSutherland}
$$\hat{\mu}(\hat{T}) = \hat{\mu}_{\text{suth}} \left(\frac{\hat{T}}{\hat{T}_{\text{suth}}} \right)^{1.5} \frac{\hat{T}_{\text{suth}} + \hat{S}}{\hat{T} + \hat{S}}. \tag{2.10}$$

Here $\hat{\mu}_{\text{suth}}$ is the viscosity evaluated at a reference temperature $\hat{T}_{\text{suth}}$, and $\hat{S}$ is a material-dependent parameter. In all calculations in this work, the value of $\hat{S}$ for air is used, i.e. $\hat{S} = 110.5\,\text{K}$. Expressions for determining the thermal conductivity and diffusion coefficients as a function of temperature can be obtained using similar expressions, as shown later in Section 2.1.2.

The enthalpy transport term due to diffusive fluxes (third therm on the right hand side of Equation (2.1c)) usually has only a small influence on the solution (Smoke and Giovangigli (1991), Goey et al. (1995), and Paxion et al. (2001)), and it is actually exactly equal to zero for systems where all species heat capacities are equal. In the present formulation this term is neglected in the energy equation.

Chemical model

Consider a system composed of N species where M chemical reactions take place. Chemical reactions can be written in generalized form as follows.

$$\sum_{k=1}^{N} \nu'_{kj} \mathcal{M}_k \rightleftharpoons \sum_{k=1}^{N} \nu''_{kj} \mathcal{M}_k \qquad \text{for} \qquad j = 1, \ldots, M\,, \tag{2.11}$$ {eq:allChemEq}

where ν'_{jk} and ν''_{jk} are the molar stoichiometric coefficients of species k in the chemical reaction j, and $\mathcal{M}_k$ represents the chemical component k.

The reaction rate of species k is $\hat{\omega}_k$, which accounts for the total amount of species k that appear or disappear due to M chemical reactions (c.f. Equations (2.11)). Its given by

$$\hat{\omega}_k = \hat{W}_k \sum_{j=1}^{M} \nu_{jk} \hat{\mathcal{Q}}_j. \tag{2.12}$$ {eq:reacRateDef}

Here $\nu_{kj} = \nu''_{kj} - \nu'_{kj}$, and $\mathcal{Q}_j$ is the rate of progress of the reaction j. They are usually modeled using Arrhenius-type expressions.

The heat release $\hat{\omega}_T$ that appears in the energy equation is related to the reaction rates according to

$$\hat{\omega}_T = -\sum_{k=1}^{N} \hat{h}_k \hat{\omega}_k = -\sum_{k=1}^{N} \Delta \hat{h}^0_k \hat{\omega}_k - \sum_{k=1}^{N} \hat{h}_{ks} \hat{\omega}_k. \tag{2.13}$$ {eq:heatRelDef}

Here, the specific enthalpy of k-th species $\hat{h}_k$ is written in terms of its formation enthalpy $\hat{h}^0_k$ and a sensible enthalpy $\hat{h}_{ks} = \int_0^{\hat{T}} \hat{c}_{p,k} \mathrm{d}\hat{T}$. The second term on the right-hand side of Equation (2.13) is usually small and is exactly zero for a mixture where all the heat capacities of each component are equal (Poinsot and Veynante, 2005). It will be neglected in the rest of the analysis.

In this work, the one-step kinetic model for the combustion of hydrocarbons presented in Fernandez-Tarrazo et al. (2006) is used. The chemical reaction is represented by a single ($M = 1$) exothermic global irreversible expression as

$$\mathrm{C}_n\mathrm{H}_m + \left(n + \frac{m}{4}\right) \mathrm{O}_2 \longrightarrow n\mathrm{CO}_2 + \frac{m}{2}\mathrm{H}_2\mathrm{O}. \tag{2.14}$$

The rate of progress of the global reaction is modeled by an Arrhenius-type expression

$$\hat{\mathcal{Q}} = \hat{B} e^{-\hat{T}_a/\hat{T}} \left(\frac{\hat{\rho} Y_F}{\hat{W}_F}\right)^a \left(\frac{\hat{\rho} Y_O}{\hat{W}_O}\right)^b. \tag{2.15}$$ {eq:DimArr}

$\hat{B}$ $(\mathrm{cm}^3/(\mathrm{mol\,s}))$	$\hat{T}_{a0}$ (K)	$\hat{\mathrm{Q}}_0$ $(\mathrm{MJ\,kmol}^{-1})$	a	b
6.9×10^{14}	15 900	802.4	1	1

Table 2.1: Base parameters used in the one-step combustion model by Fernandez-Tarrazo et al. (2006) tab:OneStepParameters

Here, the subscripts F and O refer to fuel and oxidizer, respectively. The parameter $\hat{B}$ corresponds to the pre-exponential factor, $\hat{T}_a$ is the activation temperature, and a and b are reaction orders. For a one-step reaction model, the reaction rate of the k-th component inEquation (2.12) is

$$\hat{\omega}_k = \nu_k \hat{W}_k \hat{\mathcal{Q}}. \tag{2.16}$$

With this definition, the heat release $\hat{\omega}_T$ in Equation (2.13) yields

$$\hat{\omega}_T = -\hat{W}_F \hat{\omega}_F \hat{Q}^m = -\hat{\omega}_F \hat{Q}. \tag{2.17}$$

Here $\hat{Q}^m$ is the molar heat of reaction of the one-step reaction and $\hat{Q}$ is the mass heat of reaction. $\hat{W}_F$ and $\hat{\omega}_F$ are the molar mass of the fuel species and the reaction rate of the fuel species respectively.

Within the model from Fernandez-Tarrazo et al. (2006), several parameters are adjusted to represent characteristic features of premixed flames and diffusion flames. The parameters $\hat{T}_a$ and $\hat{\mathrm{Q}}$ are defined as functions of the local equivalence ratio ϕ, which is, in turn, defined in terms of the local mass fractions of fuel Y_F and oxidizer Y_O as

$$\phi = \frac{sY_F^0}{Y_O^0} \frac{sY_F - Y_O + Y_O^0}{s(Y_F^0 - Y_F) + Y_O}, \tag{2.18}$$ {eq:equivalenceRatio}

where Y_F^0 and Y_O^0 are the mass fractions of the fuel and oxidizer flows in their corresponding feed streams, and s is the mass stoichiometric ratio, defined as $s = \nu_O \hat{W}_O / \nu_F \hat{W}_F$. The activation energy depend on ϕ as

$$\hat{T}_a(\phi) = \begin{cases} (1 + 8.250(\phi - 0.64)^2)\hat{T}_{a0} & \text{if } \phi \leq 0.64, \\ \hat{T}_{a0} & \text{if } 0.64 \leq \phi \leq 1.07, \\ (1 + 4.443(\phi - 1.07)^2)\hat{T}_{a0} & \text{if } \phi \geq 1.07, \end{cases} \tag{2.19}$$ {eq:ActivationTemperatureOneStep}

and the molar heat release according to

$$\hat{\mathrm{Q}}(\phi) = \begin{cases} \hat{\mathrm{Q}}_0 & \text{if } \phi \leq 1, \\ (1 - \alpha(\phi - 1))\hat{\mathrm{Q}}_0 & \text{if } \phi > 1. \end{cases} \tag{2.20}$$ {eq:heatReleaseOneStep}

The parameter α is a constant that depends on the hydrocarbon being considered, in particular $\alpha = 0.21$ for methane combustion.

It should be noted that Equation (2.20) yields unphysical values of $\hat{Q}$ for large values of ϕ. This problem can be avoided by setting an upper boundary value for ϕ in Equation (2.20). However, in practice, this should not have a significant effect because the non-physical values of $\hat{Q}$ appear in zones where the reaction rate $\hat{\omega}$ is very close to zero, making the factor $\hat{Q}\hat{\omega}$ in the temperature equation negligible. However, setting an upper bound for ϕ is helpful in avoiding possible numerical instabilities.

2.1.2 The reactive low-Mach Navier-Stokes equations

ssec:NonDimLowMachEquations

In the present work, the low-Mach number approximation of the governing equations is used. The derivation of the equations can be found in the works from Rehm and Baum (1978), Majda and Sethian (1985), and Müller (1998). The interested user is referred to these references for a detailed explanation of how the set of equations is derived. In what follows, the main consequences of the low Mach limit will be shown and discussed.

Recall the definition of the Mach number, $\mathrm{Ma} = \hat{u}_{\mathrm{ref}}/\hat{c}$, where $\hat{u}_{\mathrm{ref}}$ is a characteristic flow velocity and $\hat{c}$ the speed of sound. The low-Mach number limit approximation of the governing equations is used for flows where the Mach number is small, which is usually the case in typical laminar combustion systems (Dobbins and Smooke, 2010). The low-Mach equations are obtained by using standard asymptotic methods. One of the main results of the analysis is that for flows with a small Mach number, the pressure can be decomposed as

$$\hat{p}(\hat{\mathbf{x}}, \hat{t}) = \underbrace{\hat{p}_0(\hat{t})}_{\mathcal{O}(1)} + \underbrace{\hat{p}_2(\hat{\mathbf{x}}, \hat{t})}_{\mathcal{O}(\mathrm{Ma}^2)}. \tag{2.21}$$

The spatially uniform term $\hat{p}_0(\hat{t})$ is called thermodynamic pressure and only influences the system through the equation of state. It is constant in space but can change in time. For an open system, the thermodynamic pressure is also constant in time and equal to the ambient pressure, while for a closed system (e.g. a system completely bounded by walls) it changes in order to ensure mass conservation.

On the other hand, the perturbational term $\hat{p}_2(\hat{\mathbf{x}}, \hat{t})$ appears only in the momentum equations and plays a role similar to that of the pressure in the classical incompressible formulation. This perturbational term satisfies $\hat{p}_2/\hat{p} \sim \mathcal{O}(\mathrm{Ma})^2$ (Dobbins and Smooke, 2010; Nonaka et al., 2018) showing that the equation of state is satisfied only to $\mathcal{O}(\mathrm{Ma}^2)$.

Effectively, the low-Mach limit of the Navier-Stokes equations allows the calculation of systems where large density variations due to temperature differences are present, thus the formulation is not restricted to approximations such as the Boussinesq approximation for buoyancy-driven flow. In addition, this approximation truncates the mechanism of pressure wave propagation, which is a natural feature of the compressible Navier-Stokes equations. By doing this the necessity of small time-steps for resolving the wave phenomenon is completely removed and the maximum allowed time-step is greatly increased.

In this work a nondimensional formulation of the governing equations is used. Following nondimensional quantities are defined

$$\rho = \frac{\hat{\rho}}{\hat{\rho}_{\mathrm{ref}}}, \quad p = \frac{\hat{p}}{\hat{p}_{\mathrm{ref}}}, \quad \mathbf{u} = \frac{\hat{\mathbf{u}}}{\hat{u}_{\mathrm{ref}}}, \quad T = \frac{\hat{T}}{\hat{T}_{\mathrm{ref}}}, \quad c_p = \frac{\hat{c}_p}{\hat{c}_{p,\mathrm{ref}}}, \quad W_k = \frac{\hat{W}_k}{\hat{W}_{\mathrm{ref}}},$$
$$\mu = \frac{\hat{\mu}}{\hat{\mu}_{\mathrm{ref}}}, \quad D_k = \frac{\hat{D}_k}{\hat{D}_{k,\mathrm{ref}}}, \quad k = \frac{\hat{k}}{\hat{k}_{\mathrm{ref}}} \quad \nabla = \frac{\hat{\nabla}}{\hat{L}_{\mathrm{ref}}}, \quad t = \frac{\hat{t}}{\hat{t}_{\mathrm{ref}}}, \quad \mathbf{g} = \frac{\hat{\mathbf{g}}}{\hat{g}_{\mathrm{ref}}}, \quad Q = \frac{\hat{Q}}{\hat{Q}_0}.$$

Here $\hat{u}_{\mathrm{ref}}$, $\hat{L}_{\mathrm{ref}}$, $\hat{p}_{\mathrm{ref}}$, $\hat{t}_{\mathrm{ref}}$, and $\hat{T}_{\mathrm{ref}}$ are the reference velocity, length, pressure, time, and temperature, respectively, and are equal to some characteristic value for the particular studied configuration. Furthermore, $\hat{g}_{\mathrm{ref}}$ is the magnitude of the gravitational acceleration and $\hat{W}_{\mathrm{ref}}$ is the reference molecular weight. The reference transport properties $\hat{\mu}_{\mathrm{ref}}$, $\hat{k}_{\mathrm{ref}}$, $\hat{D}_{k,\mathrm{ref}}$ and the reference heat capacity of the mixture $\hat{c}_{p,\mathrm{ref}}$ are evaluated at the reference temperature $\hat{T}_{\mathrm{ref}}$.

Similarly, the reference density must satisfy the equation of state, thus $\hat{\rho}_{\text{ref}} = \hat{p}_{\text{ref}}/(\hat{\mathcal{R}}\hat{T}_{\text{ref}}\hat{W}_{\text{ref}})$. By introducing these definitions into the governing Equations (2.1a) to (2.1d) the reactive set of nondimensional low-Mach number equations is obtained. The system of differential equations to be solved reads as follows.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.22a}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\nabla p + \frac{1}{\text{Re}} \nabla \cdot \mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T - \frac{2}{3} (\nabla \cdot \mathbf{u}) \boldsymbol{I} \right) - \frac{1}{\text{Fr}^2} \rho \mathbf{g}, \tag{2.22b}$$

$$\frac{1}{\gamma} \frac{\partial \rho T}{\partial t} + \nabla \cdot (\rho \mathbf{u} T) = \frac{1}{\text{Re Pr}} \nabla \cdot \left(\frac{k}{c_p} \nabla T \right) + \text{H Da} \, \frac{\text{Q} \, \mathcal{Q}}{c_p}, \tag{2.22c}$$

$$\frac{\partial \rho Y_k}{\partial t} + \nabla \cdot (\rho \mathbf{u} Y_k) = \frac{1}{\text{Re Pr Le}_k} \nabla \cdot (\rho D \nabla Y_k) + \text{Da} \, \nu_k W_k \mathcal{Q}, \quad k = 1, \ldots, N-1. \tag{2.22d}$$

This system is solved for the primitive variables velocity $\mathbf{u} = (u_x, u_y)$, pressure p, temperature T and mass fractions $\mathbf{Y} = (Y_1, \ldots, Y_{N-1})$. Note that it is assumed that the spatial gradients of the mixture heat capacity are small, which allows to introduce c_p in the derivative of the diffusive term of Equation (2.22c). Furthermore, considering the fact that the sum of the mass fractions must always be one, the mass fraction of the last species N can be calculated with

$$Y_N = 1 - \sum_{k=1}^{N-1} Y_k. \tag{2.23}$$

Note that the form of the low-Mach equations is very similar to the Navier-Stokes equations. The major difference is in the decomposition of the pressure. This similarity is beneficial as it allows the use of similar techniques to solve the PDE system to those used for the completely incompressible case (Keshtiban et al., 2003). From now on, the sub-index of the hydrodynamic pressure $\hat{p}_2$ will be dropped and it will be called simply $\hat{p}$, further emphasizing the similarity in its role to the pressure of the incompressible equations.

Six nondimensional factors arise from the nondimensionalization process:

$$\text{Re} = \frac{\hat{\rho}_{\text{ref}} \hat{u}_{\text{ref}} \hat{L}_{\text{ref}}}{\hat{\mu}_{\text{ref}}}, \quad \text{Fr} = \frac{\hat{u}_{\text{ref}}}{\sqrt{\hat{g}_{\text{ref}} \hat{L}_{\text{ref}}}}, \quad \text{Pr} = \frac{\hat{c}_{p,\text{ref}} \hat{\mu}_{\text{ref}}}{\hat{k}_{\text{ref}}},$$

$$\text{Le}_k = \frac{\hat{k}_{\text{ref}}}{\hat{\rho}_{\text{ref}} \hat{D}_{k,\text{ref}} \hat{c}_{p,\text{ref}}}, \quad \text{Da} = \frac{\hat{B} \hat{L}_{\text{ref}} \hat{\rho}_{\text{ref}}}{\hat{M}_{\text{ref}} \hat{u}_{\text{ref}}}, \quad \text{H} = \frac{\hat{\text{Q}}_0}{\hat{c}_{p,\text{ref}} . \hat{T}_{\text{ref}}}$$

The first three equations define the Reynolds, Froude and Prandtl number, respectively. Le_k is the Lewis number of species k. Finally, Da and H are the Damköhler number and the nondimensional heat release, respectively. The nondimensional progress of the global reaction reads as follows

$$\mathcal{Q}(T, \mathbf{Y}) = \left(\frac{\rho Y_F}{M_F} \right) \left(\frac{\rho Y_O}{M_O} \right) \exp \left(\frac{-T_a}{T} \right), \tag{2.24}$$

where $T_a = \hat{T}_a / \hat{T}_{\text{ref}}$. Furthermore, the nondimensional heat release is

$$\text{Q}(\phi) = \begin{cases} 1 & \text{if } \phi \leq 1, \\ (1 - \alpha(\phi - 1)) & \text{if } \phi > 1. \end{cases} \tag{2.25}$$

with ϕ evaluated according to Equation (2.18). In the low-Mach limit, the ideal gas equation depends on the thermodynamic pressure, temperature and mass fractions. It reads in its nondimensional form

$$\rho(p_0, T, \mathbf{Y}) = \frac{p_0}{T \sum_{k=1}^{N} \frac{Y_k}{W_k}}. \tag{2.26}$$ {eq:ideal_gas}

As mentioned above, the thermodynamic pressure of a closed system is a parameter that has to be determined (for an open system is equal to the atmospheric pressure). By integrating Equation (2.26) into the entire domain Ω, the expression

$$p_0(T, \mathbf{Y}) = \frac{m_0}{\int_\Omega \left(T \sum_{k=1}^{N} \frac{Y_k}{W_k} \right)^{-1} \mathrm{d}V} \tag{2.27}$$ {eq:p0Condition}

can be derived. Here, $m_0 = \int_\Omega \rho \mathrm{d}V$ is the mass of the fluid in the closed system. Since in a closed system the total mass is constant, it can be determined using initial conditions. Similarly, the nondimensional specific heat capacity of the mixture c_p is calculated as

$$c_p(T, \mathbf{Y}) = \sum_{k=1}^{N} Y_k c_{p,\alpha}, \tag{2.28}$$ {eq:nondim_cpmixture}

and the nondimensional viscosity as

$$\mu(T) = T^{\frac{3}{2}} \frac{1 + \hat{S}}{\hat{T}_{\text{ref}} T + \hat{S}}. \tag{2.29}$$ {eq:nondim_sutherland}

The model for the transport parameters can be simplified by assuming constant values for the Prandtl and Lewis numbers (Smoke and Giovangigli, 1991). The nondimensional transport parameters are related in that case with

$$\mu = k/c_p = \rho D. \tag{2.30}$$

2.2 The flame sheet approximation

sec:FlameSheet

In this section, the concept of the flame sheet approximation is introduced, which is used in the solution algorithm to obtain solutions of the finite reaction rate system given by Equations (2.22a) to (2.22d). The ideas proposed in the work of Keyes and Smooke (1987) are followed.

Assuming that all species have the same heat capacity $c_{p,k} = c_p$ and mass diffusion coefficient $D_k = D$, that the Lewis number is unity for all species, and that combustion can be described by a single step chemical reaction, it is possible to obtain an equation for a scalar without source terms, which is a linear combination of the energy Equation (2.22c) and mass fraction Equation (2.22d). Thus, the system can be simplified to solving the low-Mach Navier-Stokes equations and an equation for the passive scalar z:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.31a}$$ {eq:MixtFracConti2}

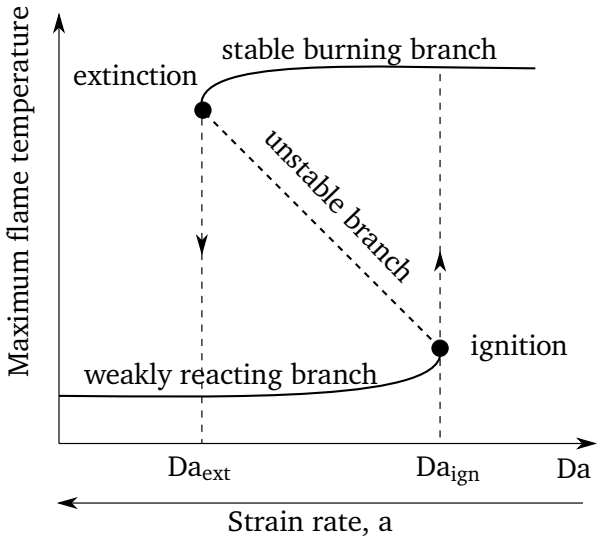


Figure 2.1: S-shaped bifurcation curve of a combustion process. fig:Sshaped

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\nabla p + \frac{1}{\mathrm{Re}} \nabla \cdot \mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T - \frac{2}{3} (\nabla \cdot \mathbf{u}) \boldsymbol{I} \right) - \frac{1}{\mathrm{Fr}^2} \rho \mathbf{g}, \tag{2.31b}$$ {eq:MixtFracMom}

$$\frac{\partial \rho z}{\partial t} + \nabla \cdot (\rho \mathbf{u} z) = \frac{1}{\mathrm{Re\,Pr}} \nabla \cdot (\rho D \nabla z), \tag{2.31c}$$ {eq:MixtFracMF}

eq:all-eq-mixfrac

which are solved together with the equation of state (Equation (2.26)) and expressions for the transport parameters (Equation (2.29)). Here z is the mixture fraction, which is a scalar that measures the local fuel/oxidizer ratio (Poinsot and Veynante, 2005). The variable z is per definition equal to unity in the fuel feed stream and equal to zero in the oxidizer feed stream. Note that the system is not closed, because ρ, μ and ρD are still functions of the temperature and mass fractions. These fields can be related to the mixture fraction using the Burke-Schumann flame structure concept (Burke and Schumann, 1928).

In the case of an infinitely fast chemical reaction, fuel and oxidizer cannot coexist. On one side of this sheet only oxidizer is found, and on the other side only fuel. The exact position of the flame sheet can be determined by finding the location of the points where the reactant mass fractions Y_F and Y_O meet in stoichiometric proportions, that is, the points where the mixture fraction $z = z_{st}$, with

$$z_{st} = \frac{Y_O^0}{Y_O^0 + s Y_F^0}. \tag{2.32}$$

Here Y_O^0 is the mass fraction of the oxidizer in the oxidizer inlet stream, and Y_F^0 is the mass fraction of fuel in the fuel inlet stream. The Burke-Schumann solution provides analytical expressions for temperature and mass fraction fields on either side of the flame sheet as a function of the mixture fraction z. For a more detailed derivation see for example the textbook from Poinsot and Veynante (2005) or the work from Keyes and Smooke (1987). The

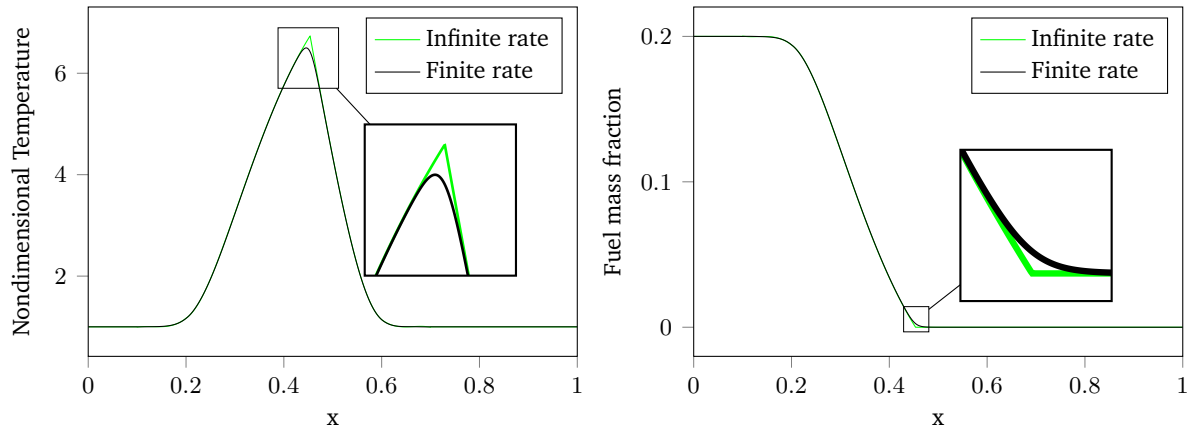


Figure 2.2: Temperature and fuel mass fraction profiles calculated in the center-line of a counter-flow flame configuration using finite chemistry (black) and the flame sheet approximation (green).
fig:MixtureFraction_finiteRateComparison

temperature is related to the mixture fraction as

$$T(z) = \begin{cases} zT_F^0 + (1-z)T_O^0 + \dfrac{QY_F^0}{c_p} z_{st} \dfrac{1-z}{1-z_{st}} & \text{if} \quad z \geq z_{st}, \\ zT_F^0 + (1-z)T_O^0 + \dfrac{QY_F^0}{c_p} z & \text{if} \quad z < z_{st}. \end{cases} \tag{2.33}$$
{eq:BS-T}

The mass fraction field of fuel and oxidizer species at either side of the flame is given by:

$$Y_F(z) = \begin{cases} Y_F^0 \dfrac{z - z_{st}}{1 - z_{st}} & \text{if} \quad z \geq z_{st}, \\ 0 & \text{if} \quad z < z_{st}, \end{cases} \tag{2.34}$$
{eq:BS-YF}

$$Y_O(z) = \begin{cases} 0 & \text{if} \quad z \geq z_{st}, \\ Y_O^0 \left(1 - \dfrac{z}{z_{st}}\right) & \text{if} \quad z < z_{st}. \end{cases} \tag{2.35}$$
{eq:BS-YO}

Finally, the mass fraction field of product species P is:

$$Y_P(z) = \begin{cases} Y_O^0 \dfrac{W_P \nu_P}{W_O \nu_O} (1-z) & \text{if} \quad z \geq z_{st}, \\ Y_F^0 \dfrac{W_P \nu_P}{W_F \nu_F} z & \text{if} \quad z < z_{st}. \end{cases} \tag{2.36}$$
{eq:BS-YP}

Once the mixture fraction field z is obtained, the temperature and mass fraction fields are uniquely defined by Equations (2.33) to (2.36), which are used to evaluate the density and the transport properties.

Note that the expressions shown above are not differentiable at the stoichiometric point, which also means that the density and transport properties are not differentiable at $z = z_{st}$. This presents a challenge for algorithms for finding solutions of the system (Rauwoens et al., 2009). Later in Section 4.4 this point will be treated.

The main idea of introducing the system of equations presented in this section is to obtain an adequate initial estimate that can be used to find a solution for the system with a finite reaction rate (Equations (2.22a) to (2.22d)). Under certain conditions, the approximation is indeed very good. In Figure 2.2, temperature and fuel mass fraction fields obtained with infinite and finite reaction rate across the center-line of a counterflow flame configuration are shown. Clearly, both solutions are very similar, differing only in the area near the flame. However, this similarity is only valid under the assumptions made to derive Equation (2.31c). In the case that the Lewis number is not equal to one, or that the heat capacities are not equal for each species, the finite-rate solution will differ slightly from that obtained with the infinite-reaction rate.

2.3 Boundary conditions

The following boundary conditions are imposed for the resolution of the finite reaction rate system (Equations (2.22a) to (2.22d)) and for the flame sheet problem (Equations (2.31a) to (2.31c)),

$$\Gamma_{\mathrm{D}}: \quad \mathbf{u} = \mathbf{u}_{\mathrm{D}}, \;\; T = T_{\mathrm{D}}, \;\; Y_k, = Y_{k,\mathrm{D}}, \;\; z = z_{\mathrm{D}}, \tag{2.37a}$$

$$\Gamma_{\mathrm{DW}}: \quad \mathbf{u} = \mathbf{u}_{\mathrm{D}}, \;\; \nabla T \cdot \mathbf{n}_{\partial\Omega} = 0, \;\; \nabla Y_k \cdot \mathbf{n}_{\partial\Omega} = 0, \;\; \nabla z \cdot \mathbf{n}_{\partial\Omega} = 0, \tag{2.37b}$$

$$\Gamma_{\mathrm{N}}: \quad \left(-p\mathbf{I} + \left(\frac{\mu}{\mathrm{Re}}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I}\right)\right)\cdot\mathbf{n}_{\partial\Omega} = 0,$$
$$\nabla T \cdot \mathbf{n}_{\partial\Omega} = 0, \;\; \nabla Y_k \cdot \mathbf{n}_{\partial\Omega} = 0, \;\; \nabla z \cdot \mathbf{n}_{\partial\Omega} = 0, \tag{2.37c}$$

$$\Gamma_{\mathrm{ND}}: \quad \left(-p\mathbf{I} + \left(\frac{\mu}{\mathrm{Re}}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) - \frac{2}{3}\mu(\nabla\cdot\mathbf{u})\mathbf{I}\right)\right)\cdot\mathbf{n}_{\partial\Omega} = 0,$$
$$T = T_{\mathrm{D}}, \;\; Y_k = Y_{k,\mathrm{D}}, \;\; z = z_{\mathrm{D}} \tag{2.37d}$$

$$\Gamma_P: \quad \mathbf{u}(\mathbf{x}) = \mathbf{u}(\mathbf{x}'), \;\; T(\mathbf{x}) = T(\mathbf{x}'), \;\; Y_k(\mathbf{x}) = Y_k(\mathbf{x}'), \;\; z(\mathbf{x}) = z(\mathbf{x}'), \tag{2.37e}$$

where $k = (1, \ldots, N-1)$ denotes the index of mass fractions. The boundary Γ_{D} represents conditions for inlets and walls, with the velocity, temperature, mass fractions and mixture fraction defined as Dirichlet boundary conditions. Boundaries Γ_{DW} are used to represent adiabatic walls, where the velocity is given as a Dirichlet boundary condition again, but with the gradients perpendicular to the wall of the transported scalars are set to zero. The boundary Γ_{N} represent an outflow of the domain with homogeneous Neumann condition for all scalars. The boundary Γ_{ND} also represents an outlet boundary condition, but with Dirichlet boundary conditions for the scalars. Finally, the boundaries Γ_P are periodic, where $\mathbf{x}$ and $\mathbf{x}'$ are periodic pairs in the domain.

3 The Discontinuous Galerkin method

ch:NumericalMethods

This chapter aims to give an overview of the basic ideas of the DG method, as well as to present the spatial and temporal discretization of the equations presented earlier. Parts of this chapter are based on the work presented at (Kummer, 2017; Kikker et al., 2020; Smuda, 2021). The reader interested in a more in-depth description of the DG method is referred to the works of (Cockburn et al., 2000; Hesthaven and Warburton, 2008; Di Pietro and Ern, 2012)

3.1 State of the art

3.2 The Discontinuous Galerkin method

3.2.1 Definitions for the discretization

ssec:SpatDiscretization

First some standard definitions and notation are introduced in the context of DG methods.

A computational domain $\Omega \subset \mathbb{R}^2$ with a polygonal and simply connected boundary $\partial\Omega$ is defined. The numerical grid is then formed by the set of non-overlapping elements $\mathfrak{K}_h = \{K_1, ..., K_J\}$ with a characteristic mesh size h, so that Ω is the union of all elements, i.e. $\Omega = \bigcup_{i=1}^{J} K_i$.

Define $\Gamma = \bigcup_j \partial K_j$ as the union of all edges (internal edges and boundary edges) and $\Gamma_I = \Gamma \setminus \partial\Omega$ as the union of all interior edges. For each edge of Γ a normal field $\mathbf{n}_\Gamma$ is defined. Particularly on $\partial\Omega$ the normal field is defined as an outer normal and $\mathbf{n}_\Gamma = \mathbf{n}_{\partial\Omega}$. For each field $u \in C^0(\Omega \setminus \Gamma_I)$, u^- and u^+ is defined, which describe the values of the variables on the interior and exterior sides of the cell:

$$u^- = \lim_{\xi \searrow 0} u(\mathbf{x} - \xi \mathbf{n}_\Gamma) \quad \text{for } \mathbf{x} \in \Gamma \tag{3.1}$$

$$u^+ = \lim_{\xi \searrow 0} u(\mathbf{x} + \xi \mathbf{n}_\Gamma) \quad \text{for } \mathbf{x} \in \Gamma_I \tag{3.2}$$

The jump and mean values of u on the inner edges Γ_I are defined as

$$[\![u]\!] = u^+ - u^- \tag{3.3}$$

$$\{u\} = \frac{1}{2}(u^- + u^+). \tag{3.4}$$

while the jump and mean values on the boundary edges $\partial\Omega$ are:

$$[\![u]\!] = u^- \tag{3.5}$$

$$\{u\} = u^-. \tag{3.6}$$

Furthermore, the broken polynomial space of a total degree k is defined as

$$\mathbb{P}_k(\mathfrak{K}_h) = \{f \in L^2(\Omega); \forall K \in \mathfrak{K}_h : f|_K \text{ is polynomial and } \deg(f|_K) \leq k\}. \tag{3.7}$$ {Eq:PolSpace}

Additionally, for $u \in \mathcal{C}^1(\Omega \setminus \Gamma)$ the broken gradient $\nabla_h u$ is defined as:

$$\nabla_h u = \begin{cases} 0 & \text{on } \Gamma \\ \nabla u & \text{elsewhere} \end{cases} \tag{3.8}$$

The broken divergence $\nabla_h \cdot u$ is defined analogously. Furthermore, the function space for test and trial functions for D_v dependent variables is defined as

$$\mathbb{V}_{\mathbf{k}} = \prod_{i=1}^{D_v} \mathbb{P}_{k_i}(K_h) \tag{3.9}$$ {Eq:Vspace}

where $\mathbf{k} = (k_1, ..., k_{D_v})$. Additionally, for a cell K we define for u_K, $v_K \in \mathbb{V}_{\mathbf{k}}$ a local inner product and a local L^2-norm as

$$(u_K, v_K)_K \coloneqq \int_K u_K v_K \mathrm{d}x, \qquad \|u_K\|_K^2 \coloneqq (u_K, u_K)_K \tag{3.10}$$

Similarly, for u_h, $v_h \in \mathbb{V}_{\mathbf{k}}$ a global inner product and global broken norm are defined as

$$(u_h, v_h)_{\Omega_h} \coloneqq \sum_{i=1}^{N} (u_h, v_h)_K, \qquad \|u_h\|_{\Omega_h}^2 \coloneqq (u_h, u_h)_{\Omega_h} \tag{3.11}$$

3.2.2 Discretization using the DG Method

sec:DiscWithDG

In this subsection the discretization of a simple problem using the DG method will be shown in order to demonstrate how it works and some of its specific characteristics. For this purpose, the discretization of a general conservation law for a scalar quantity $u = u(\mathbf{x}, t)$ governed by a nonlinear flux function $\mathbf{f}(u)$ will be considered. In addition, suitable Dirichlet boundary conditions on $\partial\Omega = \partial\Omega_D$ and initial conditions u_0 are defined. The problem reads

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f}(u) = 0, \qquad \mathbf{x} \in \Omega, \tag{3.12a}$$ {eq:consEqDG}

$$u = u_D, \qquad \mathbf{x} \in \partial\Omega_D, \tag{3.12b}$$

$$u(\mathbf{x}, 0) = u_0(\mathbf{x}), \qquad \mathbf{x} \in \Omega. \tag{3.12c}$$

eqs:DGTransportExample

The DG-method allows finding an approximate solution $u_h = u_h(\mathbf{x}, t)$ for the problem defined by Equation (3.12) by forming a linear combination of polynomial functions in each cell. The discretization procedure starts by the approximation of the domain Ω with a numerical grid $\mathfrak{K}_h$. In each cell K_j of the numerical grid a set of polynomial basis $\boldsymbol{\phi}_j = (\tilde{\phi}_{j,l})_{l=1,...,N_k} \in \mathbb{P}_k(\mathcal{K}_h)$ with a local cell support $\mathrm{supp}(\phi_j) = \overline{K_j}$ is defined. This allows to represent the local solution for each cell K_j as

$$u_j(\mathbf{x}, t) = \sum_{l=1}^{N_k} \tilde{u}_{j,l}(t) \phi_{j,l}(\mathbf{x}) = \tilde{\mathbf{u}}_j(t) \cdot \boldsymbol{\phi}_j(\mathbf{x}) \tag{3.13}$$ {eq:DGAnsatz}

The coefficients $\tilde{\mathbf{u}}_j = (\tilde{u}_{j,l})_{l=1,\dots,N_k}$ are the degrees of freedom (DOFs) of the local solution in the cell K_j, which are the unknowns of the problem. Note the time dependence of the coefficients $\tilde{\mathbf{u}}_j$, as well as the dependence of the basis functions $\boldsymbol{\phi}_j(\mathbf{x})$ on the vector $\mathbf{x}$.

This approximate solution sought is the best approximation of $u \in L^2(\Omega)$, which gives a minimum global error in the approximation space $u \in \mathbb{P}_k(\Omega)$.

$$\int_\Omega \underbrace{(u_h(x) - u(x))}_{=:r(x)}{}^2 \,\mathrm{d}V = ||u_h - u||_2^2 \to \min \tag{3.14}$$

Here r is the error of the discretization. Minimization is equivalent to requiring

$$(r(x), \phi_m) = (f_h - f, \phi_m) \stackrel{!}{=} 0 \qquad \forall \phi_m \tag{3.15}$$ {eq:L2projection}

This means that the error is orthogonal to every polynomial function ϕ_m in the approximation space. The Bramble-Hilbert lemma (Bramble and Hilbert, 1970) says, that for a p-times differentiable variable u, the error is of the order $\mathcal{O}(h^{p+1})$, which is one of the major motivations for the use of high order methods. The differentiability assumption is essential. For non smooth u the well known Gibbs phenomenon occurs.

There are two general approaches used for the representation of the solution using basis functions: modal and nodal. Each one of them present some advantages and disadvantages (Hesthaven and Warburton, 2008). In this work, a modal polynomial representation is used. The basis functions are chosen such that they are orthogonal to each other

$$\int_{K_j} \phi_{j,m}\phi_{j,n}\mathrm{d}V = \delta_{mn} \tag{3.16}$$

where δ_{mn} is the Kronecker delta. In the present work Legendre polynomials are used, since they present the orthogonality property. This property implies that the mass matrix (to be defined later) equals the identity matrix.

By inserting the approximate solution defined by Equation (3.13) in the conservation Equation (3.12a), a local residual R_j can be defined

$$R_j(\mathbf{x}, t) = \frac{\partial u_j}{\partial t} + \nabla \cdot \mathbf{f}(u_j), \quad \mathbf{x} \in K_j \tag{3.17}$$ {eq:DGResidualeq}

Minimization of this local residual is done by multiplying Equation (3.17) by the so called tests functions. In the Galerkin approach, these test functions are required to be from the same space as the trial functions, i.e. $\vartheta_{j,l} = \phi_{j,l}$. Thus, by multiplying Equation (3.17) by a trial function and integrating over the cell K_j, one obtains

$$\int_{K_j} R_j \vartheta_{j,l} \,\mathrm{d}V = \int_{K_j} \frac{\partial u_j}{\partial t}\phi_{j,l} + \nabla \cdot \mathbf{f}(u_j)\phi_{j,l} \,\mathrm{d}V \stackrel{!}{=} 0, \quad \forall \boldsymbol{\phi}_{j,l}. \tag{3.18}$$ {eq:DGminimization}

Where the minimization comes from requiring the equality to zero. Note that until this point only a cell-local discretization has been addressed. The next step for obtaining a global DG formulation is to use integration by parts for rewriting the spatial term in Equation (3.18). This is done to make the boundary edge integrals explicitly appear in the formulation, which

are used later to couple cell K_j with neighbouring cells. The partial integration process results in

$$\int_{K_j} \frac{\partial u_j}{\partial t}\phi_{j,l}\,\mathrm{d}V + \oint_{\partial K_j} \left(\mathbf{f}(u_j)\cdot\mathbf{n}_j\right)\phi_{j,l}\,\mathrm{d}S - \int_{K_j} \mathbf{f}(u_j)\cdot\nabla_h\phi_{j,l}\,\mathrm{d}V = 0, \quad \forall\phi_{j,l}, \tag{3.19}$$ {eq:DGfluxFormulation}

Note that inserting the ansatz Equation (3.13) into Equation (3.19) is problematic, since ∂K_j is shared by other cells, and in the DG method continuity of a variable is not enforced across cell boundaries. This means that in general the inner value u_j^- and the outer value u_j^+ are not equal. This problem is solved by introducing the concept of a numerical flux function, denoted here with $\hat{f}$

$$\hat{f}(u_j^+, u_j^-, \mathbf{n}_\Gamma) \approx \mathbf{f}(u_j)\cdot\mathbf{n}_j. \tag{3.20}$$

This expression defines an unique value for the flux of a given cell boundary, enforcing flux continuity. The numerical flux $\hat{f}$ couples the DOFs of neighbouring cells, and should satisfy certain mathematical and physical properties which will be discussed later. Many different numerical fluxes have been developed, and it is an active area of investigation. They differ mainly in computational cost, stability and dissipation of the scheme.

By introducing the numerical flux in Equation (3.19) the problem now reads

$$\int_{K_j} \frac{\partial u_j}{\partial t}\phi_{j,l}\,\mathrm{d}V + \oint_{\partial K_j} \left(\hat{f}(u_j^+, u_j^-, \mathbf{n}_\Gamma)\right)\phi_{j,l}\,\mathrm{d}S - \int_{K_j} \mathbf{f}(u_j)\cdot\nabla_h\phi_{j,l}\,\mathrm{d}V = 0, \quad \forall\phi_{j,l}, \tag{3.21}$$ {eq:DGNumericalfluxFormulation}

The global formulation

Note that Equation (3.21) is still a local formulation. A global solution $u(\mathbf{x}, t)$ can be defined by a piecewise polynomial approximation according to

$$u(\mathbf{x},t) \approx u_h(\mathbf{x},t) = \bigoplus_{j=1}^{J} u_j(\mathbf{x},t) = \sum_{j=1}^{J}\sum_{l=1}^{N_k} \tilde{u}_{j,l}(t)\phi_{j,l}(\mathbf{x}) \in \mathbb{P}_k(\mathfrak{K}_h) \tag{3.22}$$ {eq:globalApprox}

which corresponds to the direct sum of the J local solutions u_j. A vector $\tilde{\mathbf{u}} = \tilde{u}_{1,1}, \tilde{u}_{1,2}, \ldots, \tilde{u}_{j,l}, \ldots, \tilde{u}_{J,N_k}$ which comprises all the DOFs of the global approximation u_h is defined, and is of length $N = J \cdot N_k$.

Finally, the global formulation is obtained by inserting the ansatz Equation (3.13) into Equation (3.21), summing over all cells K_j and making use of Equation (3.22). The problem reads: Find $u_h \in \mathbb{P}_k(\mathfrak{K}_h)$, such that $\forall\phi \in \mathbb{P}_k(\mathfrak{K}_h)$

$$\int_\Omega \frac{\partial u_h}{\partial t}\phi\,\mathrm{d}V + \oint_\Gamma \hat{f}(u_h^+, u_h^-, \mathbf{n}_\Gamma)\,[\![\phi]\!]\,\mathrm{d}S - \int_\Omega \mathbf{f}(u_h)\cdot\nabla_h\phi\,\mathrm{d}V = 0, \tag{3.23}$$ {eq:semiDiscWeakForm}

The solution of this system requires finding the DOFs $\tilde{\mathbf{u}}$ of the global approximation u_h. Dirichlet boundary conditions are included in the formulation by defining at Γ_D the outer value $u_h^- = u_D$.

Note that Equation (3.23) is semi-discrete, meaning that the system of equations has been discretized in space, but not in time. Time discretization will be treated in Section 3.2.3.

Finally, after selecting suitable numerical fluxes for the various terms of the governing equations, a system is obtained that in general has the form

$$\frac{\mathrm{d}\tilde{\mathbf{u}}}{\mathrm{d}t} + \mathbf{Op}(\tilde{\mathbf{u}}) = \mathbf{b}, \tag{3.24}$$ {eq:discretMatrixForm}

Where $\mathbf{M}$ is the mass matrix, and $\mathbf{Op}$ is the operator matrix. The vector b contains the Dirichlet boundary condition. The operator matrix is defined locally by

$$(\mathbf{Op}_j)_{m,n} = \oint_{\partial K_j} \hat{f}(\tilde{u}_{j,n}, \tilde{u}_{j^*,n}, \mathbf{n}_\mathfrak{I})\phi_{j,m}\,\mathrm{d}S - \int_{K_j} \mathbf{f}(\tilde{u}_{j,n}\phi_{j,n}) \cdot \nabla_h \phi_{j,m}\,\mathrm{d}V, \tag{3.25}$$ {eq:genOpMatrix}

with j^* denoting the index of a neighbour cell. The matrix $\mathbf{Op}$ has block-diagonal structure, but also including extra diagonals which relate the DOFs of the cell with the neighbouring cells.

The global mass matrix is

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{M}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{M}_J \end{bmatrix}, \tag{3.26}$$

which is block-diagonal, since $\mathbf{M}_j$ does not depend on neighbouring cells.

$$(\mathbf{M}_j)_{m,n} = \int_{K_j} \phi_{j,m}\phi_{j,n}\,\mathrm{d}V \tag{3.27}$$ {eq:massMatrix}

The mass matrix of a cell $\mathbf{M}_j := \underline{\underline{M}}_{(j,-)\ (j,-)}$ only depends in this case on the cell-local basis functions.

Note on the numerical fluxes

As mentioned before, the numerical fluxes $\hat{f}$ have to fulfil certain physical and mathematical properties for obtaining a stable and convergent method. One of the requirement for proving the stability of the scheme is the monotonicity. First, the energy estimate is defined as

$$\|u(\mathbf{x}, t)\|_\Omega^2 \leq \|u(\mathbf{x}, 0)\|_\Omega^2, \quad \forall t \geq 0, \tag{3.28}$$ {eq:energyEstimate}

assuming homogeneous Dirichlet boundary conditions. This means that the system is stable if the energy norm $\|u(\mathbf{x}, t)\|_\Omega^2$ is strictly decreasing in the absence of inflow.

For a monotonic numerical flux is possible to show that the discrete problem Equation (3.23) satisfies the discrete equivalent of Equation (3.28). In addition to the monotonicity of the numerical flux, it is necessary that $\hat{f}$ is Lipschitz continuous.For the complete proof the interested reader is referred to the book from Di Pietro and Ern (2012).

Two more requirements are needed for the numerical flux. The first is the consistency of the flux, which can be written as

$$\hat{f}(a, a, \mathbf{n}) = \mathbf{f}(a) \cdot \mathbf{n}, \quad \forall a \in \mathbb{R}. \tag{3.29}$$ {eq:consistency}

showing that a numerical flux function should deliver the same approximate solution that the original flux function in case of a continuous variable across the interface. A direct consequence of the consistency of the numerical flux is that the weak formulation Equation (3.23) is automatically fulfilled by $u_h = u$.

Finally the last requirement is that the numerical flux should be conservative, which means that the total amount of u can only change due to fluxes across the domain boundary. This can be written as

$$\hat{f}(a,b,\mathbf{n}) = -\hat{f}(b,a,-\mathbf{n}), \quad \forall a,b \in \mathbb{R}. \tag{3.30}$$ {eq:conservativity}

All numerical fluxes used in this work for the spatial discretization of equation 2.22 fulfil these requirements and they will be shown in the next section.

3.2.3 Temporal discretization

ssec:TemporalDiscretization

This section will give a brief introduction to the most popular time-stepping techniques and then show the time-stepping method used in this work. This section is mainly based on LeVeque (2002) and Ferziger and Perić (2002). In the previous chapter the spatial discretization using a DG method was shown, resulting in the semi-discrete formulation given by Equation (3.23). The time discretization of this semi-discrete system leads to the so called method of lines, which is the name for a method first discretized in space, and later in time.

An alternative to the method of lines is the so called Rothe's method, where time is first discretized then the space. This can be advantageous in some cases, such as problems with a moving domain. Another alternative is the space-time approach, where basically the temporal coordinate is treated as another spatial dimension. Again, the method can be very attractive for some cases. However, the discretized schemes lead often to prohibitively large systems. These approaches are ignored in the present work, and the method of lines is adopted.

First the time discretization for a system with a constant mass matrix will be discussed. The process of discretization results often in a system of ordinary differential equations (ODEs) of the form

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = -\mathbf{F}(t,\mathbf{u}(t)) \qquad \text{for} \qquad t \in (0,T). \tag{3.31}$$ {eq:timestep1}

with a initial condition given by $\mathbf{u}(t=0) = \mathbf{u}^0$. The main idea of a time-stepper algorithm is to discretize the time coordinate, and advance gradually the solution $\mathbf{u}(t^n)$ in time from values at previous time levels $\mathbf{u}(t^{n-1})$, $\mathbf{u}(t^{n-2})$, ..., until a certain final time $t = T$ is reached. By integrating Equation (3.31) in time, one obtains

$$\mathbf{M}(\mathbf{u}(t^{n+1}) - \mathbf{u}(t^n)) = -\int_{t^n}^{t^{n+1}} \mathbf{F}(t,\mathbf{u}(t))\mathrm{d}t \tag{3.32}$$ {eq:timestep2}

This equation is the starting point for different class of time stepping techniques. Two kind of methods can be distinguished, depending on how the integral in Equation (3.32) is evaluated: explicit methods and explicit methods. Explicit methods are obtained when the approximation of the integral is done only by using information from old time steps, while for implicit methods the information from the actual timestep is also considered, making necessary the solution of a system of equations.

The simplest example of an explicit time-stepping method is the Explicit Euler Method:

$$\mathbf{M}\mathbf{u}(t^{n+1}) = \mathbf{M}\mathbf{u}(t^n) - \Delta t\mathbf{F}(t^n,\mathbf{u}(t^n)), \tag{3.33}$$

which is first order accurate in time. Other explicit methods exists with better properties than the Explicit Euler Method, typically using information from multiple known time levels or a interpolation of them. Adams-Bashforth methods are a good exponent of them.

Due to the local nature of the approach, explicit methods present themselves specially attractive for DG methods, particularly for hyperbolic equations. Explicit methods are relatively easy to implement, and need considerably less storage compared to implicit methods. However, explicit methods experience the disadvantage that the stability of the algorithm is heavily limited by a maximal timestep size Δt. The timestep typically scales with the grid size h and polynomial degree k by $\Delta \sim h/k$ for hyperbolic and $\Delta \sim (h/k)^2$ for parabolic problems (Gassner et al., 2007). For many problems of interest, particularly stiff systems, this limitation is highly restrictive, since very little timesteps need to be chosen in order to obtain a stable method.

Implicit methods on the other hand are specially well suited for stiff problems, as they don't suffer from the restrictive timestep limitation of explicit methods, even not being restricted at all under certain conditions. This allows using considerably bigger timesteps. Implicit methods present however the inconvenience that they require the solution of a system of equations, which for large problems is not a trivial task and specialized methods are needed. The use of implicit methods is justified, particularly for stiff problems, as the extra computational overhead originating from the resolution of the system of equations is usually smaller than the time it would take to solve the same problem using explicit schemes with very small timesteps.

The simplest implicit method is the Implicit Euler Method

$$\mathbf{M}\mathbf{u}(t^{n+1}) + \Delta t \mathbf{F}(t^{n+1}, \mathbf{u}(t^{n+1})) = \mathbf{M}\mathbf{u}(t^n) \tag{3.34}$$

Note that this is a non-linear system of algebraic equations that has to be solved for $\mathbf{u}(t^{n+1})$ iteratively. The Implicit Euler Methods is the first method of a family of backward differentiation formulas (BDF), and presents the property of being unconditionally stable, meaning that the algorithm allows an arbitrarily large timestep. This property allows the calculation of steady state solutions just by choosing a very big Δt value. However, not all BDF schemes are unconditionally stable, as it will be shown next.

Backward Differentiation Formula

In the present work BDF methods are used. In case of a non-constant mass matrix $\mathbf{M}$, they have a general formula given by

$$\frac{\beta_0}{\gamma \Delta t}\mathbf{M}(\mathbf{u}(t^n))\mathbf{u}(t^n) - \mathbf{F}(\mathbf{u}(t^n)) = -\sum_{i=1}^{s} \frac{\beta_i}{\gamma \Delta t}\mathbf{M}(\mathbf{u}(t^{n-i}))\mathbf{u}(t^{n-i}). \tag{3.35}$$

eq:BDFDiscretization

where s is the order of the BDF-scheme. The coefficients of each schema are shown in Table 3.1. The main advantage of BDF methods is their large stability regions, which make them suitable for solving stiff problems. In Figure 3.1 the stability regions for the first four BDF schemes

s	γ	β_0	β_1	β_2	β_3	β_4
Implicit Euler (BDF1)	1	1	-1			
BDF2	2	3	-4	1		
BDF3	6	11	-18	9	-2	
BDF4	12	25	-48	36	-16	3

Table 3.1: Coefficients of the BDF schemes.

tab:BDFCoeff

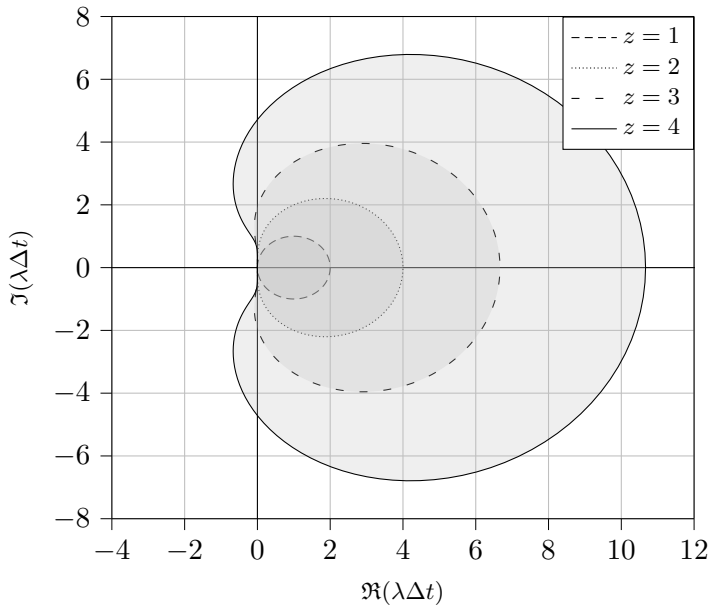


Figure 3.1: A-stability regions of the BDF schemes for different z. The areas shown in grey are unstable regions. Figure adapted from (Kikker, 2020)
Fig:AStability

are shown. It is possible to observe that only the BDF-1 (implicit Euler) and BDF-2 schemes exhibit the property that they are A-stable, which means that the stability region contains the entire left complex plane (Dahlquist, 1963). On the other hand, BDF schemes of order $s > 2$ are not A-stable. In this work however, BDF schemes up to order three have been used, as the unstable eigenvalues for $z = 3$ are comparatively small (Smuda, 2021). The solution of problems of the form of Section 3.2.3 will be treated in Chapter 4.

Special attention should be put into the discretization of the temporal derivative appearing in the continuity equation, Equation (2.22a). It has been observed that the treating this term as a source term, for cases where high density ratios are present, is problematic (Cook and Riley, 1996; Nicoud, 2000). Cook and Riley (1996) reported for a pressure projection method, and using a third-order Adams-Bashforth scheme, that the discretization of the $\partial\rho/\partial t$ is a source of instabilities. They used a second-order explicit approximation

$$\left(\frac{\partial \rho}{\partial t}\right)^n = \frac{1}{2\Delta t}\left(3\rho^n - 2\rho^{n-1} + \rho^{n-2}\right) \tag{3.36}$$
{eq:DiscretizationDrhoDT}

which is reported to be much more stable than a third-order approximation, by arguing that the extra dissipation from even-ordered schemas, compared to the dispersive effects of odd-ordered schemas, is helpful for the stability of the algorithm. However, they found that even with even-ordered time approximations of the density time derivative, the algorithm is stable only to maximum density variations up to a factor of three. The approximation of the time derivative given by Equation (3.36)is used in this work. Some comments in this will be done later in Chapter 5

3.3 Discontinuous Galerkin discretization of the low-Mach equations

The democratization methodology shown in last section is used for finding a discrete formulation of the governing equations for low-Mach reactive flows. In the next sections the discretization for the fully coupled problem with finite reaction rate, and for the flame sheet problem are shown. The chosen numerical fluxes are also shown, and some of their particularities are discussed.

In order to ensure the validity of the Ladyzenskaja-Babuška-Brezzi (or inf-sup) condition, (see Babuška (1973)) a mixed order formulation is used in all calculations, where polynomials of order k for velocity, temperature, mass fractions and mixture fractions, and of degree $k' = k - 1$ for pressure are used. This is a required compatibility condition for obtaining a well posed problem.

3.3.1 Discontinuous Galerkin discretization of the finite reaction rate problem

Here the DG discretization of the finite reaction system defined by Equations (2.22a) to (2.22d) is presented. First, the vector $\mathbf{Y}' = (Y_1, \ldots, Y_{N-1})$ is defined as the vector containing the first $(N-1)$ mass fractions and $\mathbf{s} = (s_1, \ldots, s_{N-1})$ as the vector containing the test functions for the first $(N-1)$ mass fraction equations.

The discretized form of Equations (2.22a) to (2.22d) is obtained in a similar fashion to the methodology shown in Section 3.2.2. This means, each equation is multiplied by a test function, integrated it over an element K, applying integration by parts, using an adequate numerical flux for each term and summing over all cells in order to obtain a global formulation. Note that the convective and diffusive terms of the temperature scalars T, mass fraction Y_α and mixture fraction z have the same form, so they share the same expression in their discretized form.

Finally, the discretized problem can be written as: find the numerical solution $(p_h, \mathbf{u}_h, T_h, \mathbf{Y}'_h) \in \mathbb{V}_\mathbf{k}$ such that for all test functions $(q_h, \mathbf{v}_h, r_h, \mathbf{s}_h) \in \mathbb{V}_\mathbf{k}$:

$$\mathcal{B}^1(q_h) = \mathcal{C}\left(\mathbf{u}_h, q_h, \rho(T_h, \mathbf{Y}_h)\right) + \mathcal{T}(\partial_t \rho|_{t^{n+1}}, q_h), \tag{3.37a}$$

{DiscretizedConti}

$$\begin{aligned}\mathcal{B}^2(\mathbf{v}_h) = {} & \mathcal{U}^C\left(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h, \rho(T_h, \mathbf{Y}_h)\right) + \mathcal{U}^P\left(p_h, \mathbf{v}_h\right) + \mathcal{U}^D\left(\mathbf{u}_h, \mathbf{v}_h, \mu(T_h)\right) \\ & + \mathcal{U}^S\left(\rho(T_h, \mathbf{Y}_h), \mathbf{v}_h\right),\end{aligned} \tag{3.37b}$$

{DiscretizedMomentum}

$$\begin{aligned}\mathcal{B}^3(r_h) = {} & \mathcal{S}^C\left(\mathbf{u}_h, T_h, r_h, \rho(T_h, \mathbf{Y}_h)\right) + \mathcal{S}^{D,E}\left(T_h, r_h, k/c_p(T_h)\right) \\ & + \mathcal{S}^S\left(r_h, \mathrm{Q}(T_h, \mathbf{Y}_h), \omega(T_h, \mathbf{Y}_h), c_p(T_h, \mathbf{Y}_h)\right),\end{aligned} \tag{3.37c}$$

{DiscretizedEnergy}

$$\begin{aligned}\mathcal{B}^4(s_{\alpha h}) = {} & \mathcal{S}^C\left(\mathbf{u}_h, Y_{\alpha h}, s_{\alpha h}, \rho(T_h, \mathbf{Y}_h)\right) + \mathcal{S}^{D,M}\left(Y_{\alpha h}, s_{\alpha h}, \rho D_\alpha(T_h)\right) \\ & + \mathcal{M}^S_\alpha\left(s_{\alpha h}, \omega(T_h, \mathbf{Y}_h)\right).\end{aligned} \tag{3.37d}$$

{DiscretizedMassFractions}

eqs:variatProblemFull

Add the temporal terms

where the index α takes values $\alpha = 1, \ldots, (N-1)$. Each one of the forms introduced here are shown later in Section 3.3.3.

3.3.2 Discontinuous Galerkin discretization of the flame sheet problem

Discretizing the flame sheet problem given by Equations (2.31a) to (2.31c) proceeds in a similar way. Due to the similarity of the mass fraction equation and the mixture fraction equation the discretization is analogous.

The resulting problem reads: find the numerical solution $(p_h, \mathbf{u}_h, z_h) \in \mathbb{V}_\mathbf{k}$ such that for all test functions $(q_h, \mathbf{v}_h, r_h) \in \mathbb{V}_\mathbf{k}$ we have:

{DiscretizedConti2}
$$\mathcal{B}^1(q_h) = \mathcal{C}\left(\mathbf{u}_h, q_h, \rho(z_h)\right), \tag{3.38a}$$

{DiscretizedMomentum2}
$$\mathcal{B}^2(\mathbf{v}_h) = \mathcal{U}^C\left(\mathbf{w}_h, \mathbf{u}_h, \mathbf{v}_h, \rho(z_h)\right) + \mathcal{U}^P\left(p_h, \mathbf{v}_h\right) + \mathcal{U}^D\left(\mathbf{u}_h, \mathbf{v}_h, \mu(z_h)\right) + \mathcal{U}^S\left(\rho(z), \mathbf{v}_h\right), \tag{3.38b}$$

{DiscretizedEnergy2}
$$\mathcal{B}^3(r_h) = S^C\left(\mathbf{u}_h, z_h, r_h, \rho(z_h)\right) + \mathcal{S}^{D,E}\left(z_h, r_h, \rho D(z_h)\right). \tag{3.38c}$$

eqs:variatFS

Note that density and transport parameters are dependent on the mixture fraction z. The evaluation of those parameters is done as mentioned in Section 2.2 and solved iteratively using a Newton-Dogleg type method as shown later in Section 4.2.2

3.3.3 Definitions of nonlinear forms

ssec:nonLinearforms

In the following the nonlinear forms used in this work are shown. Regarding the choice of fluxes, the "best practices" known in literature for the incompressible Navier-Stokes equation are followed. These fluxes proved to be well suited for all the problems discussed in this thesis, providing stability to the algorithm, while maintaining the accuracy of the solver.

It is well known (Pietro and Ern, 2012; Girault et al., 2004) that central difference fluxes for the pressure gradient and velocity divergence, combined with a coercive form for the viscous terms, e.g. symmetric interior penalty, gives a stable discretization for the Stokes equation. Furthermore, it is known that for all kinds of convective terms, a numerical flux which transports information in characteristic direction, e.g. Upwind, Lax-Friedrichs or Local-Lax-Friedrichs, must be used. We opted for the last one in the present implementation, as it offers a good compromise between accuracy and stability.

Continuity equation A central difference flux for the discretization of the continuity equation is used:

{eq:Conti}
$$\mathcal{C}(\mathbf{u}, q, \rho) = \oint_{\Gamma_\mathrm{I} \cup \Gamma_\mathrm{N} \cup \Gamma_\mathrm{ND} \cup \Gamma_\mathrm{P}} \{\rho \mathbf{u}\} \cdot \mathbf{n}_\Gamma \left[\!\left[q\right]\!\right] \,\mathrm{d}S - \int_\Omega \rho \mathbf{u} \cdot \nabla_h q \,\mathrm{d}V. \tag{3.39}$$

The density in Equation (3.39) is evaluated as a function of the temperature and mass fractions using the equation of state (Equation (2.26)). The term $\mathcal{B}^1$ on the left hand sides of Equation (3.37a) and Equation (3.38a) contains the Dirichlet boundary conditions:

$$\mathcal{B}^1(q) = -\oint_{\Gamma_\mathrm{D} \cup \Gamma_\mathrm{DW}} q(\rho_\mathrm{D} \mathbf{u}_\mathrm{D} \cdot \mathbf{n}_\Gamma). \,\mathrm{d}S \tag{3.40}$$

The density at the boundary ρ_D is evaluated with Equation (2.26) using the corresponding Dirichlet values of temperature and mass fractions.

$$\mathcal{T}() = \int_\Omega ?q \,\mathrm{d}V. \tag{3.41}$$

Add the source term for the continuity equation

Momentum equations The convective term of the momentum equations is discretized using a Lax-Friedrichs flux

$$\mathcal{U}^C(\mathbf{w},\mathbf{u},\mathbf{v},\rho) = \oint_\Gamma \left(\{\rho\mathbf{u}\otimes\mathbf{w}\}\,\mathbf{n}_\Gamma + \frac{\gamma_1}{2}[\![\mathbf{u}]\!]\right)\cdot[\![\mathbf{v}]\!]\,\mathrm{d}S - \int_\Omega (\rho\mathbf{u}\otimes\mathbf{w}) : \nabla_h\mathbf{v}dV. \tag{3.42}$$

{eq:Mom_convective}

The Lax-Friedrichs parameter γ_1 is calculated as Klein et al. (2016)

$$\gamma_1 = \max\left\{2\overline{\rho^+}|\overline{\mathbf{u}^+}\cdot\mathbf{n}^+|, 2\overline{\rho^-}|\overline{\mathbf{u}^-}\cdot\mathbf{n}^-|\right\}, \tag{3.43}$$

{eq:vardens_lambda}

where $\overline{\rho_h^\pm}$ and $\overline{\mathbf{u}^\pm}$ are the mean values of $\rho^\pm$ and $\mathbf{u}^\pm$ in $K^\pm$, respectively.
The pressure term is discretized by using a central difference flux

$$\mathcal{U}^P(p,\mathbf{v}) = \oint_{\Gamma\setminus\Gamma_N\setminus\Gamma_{ND}} \{p\}\,([\![\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma)\,\mathrm{d}S - \int_\Omega p\nabla_h\cdot\mathbf{v}\,\mathrm{d}V. \tag{3.44}$$

{eq:Mom_pressure}

The diffusive term of the momentum equations is discretized using an Symmetric Interior Penalty (SIP) formulation (Shahbazi, 2005)

$$\begin{aligned}\tilde{\mathcal{U}}^D(\mathbf{u},\mathbf{v},\mu) = &\int_\Omega \left(\mu\left((\nabla_h\mathbf{u}) + (\nabla_h\mathbf{u})^T - \frac{2}{3}(\nabla_h\cdot\mathbf{u})\boldsymbol{I}\right)\right) : \nabla_h\mathbf{v}\,\mathrm{d}V\\ &- \oint_{\Gamma\setminus\Gamma_N\setminus\Gamma_{ND}} \left(\left\{\mu(\nabla_h\mathbf{u} + \nabla_h\mathbf{u}^T - \frac{2}{3}(\nabla_h\cdot\mathbf{u})\boldsymbol{I})\right\}\mathbf{n}_\Gamma\right)\cdot[\![\mathbf{v}]\!]\,\mathrm{d}S\\ &- \oint_{\Gamma\setminus\Gamma_N\setminus\Gamma_{ND}} \left(\left\{\mu(\nabla_h\mathbf{v} + \nabla_h\mathbf{v}^T - \frac{2}{3}(\nabla_h\cdot\mathbf{v})\boldsymbol{I})\right\}\mathbf{n}_\Gamma\right)\cdot[\![\mathbf{u}]\!]\,\mathrm{d}S\\ &+ \oint_{\Gamma\setminus\Gamma_N\setminus\Gamma_{ND}} \eta\mu_{\max}[\![\mathbf{u}]\!][\![\mathbf{v}]\!]\,\mathrm{d}S.\end{aligned} \tag{3.45}$$

{eq:Mom_diffusive}

The viscosity μ is evaluated as a function of temperature according to Equation (2.29) and $\mu_{\max} = \max(\mu^+,\mu^-)$. Additionally η is the penalty term of the SIP formulation, which has to be chosen big enough to ensure coercivity of the form, but also as small as possible in order to not increase the condition number of the problem. The estimation of the penalty term is based on an expression of the form

$$\eta = \eta_0\frac{A(\partial K)}{V(K)}, \tag{3.46}$$

eq:PenaltyFactor

where for a two-dimensional problem A is the perimeter and V the area of the element. Parameter η_0 is a safety factor. If not stated otherwise, the value $\eta_0 = 4$ is set in all calculations. Further information on the determination of the penalty term of the SIP formulation η and the penalty term of the Lax-Friedrichs γ_1 can be found in the works from Hesthaven and Warburton (2008) and Hillewaert (2013).

Note that the diffusive term of the momentum equations is scaled by the Reynolds number, obtaining finally

$$\mathcal{U}^D(\mathbf{u},\mathbf{v},\mu) = \frac{1}{\mathrm{Re}}\tilde{\mathcal{U}}^D(\mathbf{u},\mathbf{v},\mu) \tag{3.47}$$

The source term arising due to body forces is:

$$\mathcal{U}^S(\rho,\mathbf{v}) = \frac{1}{\mathrm{Fr}^2}\int_\Omega \rho\frac{\mathbf{g}}{\|\mathbf{g}\|}\cdot\mathbf{v}\,\mathrm{d}V. \tag{3.48}$$

{eq:Mom_source}

Finally, the right hand sides of Equation (3.37b) and Equation (3.38b) contain the information from Dirichlet boundary conditions:

$$\mathcal{B}^2(\mathbf{v}) = -\oint_{\Gamma_\mathrm{D}} \left((\rho \mathbf{u}_\mathrm{D} \otimes \mathbf{u}_\mathrm{D})\mathbf{n}_\Gamma + \frac{\gamma_1}{2}\mathbf{u}_\mathrm{D}\right) \cdot \mathbf{v}\,\mathrm{d}S + \oint_{\Gamma_\mathrm{D}} \mu_\mathrm{D}\mathbf{u}_\mathrm{D} \cdot (\nabla_h \mathbf{v}\mathbf{n}_\Gamma + \nabla_h \mathbf{v}^T \mathbf{n}_\Gamma - \eta\mathbf{v})\,\mathrm{d}S. \tag{3.49}$$

The Dirichlet viscosity value μ_D is calculated from Equation (2.29) using the Dirichlet values of the temperature at the boundary.

Scalar equations Since the convective and diffusive term for the temperature, mass fractions and mixture fraction share a similar form, here their discretized expressions are summarized in terms of an arbitrary scalar X (corresponding to T in the energy equation, Y_α in the equation for species α and z for the mixture fraction equation) and transport parameter ξ (i.e. k/c_p in the energy equation, and (ρD) for the mass fraction and mixture fraction equations). The convective term of the scalars is discretized using a Lax-Friedrichs flux

$$\mathcal{S}^C(\mathbf{u}, X, r, \rho) = \oint_\Gamma \left(\{\rho\mathbf{u}X\} \cdot \mathbf{n} + \frac{\gamma_2}{2}[\![X]\!]\right)[\![r]\!]\,\mathrm{d}S - \int_\Omega (\rho\mathbf{u}X \cdot \nabla_h r)dV. \tag{3.50}$$

{eq:scalar_convective}

The Lax-Friedrichs parameter γ_2 is calculated as

$$\gamma_2 = \max\left\{\overline{\rho^+}|\overline{\mathbf{u}^+} \cdot \mathbf{n}^+|, \overline{\rho^-}|\overline{\mathbf{u}^-} \cdot \mathbf{n}^-|\right\}. \tag{3.51}$$

{eq:vardens_lambda2}

The diffusion term of scalars is discretized again with a SIP formulation:

$$\begin{aligned}\mathcal{S}^D(X, r, \xi) = &\int_\Omega (\xi\nabla_h X \cdot \nabla_h r)\,\mathrm{d}V \\ &- \oint_{\Gamma\setminus\Gamma_\mathrm{N}\setminus\Gamma_\mathrm{ND}} (\{\xi\nabla_h X\} \cdot \mathbf{n}[\![r]\!] + \{\xi\nabla_h r\} \cdot \mathbf{n}[\![X]\!] - \eta\xi_{\max}[\![X]\!][\![r]\!])\,\mathrm{d}S.\end{aligned} \tag{3.52}$$

{eq:Temp_diffusive}

The transport parameter ξ is calculated as a function of temperature using Equation (2.29) and $\xi_{\max} = \max(\xi^+, \xi^-)$. The diffusive term for the temperature equation and mixture fraction equation is scaled by the Reynolds and Prandtl number as

$$\mathcal{S}^{D,E}(T, r, k/c_p) = \frac{1}{\mathrm{Re\ Pr}}\mathcal{S}^D(T, r, k/c_p) \tag{3.53}$$

Similarly the diffusive term for the mass fraction equations is

$$\mathcal{S}^{D,M}(Y_{\alpha h}, s_{\alpha h}, \rho D_\alpha) = \frac{1}{\mathrm{Re\ Pr\ Le}_\alpha}\mathcal{S}^D(Y_{\alpha h}, s_{\alpha h}, \rho D_\alpha) \tag{3.54}$$

The boundary condition term of the corresponding scalar equation is:

$$\mathcal{B}^3(r) = -\oint_{\Gamma_\mathrm{D}\cup\Gamma_\mathrm{ND}} \left((\rho_D \mathbf{u}_D X_D) \cdot \mathbf{n}_\Gamma + \frac{\gamma_2}{2}X_D\right) r\,\mathrm{d}S + \oint_{\Gamma_\mathrm{D}\cup\Gamma_\mathrm{ND}} \xi_D X_D(\nabla_h r \cdot \mathbf{n}_\Gamma - \eta r)\,\mathrm{d}S. \tag{3.55}$$

Here, X_D is the Dirichlet value of the scalar X on boundaries and ξ_D is the corresponding transport parameter calculated,which is calculated with Equation (2.29) using the Dirichlet

values of the temperature at the boundary. Finally, the volumetric source term of the energy and mass fraction equations are defined as follows:

$$\mathcal{S}^S(r, \mathrm{Q}, \omega, c_p) = \mathrm{H}\ \mathrm{Da} \int_\Omega \frac{\mathrm{Q}\omega}{c_p} r \,\mathrm{d}V, \tag{3.56}$$

$$\mathcal{M}_\alpha^S(s_\alpha, \omega) = \mathrm{Da} \int_\Omega \nu_\alpha M_\alpha \omega s_\alpha \,\mathrm{d}V. \tag{3.57}$$

The heat release Q is calculated with Equation (2.25), the reaction rate ω is evaluated using Equation (2.24) and the mixture heat capacity with Equation (2.28).

4 Computational methodology

sec:CompMethodology

This section has the objective of introducing the computational strategies used in the present work for the solution of low-Mach number flows. Parts of this section are based on the work published by Kikker et al. (2020) and Gutiérrez-Jorquera and Kummer (2022).

BoSSS also features a method for solving highly nonlinear problems with a homotopy strategy. Further details on the used Newton method solver, the homotopy strategy and its implementation are given in the next sections, which are adapted from Kikker et al. (2020). For information on the mentioned orthonormalization multigrid algorithm we refer the interested reader to the work of Kummer et al. (2021). First, the globalized Newton method is presented that allows the resolution of a nonlinear system of equations. In addition, comments on the termination criteria are given.

The newton algorithm presented in this section is a collaborative work of the *BoSSS* code developers group and has also been used for the simulation of viscoelastic (Kikker et al., 2020) flows, among others.

In the early stages of development of the XNSEC solver the BoSSS code featured a framework for the solution of nonlinear systems using Picard iterations. These proved to be useful for systems where no large nonlinearities exist. Although the use of Picard iterations was useful for solving various types of problems (particularly problems involving incompressible flows), the method did not prove to be a particularly robust method, since it requires user-defined under-relaxation parameters in order to obtain a stable algorithm. These parameters are highly dependent on the problem at hand and require user experience to be adequately chosen. This motivated the development of the implementation of a Newton method for the resolution of the non-linear system. The use of a newton method proved to be very successful for all testcases treated in the present work.

We note however that this globalization strategy is still not sufficient to ensure convergence for some of the test cases presented, namely for high Rayleigh numbers for the differentially heated cavity problem. For those cases we use a homotopy strategy, where we start with a low homotopy-parameter, a parameter for which the solution of the problem is not hard to find, which is gradually and carefully increased until convergence for the aimed value is reached (cf. Section 4.3).

The solution algorithm developed for this work consists in various ingredients

Newton solver Efficient Calculation of the Jacobian matrix Homotopy strategy Initialization strategy for steady state combustion problems.

4.1 Solver structure

TODO: Steady state calculations are obtained by using a Implicit Euler timesteping scheme. Since the scheme is unconditionally stable (see Section 3.2.3), it is possible to use a very large timestep to obtain the steady state solution. In particular, a value $\Delta t = 1.7976931348623157E\cdot$

10^{304} is used, which is four orders of magnitude lower than the largest possible value of a double-precision floating-point number.

4.2 Solution of the nonlinear problem

The variational problem defined by Equations (3.37a) to (3.37d) can be cast into a more compact notation. By subtracting all terms from the right-hand sides from the terms of the left-hand sides of Equations (3.37a) to (3.37d) the problem can be written as: Find $\mathbf{U}_h \in \mathbb{V}_\mathbf{k}$

$$\mathcal{N}(\mathbf{U}_h, \mathbf{V}_h) = 0 \quad \forall\, \mathbf{V}_h \in \mathbb{V}_\mathbf{k}, \tag{4.1}$$ {eq:CompactVariational}

for $\mathbf{U}_h = (p_h, \mathbf{u}_h, T_h, \mathbf{Y}'_h)$ and $\mathbf{V}_h = (q_h, \mathbf{v}_h, r_h, \mathbf{s}_h)$. A basis is assumed as $\underline{\mathbf{\Phi}} = (\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_L)$ of $\mathbb{V}_\mathbf{k}$, written as a row vector, with $L \coloneqq \dim(\mathbb{V}_\mathbf{k})$. Then $\mathbf{U}_h$ can be represented as $\mathbf{U}_h = \underline{\mathbf{\Phi}} \cdot \mathbf{U}$. The nonlinear problem (4.1) can then be expressed as

$$\mathcal{A}(\mathbf{U}) = 0, \tag{4.2}$$ {Eq:nonLinSystem}

with the nonlinear function $\mathbb{R}^L \ni \mathbf{U} \mapsto \mathcal{A}(\mathbf{U}) \in \mathbb{R}^L$. The i-th component of $\mathcal{A}(\mathbf{U})$, can be defined by $\mathcal{N}(-,-)$ through the relation $[\mathcal{A}(\mathbf{U})]_i = \mathcal{N}(\underline{\mathbf{\Phi}} \cdot \mathbf{U}, \mathbf{\Phi}_i)$.

4.2.1 Newton's method

Newton's method is a very popular and well known iterative method used for finding roots of nonlinear systems. The method is particularly attractive because under certain conditions it can exhibit quadratic convergence (Deuflhard, 2011). In this section the method will be briefly described. For more information the interested reader is referred to the textbook from Kelley (1995).

Consider the linearization of Equation (4.2) around $\mathbf{U}_n$,

$$\mathcal{A}(\mathbf{U}_n) + \partial\mathcal{A}(\mathbf{U}_n) \underbrace{(\mathbf{U}_{n+1} - \mathbf{U}_n)}_{=:\mathbf{s}'_n} = 0. \tag{4.3}$$ {eq:LinearizedSys}

Here is $\partial\mathcal{A}$ the Jacobian matrix of $\mathcal{A}$, defined as

$$\partial\mathcal{A}_{ij}(\mathbf{U}) \coloneqq \frac{\partial \mathcal{A}_i}{\partial U_j}(\mathbf{U}). \tag{4.4}$$ {Eq:Jacobian}

By repeatedly solving Equation (4.3) one obtains a standard Newton scheme for Equation (4.2), yielding a sequence of approximate solutions $\mathbf{U}_0, \mathbf{U}_1, \mathbf{U}_2, \ldots$ obtained from an initial guess $\mathbf{U}_0$ through the iteration scheme $\mathbf{U}_{n+1} = \mathbf{U}_n + \mathbf{s}'_n$. In the classical undamped Newton method, the correction step $\mathbf{s}'_n$ is set to be the whole Newton-step, i.e $\mathbf{s}'_n = \mathbf{s}_n$ with

$$\mathbf{s}_n \coloneqq -\partial\mathcal{A}(\mathbf{U}_n)^{-1}\mathcal{A}(\mathbf{U}_n), \tag{4.5}$$ {eq:NewtonStep}

which is computed using a direct solver. Unfortunately, convergence of the Newton method for any starting value $\mathbf{U}_0$ is not guaranteed.

A big drawback of Newton method is the calculation of the Jacobian matrix, since its direct calculation using usual methods can be computationally expensive. The *BoSSS* framework provides an efficient algorithm for the evaluation of the Jacobian, and is presented in Section 4.2.4

4.2.2 Dogleg Method

sec:newton

In order to increase robustness when the distance between $\mathbf{U}_0$ and the exact solution $\mathbf{U}$ is large, we employ a globalization approach, presented by Pawlowski et al. Pawlowski et al. (2006) and Pawlowski et al. (2008), known as the Dogleg-method, or Newton-Dogleg method. Here, we intend to give only the central ideas of method and refer to the original works for further details. Obviously, the exact solution of Equation (4.2) is also a minimum of the functional

$$f(\mathbf{U}) \coloneqq \frac{1}{2} \|\mathcal{A}(\mathbf{U})\|_2^2 . \tag{4.6}$$

One observes that $\nabla f(\mathbf{U}) = \partial\mathcal{A}(\mathbf{U})^T \mathcal{A}(\mathbf{U})$. For $\mathbf{U}_n$, the approximate Cauchy point with respect to the 2-norm, is defined as the minimizer $\mathbf{g}_n$ of $\|\mathcal{A}(\mathbf{U}_n) + \partial\mathcal{A}(\mathbf{U}_n)\mathbf{g}_n\|_2$ in the direction of steepest decent, i.e. $\mathbf{g}_n = \lambda \nabla f(\mathbf{U}_n)$, $\lambda \in \mathbb{R}$. Substituting $\mathbf{w} \coloneqq -\partial\mathcal{A}(\mathbf{U}_n)\nabla f(\mathbf{U}_n)$, $\mathbf{g}_n$ is given by

$$\mathbf{g}_n = \frac{\mathcal{A}(\mathbf{U}_n) \cdot \mathbf{w}}{\mathbf{w} \cdot \mathbf{w}} \nabla f(\mathbf{U}_n). \tag{4.7}$$

{eq:CauchyPoint}

For the Newton-Dogleg method, the correction step $\mathbf{s}'_n$ is chosen along the so-called Dogleg curve, which is the piece-wise linear curve from the origin to $\mathbf{g}_n$ and further to $\mathbf{s}_n$. The selection of $\mathbf{s}'_n$ on this curve is determined by the trust-region diameter $\delta > 0$:

- If $\|\mathbf{s}_n\|_2 \leq \delta$, $\mathbf{s}'_n = \mathbf{s}_n$.
- If $\|\mathbf{g}_n\|_2 \leq \delta$ and $\|\mathbf{s}_n\|_2 > \delta$, $\mathbf{s}'_n$ is chosen on the linear interpolation from $\mathbf{g}_n$ to $\mathbf{s}_n$ so that $\|\mathbf{s}'_n\|_2 = \delta$: For the ansatz $\mathbf{s}'_n = \tau\mathbf{s}_n + (1-\tau)\mathbf{g}_n$, the interpolation factor τ is given as $\tau = (a^2 - c + \sqrt{(a^2+b^2-2c)\delta^2 - a^2b^2 + c^2})/(a^2+b^2-2c)$ with $a = \|\mathbf{g}_n\|_2$, $b = \|\mathbf{s}_n\|_2$ and $c = \mathbf{g}_n \cdot \mathbf{s}_n$.
- If $\|\mathbf{g}_n\|_2 > \delta$, $\mathbf{g}_n = (\delta/\|\mathbf{g}_n\|_2)\mathbf{g}_n$.

The choice and adaptation of the trust region diameter δ throughout the Newton-Dogleg procedure follows a sophisticated heuristic, mainly based on comparing the actual residual reduction $\text{ared}_n \coloneqq \|\mathcal{A}(\mathbf{U}_n)\|_2 - \|\mathcal{A}(\mathbf{U}_n + \mathbf{s}'_n)\|_2$ with the predicted residual reduction $\text{pred}_n \coloneqq \|\mathcal{A}(\mathbf{U}_n)\|_2 - \|\mathcal{A}(\mathbf{U}_n) + \partial\mathcal{A}(\mathbf{U}_n)\mathbf{s}'_n\|_2$; For the direct solver used in this work pred_n simplifies to $\text{pred}_n \coloneqq \|\mathcal{A}(\mathbf{U}_n)\|_2$. We replicate the algorithm here, for the sake of completeness:

(1) Set $n = 0$, $\delta_n = \min(10^{10}, \max(2 \cdot 10^{-6}, \|\mathbf{s}_0\|_2))$.

(2) Compute the Newton step $\mathbf{s}_n$ and the Cauchy point $\mathbf{g}_n$ and find $\mathbf{s}'_n$ on the Dogleg curve with respect to the recent δ_n.

(3) While $\text{ared}_n \leq \text{pred}_n$ do: Update trust region diameter $\delta_n \leftarrow 0.5\ \delta_n$ and re-compute $\mathbf{s}'_n$. If $\delta_n < 10^{-6}$ terminate abnormally and mark the computation as failed.

(4) If the convergence criterion (see below) is fulfilled, terminate and mark the computation as success.

(5) Perform a final update of the trust region: Set

$$\delta_{n+1} = \begin{cases} \max(10^{-6}, \|\mathbf{s}_n\|_2) & \text{if } \text{ared}_n/\text{pred}_n < 0.1 \text{ and } \|\mathbf{s}_n\|_2 \delta_n \\ \max(10^{-6}, 0.25 \cdot \delta_n) & \text{else, if } \text{ared}_n/\text{pred}_n < 0.1 \\ \min(10^{10}, 4 \cdot \delta_n) & \text{else, if } \text{ared}_n/\text{pred}_n > 0.75 \\ \delta_n & \text{otherwise} \end{cases}$$

Set $\mathbf{U}_{n+1} = \mathbf{U}_n + \mathbf{s}'_n$, update $n \leftarrow n+1$ and return to step (2).

All constants used in the algorithm above have been taken from the work of Pawlowski et al. Pawlowski et al. (2006) For a detailed description of the underlying ideas we also refer to these works, which in turn are based on algorithms from Dennis and Schnabel's textbook. Dennis and Schnabel (1996)

4.2.3 Linear solver

ssec:LinearSolver

The computation of the Newton step according to Equation (4.5) requires the inversion of the Jacobi matrix. This is done by means of the in *BoSSS* integrated orthonormalization multigrid algorithm Kummer et al. (2021), which at the lowest multigrid levels makes use of the sparse direct solver `PARDISO` (Parallel Sparse Direct and Multi-Recursive Iterative Linear Solvers), originally developed by Schenk et al. (Schenk et al., 2000; Schenk and Gärtner, 2002; Schenk and Gärtner, 2004), from the "Intel(R) Parallel Studio XE 2018 Update 3 Cluster Edition for Windows" library collection to solve the linear system.

For some of the testcases presented here, particularly the cases with combustion, the use of `PARDISO` for the solution of the linear problem resulted in memory problems. An active field of study in the BoSSS development group is that of iterative algorithms for solving linear systems. The BoSSS code features a multigrid orthonormalization method, using additive Schwarz schemes as smoothers for all multigrid levels, except the coarsest one, where `PARDISO` is used. This method of solution proved to be adequate to solve systems that are too big to be solved directly by `PARDISO`, and is adopted for all combustion calculations.

What do i exactly do with the linear system? Preconditioning?

4.2.4 Calculation of the Jacobian matrix

ssec:EvalJacobian

During the development process of the XNSEC solver different strategies for the calculation of the Jacobian matrix $\partial\mathcal{A}$ were tested and are shown here.

First, it is interesting to show the relationship existing between two well known methods for solving nonlinear systems: Picard iterations and Newton's method. First, note that the non-linear problems Equation (4.2) appearing in this work have the structure

$$\mathcal{A}(\mathbf{U}) \coloneqq A(\mathbf{U})\mathbf{U} - \mathbf{b}. \tag{4.8}$$

Thus, the Jacobian matrix Equation (4.4) can be written as

$$\partial\mathcal{A}_{ij}(\mathbf{U}) = A_{ij} + \sum_k \frac{\partial A_{ik}}{\partial U_j} U_K = A_{ij} + A'_{ij} U_K. \tag{4.9}$$

Inserting these , the linear system to be solved using Newton's method can be written as

$$\underbrace{A(\mathbf{U}_n)(\mathbf{U}_n + \mathbf{s}'_n) - \mathbf{b}}_{\text{Picard system}} + A'(\mathbf{U}_n)\mathbf{U}_n\mathbf{s}'_n = 0 \tag{4.10}$$

This makes the relationship between the two algorithms apparent. Unlike Picard's method of iterations, Newton's method requires additionally the evaluation of the Jacobi matrix, which must be approximated in some way. This section shows three strategies that were used throughout the development of this work for that purpose.

Ad-hoc linearization of the Jacobian matrix

For some problems, particularly saddle-point problems, the Jacobian A' can be approximated fairly well by simply evaluating the operator matrix. Thus, the system to be solved for Newton Iteration yields

$$A(\mathbf{U}_n)(\mathbf{U}_n + \mathbf{s}'_n) - \mathbf{b} + A(\mathbf{U}_n)\mathbf{U}_n\mathbf{s}'_n = 0 \tag{4.11}$$

This strategy offers a computationally cheap algorithm to obtain a solution of the nonlinear problem. However, the method offers limited robustness, and is known to be prone to fail for non-saddle-point problems (Kikker, 2020).

Approximation of the Jacobian matrix by finite differences

A straightforward way of calculating the Jacobian matrix is to use forward finite differences as an approximation.

$$A'(\mathbf{U})_j \coloneqq \frac{A(\mathbf{U} + \varepsilon \|\mathbf{U}\| \mathbf{e}_j) - A(\mathbf{U})}{\varepsilon \|\mathbf{U}\|} \tag{4.12}$$

where $\mathbf{e}_j$ is the unit vector with jth component equal to one, and zero in all other components. The value of ε should be chosen small, as usual when calculating finite differences, but also large enough not to disturb the calculations due to problems caused by floating-point rounding calculations. For all calculations in this work, the value $\varepsilon = \sqrt{\texttt{eps}}$ was adequate, where $\texttt{eps} = 2.22044604925031 \cdot 10^{-16}$ is the floating point accuracy for double precision.

The calculation of the forward finite difference approximation is a costly operation, especially for large systems, where it can be particularly prohibitive. However it offers a robust way to approximate the Jacobian. Strategies for improve the efficiency of this calculation exists, such as the use of analytic Jacobians, or use of the sparsity patterns of the Jacobian (Kelley, 1995). These are not treated in the present work.

Approximation of the Jacobian from differentiation of equation components

The finite difference Jacobi matrix calculation shown above is a fairly simple but computationally expensive calculation. The *BoSSS* code is capable of evaluating the Jacobian matrix automatically from the equation components given in Section 3.2.1. First, note that $\mathcal{A}(\mathbf{U})$ could be written as

$$[\mathcal{A}(\mathbf{U})]_i = \mathcal{N}(\mathbf{U}_h, \mathbf{\Phi}_i) = \int_{\Omega_h} N_1(\mathbf{x}, \mathbf{U}_h, \nabla\mathbf{U}_h)\cdot\mathbf{\Phi}_i + N_2(\mathbf{x}, \mathbf{U}_h, \nabla\mathbf{U}_h)\cdot\nabla\mathbf{\Phi}_i \mathrm{dV} + \oint_\Gamma \ldots \mathrm{dS}. \tag{4.13}$$

{eq:NonlinearGestalt}

The edge integral, which is left out in Equation (4.13), can be expressed analogously to the volume integral, i.e. as a sum over four nonlinear functions, multiplied by $\mathbf{\Phi}_i^+$, $\mathbf{\Phi}_i^-$, $\nabla\mathbf{\Phi}_i^+$ and $\nabla\mathbf{\Phi}_i^-$, respectively. These functions themselves may include the dependence on $\mathbf{x}$, $\mathbf{U}_h^+$, $\mathbf{U}_h^-$, $\nabla\mathbf{U}_h^+$ and $\nabla\mathbf{U}_h^-$. The is however omitted here for the sake of compactness, but the treatment is analogue. Realizing that

$$\frac{\partial \mathbf{U}_h}{\partial \mathbf{U}_j} = \mathbf{\Phi}_j \tag{4.14}$$

and by application of the chain rule, it is possible to derive an expression for the calculation of the entries of the Jacobian matrix from the equation components as

$$\partial\mathcal{A}_{ij}(\mathbf{U}) = \int_{\Omega_h} (\partial_{\mathbf{U}_h} N_1(\mathbf{x}, \mathbf{U}_h, \nabla\mathbf{U}_h)\mathbf{\Phi}_j + \partial_{\nabla\mathbf{U}_h} N_1(\mathbf{x}, \mathbf{U}_h, \nabla\mathbf{U}_h)\nabla\mathbf{\Phi}_j) \cdot \mathbf{\Phi}_i + \ldots \,\mathrm{dV} + \oint_{\Gamma} \ldots \,\mathrm{dS}. \tag{4.15}$$

{eq:JacobiGestalt}

All omitted terms in Equation (4.15) can be approximated analogously to the contributions for N_1. In the *BoSSS* code, derivatives $\partial_{\mathbf{U}_h} N_1(\ldots)$ and $\partial_{\nabla\mathbf{U}_h} N_1(\ldots)$ are approximated by a finite difference, using a perturbation by $\sqrt{\varepsilon}$ in the respective argument.

This approach has the significant improvement that it offers an efficient and accurate way to obtain the Jacobian matrix, unlike the two approaches mentioned above. For this reason, this option is the one chosen for the resolution of all the testcases shown in this work.

4.2.5 Termination criterion

ssec:TerminationCriterion

A simple approach to determine that a Newton-Dogleg loop can be terminated is to check whether the residual norm has fallen below a certain threshold, i.e. $\|\mathcal{A}(U_n)\| \leq$ tol. A universal choice for the tolerance is indeed difficult, especially for investigations of convergence properties (cf. Section 5.2.3 and Section 5.3.3). If it is chosen too low, the algorithm may never terminate, because of dominating numerical round-off errors. On the other hand, if it is chosen too high, the error of the premature termination may dominate the error of the spatial discretization and one cannot take the full advantage of the high-order method. Therefore the goal is to continue the Newton-Dogleg method until the lowest possible limit dictated by floating point accuracy is reached. To identify the limit in a robust way, we first define the residual-norm skyline as

$$\mathrm{sr}_n := \min_{j \leq n} \|\mathcal{A}(\mathbf{U}_j)\| \tag{4.16}$$

and, for $n \geq 2$, the averaged reduction factor

$$\mathrm{arf}_n := \frac{1}{2}\left(\frac{\mathrm{sr}_{n-2}}{\max\{\mathrm{sr}_{n-1}, 10^{-100}\}} + \frac{\mathrm{sr}_{n-1}}{\max\{\mathrm{sr}_n, 10^{-100}\}}\right). \tag{4.17}$$

The Newton-Dogleg method is terminated if

$$n \geq 2 \text{ and } \mathrm{sr}_n \leq 10^{-5} + 10^{-5}\|\mathbf{U}_n\|_2 \text{ and } \mathrm{arf}_n < 1.5. \tag{4.18}$$

For the computations in this work, this choice guarantees that the nonlinear system is solved as accurately as possible. It secures that the numerical error is dominated by the error of the spatial or temporal discretization and not by the termination criterion of the Newton-Dogleg method. The skyline approach ensures robustness against oscillations close to the lower limit.

4.3 Homotopy method

sec:HomotopyMethod

Although the Newton-Dogleg method works well for a variety of cases, we experienced convergence problems for some of the test cases presented in next section. In particular, for the differentially heated cavity test case, the method was not successful on finding a convergent solution for a Rayleigh number $Ra \geq 10^5$ within 60 Newton iterations. In such cases we used a homotopy strategy, which is loosely based on ideas from the textbook of Deuflhard (2011).

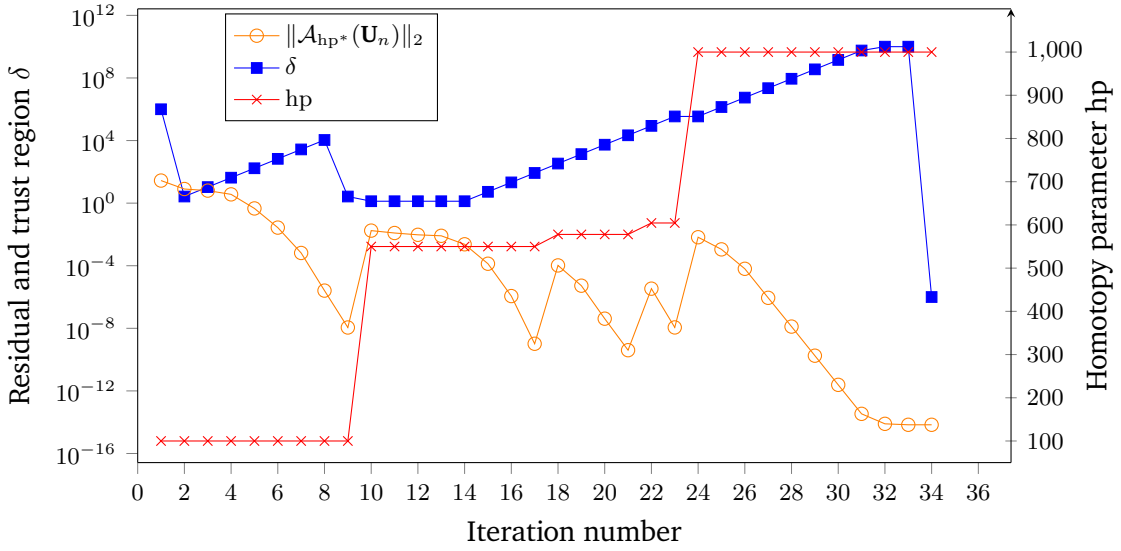


Figure 4.1: behavior of the homotopy method for the differentially heated cavity test case. The homotopy parameter hp in this case is the Reynolds number. fig:Homotopyevolution

We start by identifying a parameter that makes the solution of the nonlinear problem difficult to solve. In the following we will refer to this variable as the homotopy parameter. The main idea of the homotopy strategy consists of solving a series of simpler problems, starting with a parameter where the problem is easy to solve, and carefully increasing it until the desired value is reached. Let Hp denote the value of the homotopy parameter for which a solution is being sought. Let

$$\mathcal{A}_{\mathrm{hp}^*}(\mathbf{U}) = 0 \tag{4.19}$$

{eq:NonlinearAt-hp}

be the discretized system for a certain intermediate homotopy-parameter hp^*, between 0 and the 'target' homotopy-parameter Hp, i.e. $0 \leq \mathrm{hp}^* \leq \mathrm{Hp}$. Furthermore, let $\mathbf{U}_{\mathrm{hp},\epsilon}$ be an approximate solution to the problem (4.19) with $\mathrm{hp}^* = \mathrm{hp}$, up to a tolerance ϵ, i.e.

$$\|\mathcal{A}_{\mathrm{hp}}(\mathbf{U}_{\mathrm{hp},\epsilon})\|_2 \leq \epsilon. \tag{4.20}$$

For the sake of clarity when discussing the algorithm which follows below, we distinguish between the intermediate homotopy-parameter hp for which we assume to already have found an acceptable solution and the next homotopy-parameter hp^* that we are currently trying to find a solution for. For any $\mathrm{hp}^* > \mathrm{hp}$ we set $\epsilon = 10^{-5}\,\|\mathcal{A}_{\mathrm{hp}*}(\mathbf{U}_{\mathrm{hp},\epsilon})\|_2$, i.e. we aim for a residual norm reduction of at least five orders of magnitude with respect to the initial residual norm. If $\mathrm{hp}^* = \mathrm{Hp}$, the termination criterion presented in section 4.2.5 is applied. An approximate solution for the target homotopy-parameter is found by the following recipe:

(1) Set $\mathrm{hp} = 0$, i.e. start by obtaining an (approximate) solution $\mathbf{U}_{0,\epsilon}$.

(2) Search for a an increased homotopy-parameter hp^*: Find the minimal $i \geq 0$ so that for $\mathrm{hp}^* = \frac{1}{2^i}(\mathrm{Hp} - \mathrm{hp}) + \mathrm{hp}$ one has $\left\|\mathcal{A}_{\mathrm{hp}^*}(\mathbf{U}_{\mathrm{hp},\epsilon})\right\|_2 \leq \delta_{\max}\,\|\mathcal{A}_{\mathrm{hp}}(\mathbf{U}_{\mathrm{hp},\epsilon})\|_2$ Here, $\delta_{\max}$ is the maximal allowed increase of the residual for an increased homotopy-parameter hp^*; $\delta_{\max}$ is adapted in the following steps, as an initial guess we use $\delta_{\max} = 10^6$.

(3) Use the Newton-Dogleg method to compute an approximate solution to the problem (4.19), for the homotopy-parameter hp^*, using the solution $\mathbf{U}_{\mathrm{hp},\epsilon}$ as an initial guess.

- If the Newton-Dogleg method did not converge successfully within ten steps, the homotopy-parameter increase from hp to hp^* was probably too large. Set $\delta_{\max} \leftarrow 0.2\delta_{\max}$ and go to step (2).
- If the Newton-Dogleg method reached its convergence criterion and if the target homotopy-parameter is reached, i.e. $\mathrm{hp}^* = \mathrm{Hp}$, the algorithm has successfully found an approximate solution for $\mathcal{A}_{\mathrm{Hp}}(\mathbf{U}) = 0$ and can terminate.
- Otherwise, if the Newton-Dogleg method converged successfully, but is below the target homotopy-parameter: Accept the solution and set $\mathrm{hp} \leftarrow \mathrm{hp}^*$. If the Newton-Dogleg method took less than three iterations to reach the convergence criterion, set $\delta_{\max} \leftarrow 8\delta_{\max}$. Return to step (2).

An exemplary run of the method is shown in Figure 4.1. The homotopy parameter hp in this particular case is the Reynolds number. The homotopy-parameter hp was increased for iterations 10, 18, 22 and 24, causing an increase of the residuals $\|\mathcal{A}_{\mathrm{hp}^*}(\mathbf{U}_n)\|_2$, leading to a convergent solution after 34 Newton iterations. The presented algorithm offers a robust method for finding steady-state solutions of highly nonlinear systems.

4.4 Initialization of combustion applications

ssec:MethodCombustion

The proposed algorithm for obtaining steady state solutions of finite reaction rate combustion problems involves first solving the problem assuming an infinite reaction rate (the flame sheet problem). This requires first solving the system presented in Section 2.2, where Equations (3.38a) to (3.38c) need to be solved in a coupled manner together with the expressions that link the temperature and mass fractions to the mixture fraction in order to be able to evaluate the density and . This idea has been already employed in various works (Smooke et al., 1986; Smooke and Giovangigli, 1992). The reason for the use of this pre-step is twofold:

- The system of Equations (2.22a) to (2.22d) presents multiple solutions. One is the pure mixing (frozen) solution, where no chemical reaction has taken place, and the other one is the ignited solution, where the flame is present. Using the flame sheet solution as initial estimate ensures that the path taken by Newton's algorithm will tend towards the ignited solution.
- Solving Equations (2.22a) to (2.22d) using a Newton-type method requires adequate starting estimates in order to converge. Using the flame sheet solution as initial estimate improves the convergence properties of the method.

It should be noted regarding the solution of the flame sheet problem (cf. Section 2.2) that the sharp change in the primitive variables around $z = z_{st}$ is problematic in certain scenarios. In particular, the non-smoothness of the derived variables could lead to Gibbs phenomenon-type problems. This inconvenient can be remedied by using a regularized form of the equations. The smoothing function $\mathcal{H}$ is defined as

$$\mathcal{H}(z) \approx \frac{1}{2}(1 + \tanh(\sigma(z - z_{st}))), \tag{4.21}$$

{eq:regularization_MF}

This function is useful for creating a smooth transition between two functions, since it returns values close to 0 for $z \ll z_{st}$ and values close to 1 for $z \gg z_{st}$. The sharpness of the transition at the point $z = z_{st}$ is dictated by the parameter σ. In Figure 4.2 the smoothing function H using different smoothing parameters σ is shown. Clearly, increasing the value of σ increases the sharpness of the transition at the point z_{st}. For a very big σ value the function H resembles the Heaviside step function Using Equation (4.21) the temperature and mass fraction fields

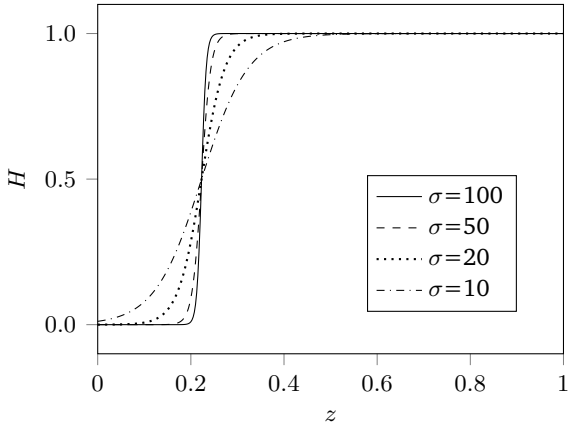


Figure 4.2: Smoothing function at $z_{\text{st}} = 0.22$ for different smoothing parameters σ. {fig:SmoothingFunc}

can be written as

$$T(z) = zT_F^0 + (1-z)T_O^0 + \frac{QY_F^0}{c_p} z_{st} \frac{1-z}{1-z_{st}} \mathcal{H}(z) + \frac{QY_F^0}{c_p} z\left(1 - \mathcal{H}(z)\right), \quad \text{(4.22a)} \{eq:BS-TR\}$$

$$Y_F(z) = Y_F^0 \frac{z - z_{st}}{1 - z_{st}} \mathcal{H}(z), \quad \text{(4.22b)} \{eq:BS-YFR\}$$

$$Y_O(z) = Y_O^0 \frac{z_{st} - z}{z_{st}} (1 - \mathcal{H}(z)), \quad \text{(4.22c)} \{eq:BS-YOR\}$$

$$Y_P(z) = Y_O^0 \frac{M_P \nu_P}{M_O \nu_O} (1-z)\mathcal{H}(z) + Y_F^0 \frac{M_P \nu_P}{M_F \nu_F} z(1 - \mathcal{H}(z)), \quad \text{(4.22d)} \{eq:BS-YPR\}$$

$$Y_N(z) = (1 - Y_F^0)z + (1 - Y_O^0)(1 - z). \quad \text{(4.22e)} \{eq:BS-YNR\}$$

The use of this regularized form of the equations results in practice on a spreading of the flame front, which eases the numerical calculation (Braack et al., 1997). In Figure 4.3 the effect of the smoothing factor σ on calculations of a flame in a counter-flow configuration are shown. It can be clearly observed how for decreasing σ the solution becomes smoother.

One question one could certainly ask is under what flame conditions the infinite reaction rate solution (also called flame sheet solution in the following) effectively is a good initial estimate for Newton's algorithm. Obviously for systems that respect the assumptions done for the flame sheet the obtained solution will be very close to the finite-rate solution (see Figure 2.2). The assumption of an infinitely fast chemical reaction implies that the time scales associated with the chemical reaction are infinitely smaller than the flow scales, or in other words, Da $\to \infty$. For this reason, the flame sheet solution is expected to give a similar solution for cases close to equilibrium (where the Damköhler number is large). On the other hand, in

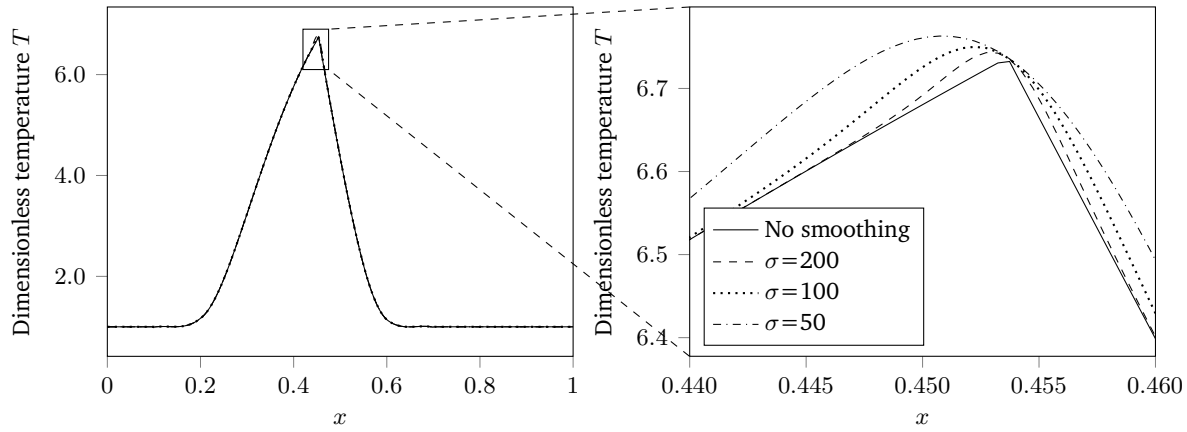


Figure 4.3: Temperature profile calculated in the center-line of a counter-flow flame configuration for different smoothing parameters σ.

fig:smoothings

cases that are far from equilibrium, as, for example, in the case of a flame in conditions close to extinction, it is expected that the flame sheet solution will depart considerably from the solution with a finite reaction rate.

It should be noted that within the derivation of the equations for the flame sheet it is only assumed that the heat capacity is the same for all components ($c_{kp} = c_p$), but it is still possible to consider a dependence on temperature. However, this introduces a difficulty, since the evaluation of the temperature with Equation (2.34) requires c_p, which according to Equation (2.28), depends in turn on the temperature. Solving the system of equations required to obtain c_p and T is very expensive, since it would require solving it every time the temperature must be evaluated -in particular for the evaluation of the density ρ and transport parameters μ and ρD-. This problem can be solved by simply assuming a constant representative value of c_p.

The problem that now arises is the selection of a suitable c_p. In the work by Xu and Smooke (1993) it is suggested to estimate it simply on the basis of experimental measurements, or also by selecting some representative value, such as c_p evaluated at the adiabatic temperature and stoichiometric conditions. In particular, in this work the value $\hat{c}_p = 1.3\,\mathrm{kJ\,kg^{-1}\,K^{-1}}$ was adequate for all calculations. This constant value of the heat capacity proved to yield a flame sheet solution which is an adequate estimate for finite-rate simulations, even for cases with a nonconstant heat capacity.

In a similar fashion, the assumption of unity Lewis number in the flame sheet system delivers a solution that slightly deviates from the solution of the finite chemistry rate problem with nonunity Lewis numbers. Nevertheless, this small deviation does not preclude the use of the flame sheet solution as an adequate initial estimate for Newton's method.

Adaptive Mesh Refinement

ssec:MeshRefinement

The area where the chemical reaction takes place is usually a thin region whose thickness is defined by the availability of reactants.

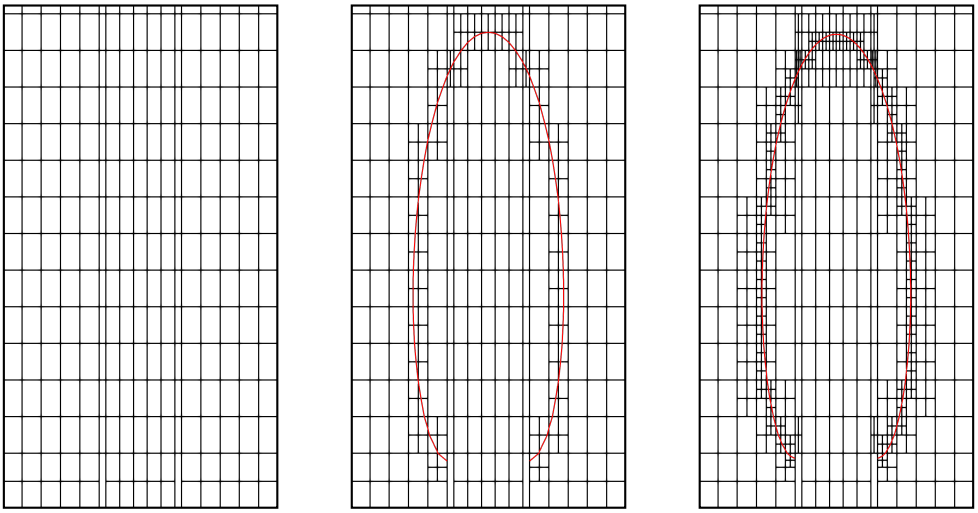


Figure 4.4: Adaptive mesh refinement around the stoichiometric surface in a coflow flame configuration.

4.4.1 Solver safeguard

Another strategy that proved to be useful in improving the convergence properties of the iterative scheme is the use of a solver safeguard, which is used to avoid unphysical solutions during the solution procedure, such as negative temperatures or temperatures higher than the adiabatic temperature. In particular, the idea is to clip values from the solution fields delivered by Newton's algorithm which are known to be unphysical, and limit the solution fields by user defined values. This clipping emulates in a sense the effect of schemes such as Total Variation diminishing Method (TVM) or Essentially Non-Oscillatory methods (ENO) (Nicoud, 2000).

Particularly, for an arbitrary scalar ξ the values are bounded in the range $[\xi_{\text{min}} - \epsilon_{\text{safe}}, \xi_{\text{max}} + \epsilon_{\text{safe}}]$, where ξ_{min} and ξ_{max} are user defined bounds, and $\epsilon_{\text{safe}} = 10^{-4}$. For example, the mass fractions by definition should have a value between zero and one, thus $Y_{k,\text{min}} = 0$ and $Y_{k,\text{max}} = 1$. For certain problems, particularly problems involving combustion, it could be also useful to limit the value of the temperature, which can be bounded using the inlet conditions as the minimum value, and the adiabatic temperature as the maximum value.

The occurrence of these non-physical values is not always problematic, and in theory the Newton algorithm above should be able to correctly handle them and finally find the solution of the nonlinear system. However in certain cases this can lead to problems. Just to mention one example, a negative temperature would result in a imaginary value of the viscosity if it is calculated according to Equation (2.29). Particularly for problems with sharp gradients this could be problematic due to dispersion phenomena.

5 Numerical results

ch:results

The following sections show a comprehensive solver validation using various test cases presented in increasing levels of complexity. The tests also allow to highlight some of the benefits of the DG-method and the algorithms introduced before in this work, namely the globalized Newton method, homotopy strategy and the initialization of combustion applications using flame sheet estimates.

First, in Section 5.1 the applicability of the solver is analyzed for isothermal single-component systems. Later in Section 5.2 several single-component non-isothermal configurations are studied. Finally, in Section 5.3 test cases for multicomponent non-isothermal systems are presented, with a particular emphasis on systems where combustion is present.

All calculations shown here were performed on a cluster with the following speficiations.

- **CPU** 4x8 cores (Intel(R) Xeon(R) CPU E5-4627 v2)
- **CPU vector extension** Intel® AVX
- **CPU speed** basis 3.30 GHz, turbo 3.6 GHz
- **Memory** 512 GByte (DDR3-1600)

Unless otherwise stated, all calculations use the termination criteria presented in Section 4.2.5. Some of the results presented in this section have been published in (Gutiérrez-Jorquera and Kummer, 2022).

5.1 Single-component isothermal cases

sec:SingleCompIsotCase

The XNSEC solver presented in the previous chapter is validated first for single-component isothermal cases. In these cases, only the continuity and momentum equations are solved. The energy equation and the species concentration equations are replaced by the conditions $T = 1.0$ and $Y_0 = 1.0$ in the entire domain. This means that the physical properties of the flow (density and viscosity) are constant. Therefore, the flow is totally incompressible, since the density shows no thermodynamic or hydrodynamic dependence.

5.1.1 Lid-driven cavity flow

The lid-driven cavity flow is a classic test problem used for the validation of Navier-Stokes solvers. The system configuration is shown in Figure 5.1. It consists simply of a two-dimensional square cavity enclosed by walls whose upper boundary moves at constant velocity, causing the fluid to move. Benchmark results can be found widely in the literature for different Reynolds numbers. In this section, the results obtained with the XNSEC solver are compared with those published by Botella and Peyret (1998).

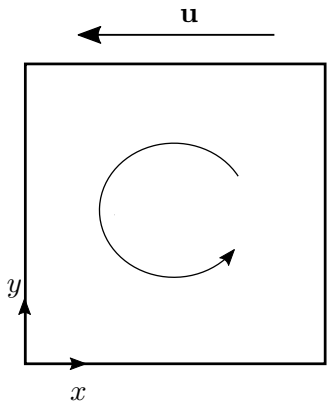


Figure 5.1: Schematic representation of the Lid-Driven cavity flow. fig:LidDrivenCavity

The problem is defined in the domain $\Omega = [0, 1] \times [0, 1]$. The system is solved for the velocity vector $\mathbf{u} = (u, v)$ and the pressure p. All boundary conditions are Dirichlet-type, particularly with $\mathbf{u} = (-1, 0)$ for the boundary at $y = 1$ and $\mathbf{u} = (0, 0)$ for all other sides. The gravity vector is set to $\mathbf{g} = (0, 0)$. A Cartesian mesh with extra refinement at both upper corners is used, and is shown in Figure 5.2. The refinement was done to better represent the complex effects that take place in the corners. The streamline plot presented in Figure 5.2 shows the different vortex structures typical of this kind of system, where in addition to the main vortex of the cavity, smaller structures appear in the corners.

The lid-driven cavity was calculated for a Reynolds number $\mathrm{Re} = 1000$. For the calculations presented here, the polynomial degree is set to four for both velocity components and three for the pressure. A regular Cartesian mesh with 16×16 elements is used with extra refinement in the corners. In Figure 5.3 a comparison of the calculated velocity with the DG-Solver and the velocities provided by the benchmark is shown. Clearly, very good agreement is obtained, even by using a relatively coarse mesh (the benchmark result uses a grid with 160×160 elements).

A more rigorous comparison of results is presented in Table 5.1, where the extreme values of the velocity components calculated through the centerline of the cavity are compared with

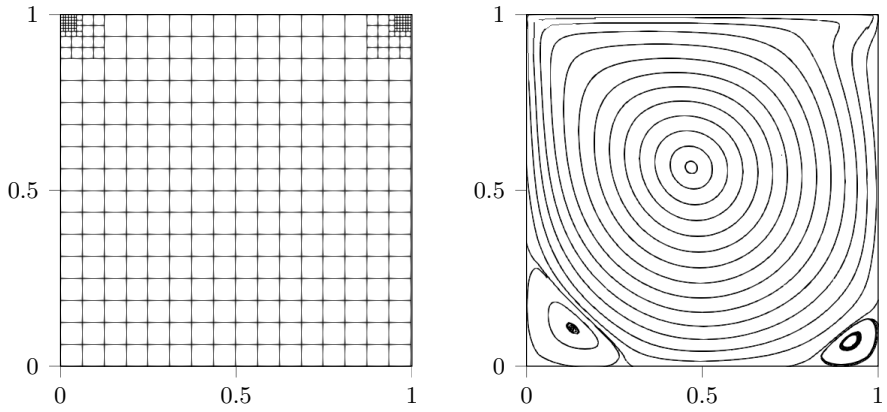


Figure 5.2: Mesh and streamlines of the lid-driven cavity flow with $\mathrm{Re} = 1000$. fig:LiddrivenMesh

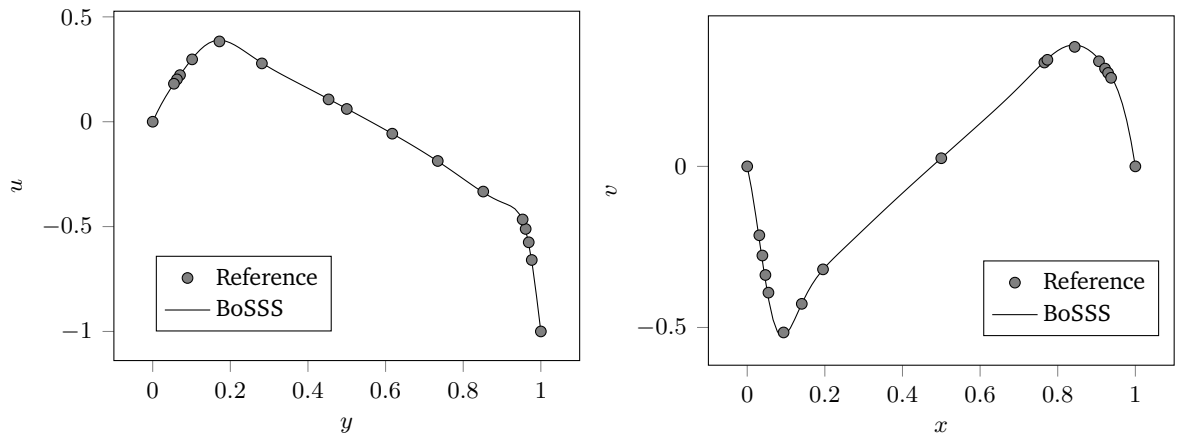


Figure 5.3: Calculated velocities along the centerlines of the cavity and reference values. Left plot shows the x-velocity for $x = 0.5$. Right plot shows the y-velocity for $y = 0.5$ fig:LidVelocities

Mesh	$u_{\max}$	$y_{\max}$	$v_{\max}$	$x_{\max}$	$v_{\min}$	$x_{\min}$
16×16	0.3852327	0.1820	0.3737295	0.8221	-0.5056627	0.0941
32×32	0.3872588	0.1821	0.3760675	0.8227	-0.5080496	0.0943
64×64	0.3897104	0.1748	0.3774796	0.8408	-0.5248360	0.0937
128×128	0.3886452	0.1720	0.3770127	0.8422	-0.5271487	0.0907
256×256	0.3885661	0.1717	0.3769403	0.8422	-0.5270653	0.0907
Reference	0.3885698	0.1717	0.3769447	0.8422	-0.5270771	0.0908

Table 5.1: Extrema of velocity components through the centerlines of the lid-driven cavity for $\mathrm{Re} = 1000$. Reference values obtained from Botella and Peyret (1998) tab:LidCavityExtrema

the results presented by Botella and Peyret (1998). Different mesh resolutions were used for this comparison, particularly meshes with 16×16, 32×32, 64×64, 128×128 and 256×256 elements, each with extra refinement at the corners. It can be clearly seen how for the finest mesh the results obtained with the DG-solver are extremely close to the reference. A difference is only appreciated at the fifth digit after the decimal point for the velocity components. In case of the position of the extremal values no difference is observed. It can also be seen that the results obtained with the coarser meshes are still very close to those of the reference. It is worth mentioning that this comparison only considering number of elements could be considered unfair, since the reference uses another method for solving the governing equations. One of the advantages of the DG method is that choosing higher-order polynomials allows more information to be packed into each cell.

5.1.2 Backward-facing step

ssec:BackwardFacingStep

The backward-facing step problem is another classical configuration widely used for validation of incompressible CFD codes. It has been widely studied theoretically, experimentally, and numerically by many authors in the last decades (see, for example, Armaly et al. (1983), Barkley et al. (2000), and Biswas et al. (2004)). In Figure 5.4 a schematic representation

of the problem is shown. It consists of a channel flow (usually considered fully developed) that is subjected to a sudden change in geometry that causes separation and reattachment phenomena. For these reasons, this case can be considered more challenging than the one presented in the previous section, since special care of the mesh used has to be taken in order to capture accurately all complex phenomena taking place.

Although the backward-facing step problem is known to be inherently three-dimensional, it has been shown that it can be studied as a two-dimensional configuration along the symmetry plane for moderate Reynolds numbers. For the range of Reynolds numbers used in the calculations presented here, the two-dimensional assumption is justified (Barkley et al., 2000; Biswas et al., 2004). The origin of the coordinate system is set in the bottom part of the step. The step height S and channel height h characterize the system. Results in the literature are often reported as a function of the expansion ratio, defined as ER $= (h + S)/h$.

A series of simulations were performed with the objective of reproducing the results reported by Biswas et al. (2004), where the backward-facing step was calculated for Reynolds numbers up to 400 and for an expansion ratio of ER $= 1.9423$. In particular, the reported lengths of detachment and reattachment are used as a means of comparison with the results from the XNSEC solver.

The Reynolds number for the backward-facing step configuration is defined in the literature in many forms. Here, the definition based on the step height $\hat{S}$ and the mean inlet velocity $\hat{U}_{\text{mean}}$ is adopted as the reference length and velocity, resulting in

$$\text{Re} = \frac{\hat{S}\hat{U}_{\text{mean}}}{\hat{\nu}}. \tag{5.1}$$

The boundary at $x = -L_0$ is an inlet boundary condition, where a parabolic profile is defined with

$$u(y) = -6\frac{(y - S)(y - (h + S))}{h^2} \tag{5.2}$$

The system is isothermal, and the fluid is assumed to be air. The step length is set $S = 1$ and $h = 1.061$. To minimize the effects of the outlet boundary condition on the part of interest in the system, the length L of the domain is set to $L = 70S$. All other boundaries are fixed walls. From prior calculations, the effect of the domain length before the step was found to have almost no impact on the results and is set to $L_0 = S$. Preliminary studies showed that the calculated reattachment and detachment lengths are highly sensitive to the mesh resolution. For all simulations in this section, a structured grid with 88,400 elements is used. To better resolve the complex structures that occur in this configuration, smaller elements are used in the vicinity of the step, as seen in Figure 5.5. A polynomial degree of three was chosen for both velocity components and two for pressure.

The backward-facing step configuration exhibits varying behavior as the number of Reynolds changes. For small Reynolds numbers, a single vortex, usually called the primary vortex, appears in the vicinity of the step. Furthermore, as the Reynolds number increases, a second vortex eventually appears on the top wall, as shown schematically in Figure 5.4. The detachment and reattachment lengths of the vortices are values that are usually reported in the literature. It is possible to determine the detachment position by finding the point along the wall where the velocity gradient normal to the wall acquires a value equal to zero.

Figure 5.7 shows the detachment and reattachment lengths of the primary and secondary vortices obtained with the XNSEC solver for different Reynolds numbers, which are also

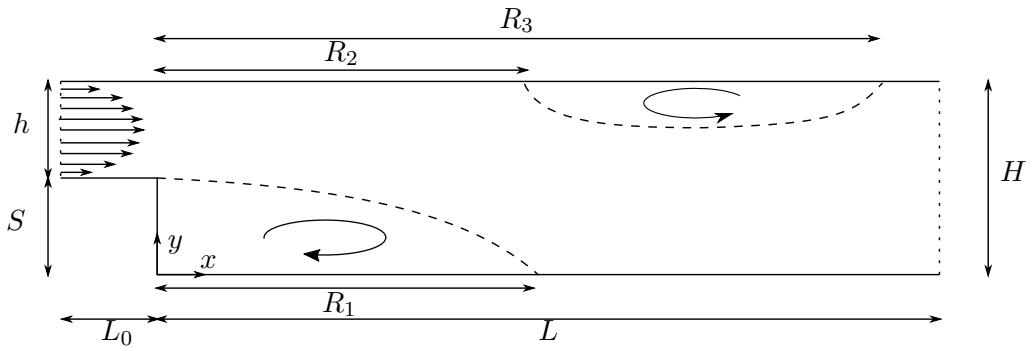


Figure 5.4: Schematic representation (not to scale) of the backward-facing step. Both primary and secondary vortices are shown. BFSsketch

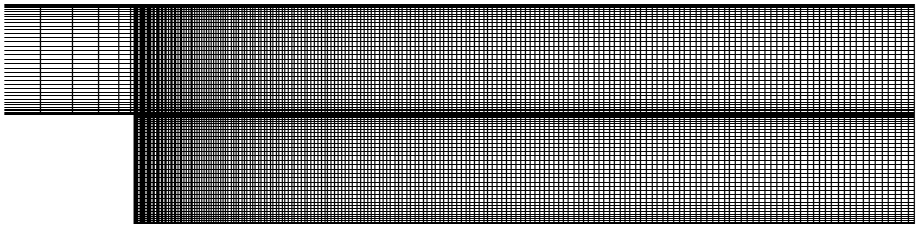


Figure 5.5: Mesh used for the backward-facing step configuration. bfsmesh

compared with the results presented in the reference paper from Biswas et al. (2004). Cubic splines have been used to accurately locate this point. It can be seen that the results for the detachment lengths of the primary vortex R_1 are in very good agreement with those of the reference. In the case of the secondary vortex, it is possible to see a very minimal deviation for the lengths of the reattachment R_3, hinting at a possible spatial underresolution far away from the step. It is interesting to note that, despite the fact that the reference does not report the existence of a secondary vortex for $\mathrm{Re} = 200$, it was possible to observe it with the XNSEC solver. The results allow us to conclude that it is possible to study flows with complex behavior for low- to moderate Reynolds numbers, at least in the isothermal case. In the next section, a non-isothermal case of this configuration will be studied.

It is worth mentioning that the evaluation of the global order of accuracy of the solver using the two incompressible test cases presented in this section is problematic due to the presence of singularities. Specifically the points at the corners at the coordinates $\mathbf{x} = (0, 1)$ and $\mathbf{x} = (1, 1)$ of the Lid-driven cavity (where the pressure is not finite according to Botella and Peyret (1998)), and at the corner of the step $\mathbf{x} = (0, S)$ of the backward-facing step are problematic. The accuracy of the solver will be assessed later in Section 5.2.2 making use of a analytical solution and in Section 5.2.3 using a solution obtained with a high spatial resolution.

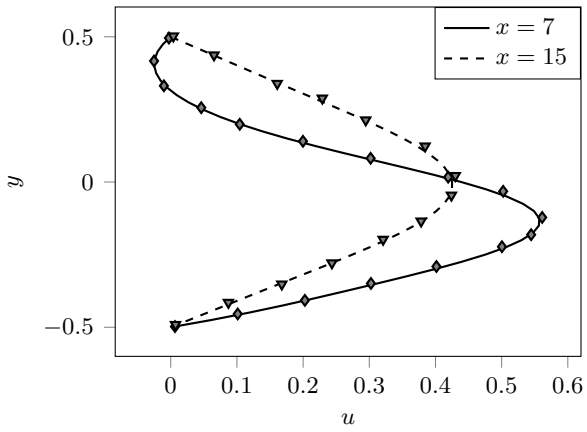


Figure 5.6: Distribution of x-component of velocity in the backward-facing step configuration for a Reynolds number of 400. Solid lines correspond to results obtained with the XNSEC solver.
fig:uvelBFS

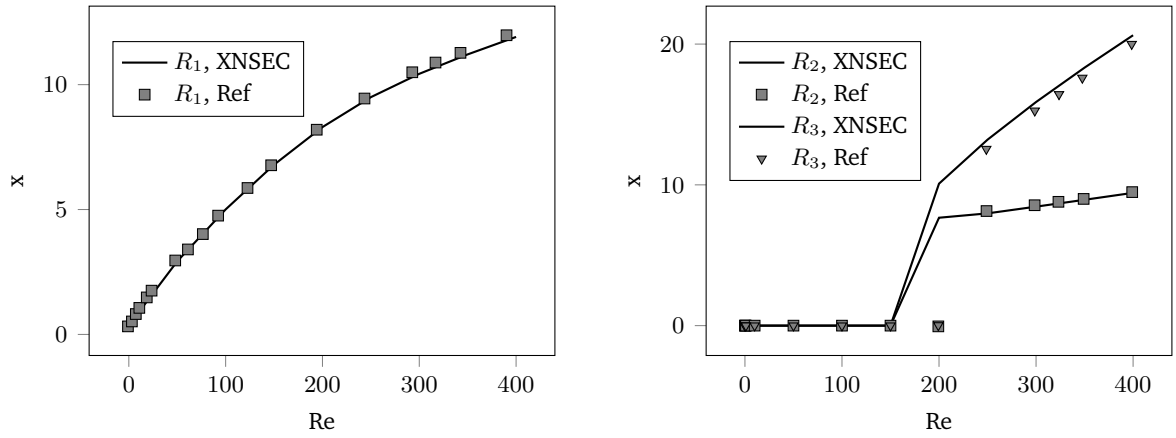


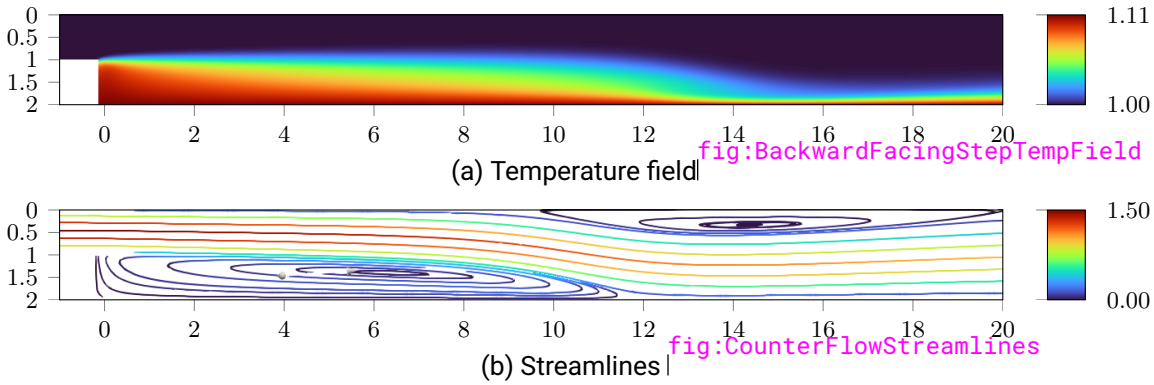
Figure 5.7: Detachment and reattachment lengths of the primary (left figure) and secondary (right figure) recirculation zones after the backward-facing step compared to the reference solution (Biswas et al., 2004).
fig:Re_De_Attachmentlengths

5.2 Single-component non-isothermal cases

sec:SinCompNonIsothermCase

For the test cases presented in this section, the equations for continuity, momentum and energy are solved. All systems are assumed to be single-component, thus $N = 1$ and $Y_0 = 1.0$. The tests shown in this section shall serve as a throughout validation of the XNSEC solver spatial and temporal discretization for low-Mach variable density flows.

First in Section 5.2.1 an extension of the backward-facing step configuration for a non-isothermal system is shown, and benchmark values are compared to references. Later in Section 5.2.2 a Couette flow configuration that presents a temperature gradient in the vertical direction is studied and compared to an analytical solution. Additionally the numerical accuracy of the solver is shown and compared to calculations using a SIMPLE-type algorithm. Later in Section 5.2.3 a heated square cavity configuration is studied to assess the solver's ability to simulate steady state flow configurations of variable density in closed systems. In Section 5.2.4



(a) Temperature field fig:BackwardFacingStepTempField

(b) Streamlines fig:CounterFlowStreamlines

Figure 5.8: Temperature profile and streamlines corresponding to the backward-facing Step configuration for Re $= 400$ and an expansion ratio of two. fig:BFS_Temperature_Streamlines

the flow over a heated cylinder is studied, which serves as a test for unsteady calculations of non-isothermal systems. Finally in Section 5.2.5 a classical Rayleigh-Bénard convection system is treated, showing the behavior of the solver for unstable systems.

5.2.1 Heated backward-facing step

ssec:HeatedBackwardFacingStep

As an extension to the previous casem the backward-facing step configuration in a non-isothermal configuration is studied, where the bottom wall is heated to a constant temperature.

In this section the configuration for a heated backward-facing step proposed in the work of Xie and Xi (2016) is solved. The fluid entering the system has a temperature equal to $\hat{T}_0 = 283\,\mathrm{K}$ and the bottom wall is set to a constant temperature of $\hat{T}_1 = 313\,\mathrm{K}$. The inlet temperature is used as the reference temperature, obtaining $T_0 = 1.0$ and $T_1 = 1.106$. In the work of Xie and Xi (2016) results are reported for the local Nusselt numbers and the local friction coefficients f_d along the bottom wall ($y = 0$) for different expansion ratios and Reynolds numbers. By combining the definition of the Nusselt number ($\mathrm{Nu} = \hat{h}\hat{L}/\hat{\lambda}$), Newton's law of cooling ($\hat{\mathbf{q}} = \hat{h}(\hat{T}_0 - \hat{T}_1)$), and Fourier's law of heat conduction ($\hat{\mathbf{q}} = \hat{\lambda}\hat{\nabla}\hat{T}$) a expression for the local Nusselt number is obtained.

$$\mathrm{Nu}_{\mathrm{loc}} = \frac{\hat{L}}{\hat{T}_0 - \hat{T}_1}\hat{\nabla}\hat{T}\cdot\hat{\mathbf{n}} \tag{5.3}$$

where $\hat{L}$ is the reference length. $\hat{L} = \hat{S}$ is chosen to be consistent with the definition of the Reynolds number of the reference. Furthermore, the local friction factor can be written as

$$f_d = \frac{8\hat{\nu}}{(\hat{U}_{\mathrm{mean}})^2}\hat{\nabla}\hat{u}\cdot\hat{\mathbf{n}} \tag{5.4}$$

Simulations were conducted for different Reynolds numbers and expansion ratios. In Figure 5.8 the temperature field and the streamlines corresponding to a calculation with Re $= 700$ are shown. Here, the apparition of the secondary vortex is seen in the top wall. Note that only a small part of the computational domain is shown. Far away from the step, a lightly skewed parabolic velocity profile is obtained, which is influenced by the density variations on the vertical direction.

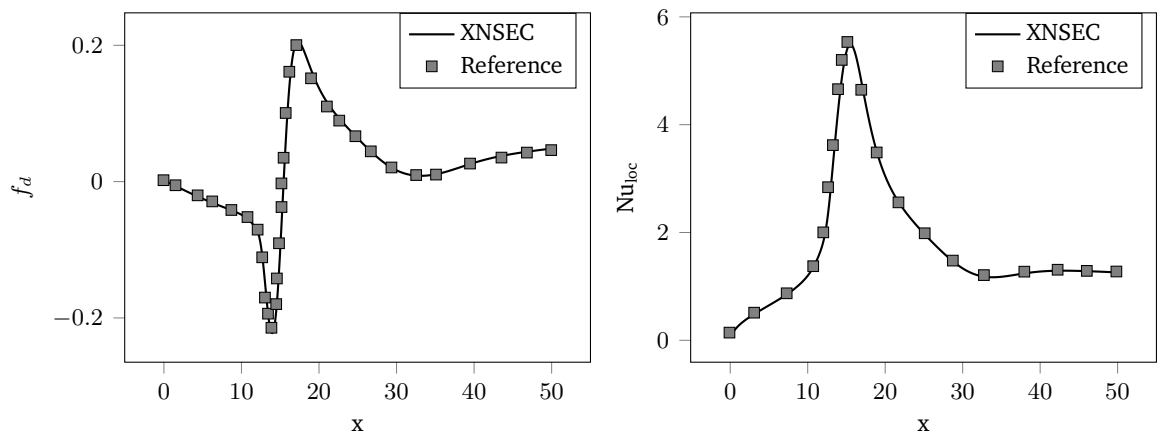


Figure 5.9: Local friction factor and local Nusselt number along the bottom wall of the backward-facing step for Re $= 700$ and an expansion ratio of two. The solid lines corresponds to our solution and the marks to the reference (Hennink, 2022) fig:fd_Nu_plot

For this range of temperature differences, the temperature profile is just influenced by conductive effects, since no appreciable natural convection phenomena appears. For larger temperature differences, Rayleigh-Bénard type instabilities would appear in the flow. This type of system will be treated later in Section 5.2.5.

It should be noted here that the results obtained using the XNSEC solver are substantially different from those reported by Xie and Xi (2016), and will not be shown here. However, in the work of Hennink (2022) the same is also reported, stating that with his method it was not possible to reproduce the results presented by Xie and Xi (2016).

In Figure 5.9 the local friction factor and local Nusselt number along the wall $y = 0$ are plotted for Re $= 700$ and ER $= 2$. Comparing the results from the XNSEC solver with those reported in Hennink (2022) a very good agreement can be observed. With this test it is possible to confirm that the XNSEC solver is able to deal with complex systems where heat transfer is present. However, for the range of temperature differences involved in this case, the variation of physical parameters such as density, viscosity and thermal conductivity with respect to temperature has no appreciable influence on the simulated flow fields. The next two test-cases will show how the XNSEC solver is able to simulate low-Mach number flows with a larger temperature difference.

5.2.2 Couette flow with vertical temperature gradient

ssec:CouetteFlowTempDiff

As a further test case for the low-Mach solver, a Couette flow with a vertical temperature gradient is considered. This configuration was already studied in Klein et al. (2016), where the SIMPLE algorithm was used in an DG framework for the solution of the governing equations. In this section, the results from said publication are reproduced by using the XNSEC solver, which features a fully coupled algorithm, in contrast to the SIMPLE solver, which solves the system in a segregated way. Additionally, it will be shown how the implemented solver performs in relation to the SIMPLE based solver in terms of runtime.In Figure 5.10 a schematic representation of the test case is shown. The domain is chosen as $\Omega = [0, 1] \times [0, 1]$, and Dirichlet boundary conditions are used for all boundaries. The upper wall corresponds to a

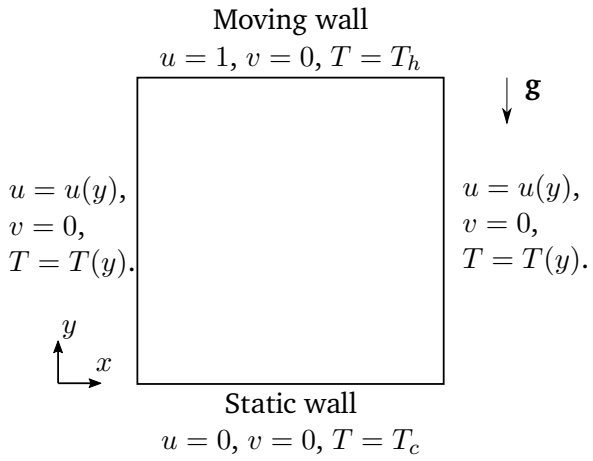


Figure 5.10: Schematic representation of the Couette flow with temperature difference test case. fig:CouetteTempDiff_scheme

moving wall ($u = 1$) with a fixed temperature $T = T_h$. The bottom wall is static ($u = 0$) and has a constant temperature $T = T_c$. Additionally, the system is subjected to a gravitational field, where the gravity vector only has a component in the y direction. Under these conditions, the x-component of velocity, pressure, and temperature are only dependent on the y coordinate, that is, $u = u(y)$, $T = T(y)$ and $p = p(y)$. The governing equations (Equation (2.1)) reduce to

$$\frac{1}{\mathrm{Re}}\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) = 0, \tag{5.5a}$$

$$\frac{\partial p}{\partial y} = -\frac{\rho}{\mathrm{Fr}^2}, \tag{5.5b}$$

$$\frac{1}{\mathrm{Re\,Pr}}\frac{\partial}{\partial y}\left(\lambda\frac{\partial T}{\partial y}\right) = 0. \tag{5.5c}$$

eq:AllCouetteEquations

By assuming a temperature dependence of the transport properties according to a Power Law ($\mu = \lambda = T^{2/3}$) it is possible to find an analytical solution for this problem.

$$u(y) = C_1 + C_2\left(y + \frac{T_c^{5/3}}{T_h^{5/3} - T_c^{5/3}}\right)^{3/5}, \tag{5.6a}$$

$$p(y) = -\frac{5p_0}{2\mathrm{Fr}^2}\frac{\left(y\left(T_h^{5/3} - T_c^{5/3}\right) + T_c^{5/3}\right)^{2/5}}{\left(T_h^{5/3} - T_c^{5/3}\right)} + C, \tag{5.6b}$$

$$T(y) = \left(C_3 - \frac{5}{3}C_4 y\right)^{3/5}. \tag{5.6c}$$

eq:AllCouetteSolutions

Where the constants C_1, C_2, C_3 and C_4 are determined using the boundary conditions on the upper and lower walls and are given by

$$C_1 = \frac{\left(\frac{T_c^{5/3}}{T_h^{5/3}-T_c^{5/3}}\right)^{3/5}}{\left(\frac{T_c^{5/3}}{T_h^{5/3}-T_c^{5/3}}\right)^{3/5} - \left(\frac{T_h^{5/3}}{T_h^{5/3}-T_c^{5/3}}\right)^{3/5}} \tag{5.7a}$$

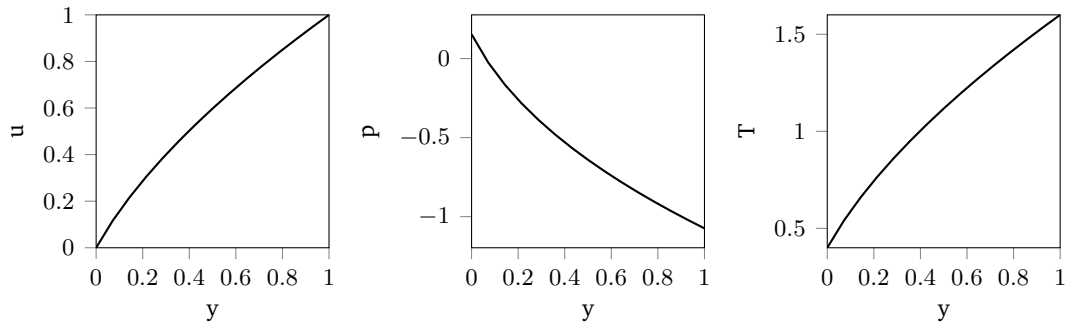


Figure 5.11: Solution of the Couette flow with vertical temperature gradient using a Power-Law. fig:CouetteSolution

$$C_2 = \frac{1}{\left(\frac{T_h^{5/3}}{T_h^{5/3}-T_c^{5/3}}\right)^{3/5} - \left(\frac{T_c^{5/3}}{T_h^{5/3}-T_c^{5/3}}\right)^{3/5}} \tag{5.7b}$$

$$C_3 = T_c^{5/3}, \tag{5.7c}$$

$$C_4 = \frac{3}{5}\left(T_c^{5/3} - T_h^{5/3}\right) \tag{5.7d}$$

and C is a real-valued constant determined by an arbitrary zero level for the pressure. The dimensionless parameters are set as $\mathrm{Re} = 10$ and $\mathrm{Pr} = 0.71$, $T_h = 1.6$, and $T_c = 0.4$ for all calculations. The system is considered open and the thermodynamic pressure is $p_0 = 1.0$. The Froude number is calculated as

$$\mathrm{Fr} = \left(\frac{2\mathrm{Pr}(T_h - T_c)}{(T_h + T_c)}\right)^{1/2}. \tag{5.8}$$ {eq:FroudeNumber1}

A derivation for Equation (5.8) will be given in Section 5.2.3. In Figure 5.11 the solutions for the velocity, pressure and temperature are shown. The results are for a mesh with 26×26 elements and a polynomial degree of three for u and T, and a polynomial degree of two for p. The vertical velocity v is zero everywhere.

h-convergence study

The convergence properties of the DG method for this nonisothermal system were studied using the analytical solution described before. The domain is discretized and solved in uniform Cartesian meshes with 16×16, 32×32, 64×64 and 128×128 elements. The polynomial degrees for the velocity and temperature are changed from one to four and for the pressure from zero to three. The convergence criterion described in Section 4.2.5 was used for all calculations. The analytical solutions given by Equation (5.6) are used as Dirichlet boundary conditions on all the boundaries of the domain. The global error is calculated against the analytical solution using a L^2 norm. In Figure 5.12 the results of the h-convergence study are shown. Recall that, for increasing polynomial order, the expected order of convergence is given by the slope of the line curve when cell length and errors are presented in a log-log plot. Due to the mixed-order formulation used, the slopes should be equal to k for the pressure and equal to $k + 1$ for all other variables. It is observed that the expected convergence rates are reached for all variables.

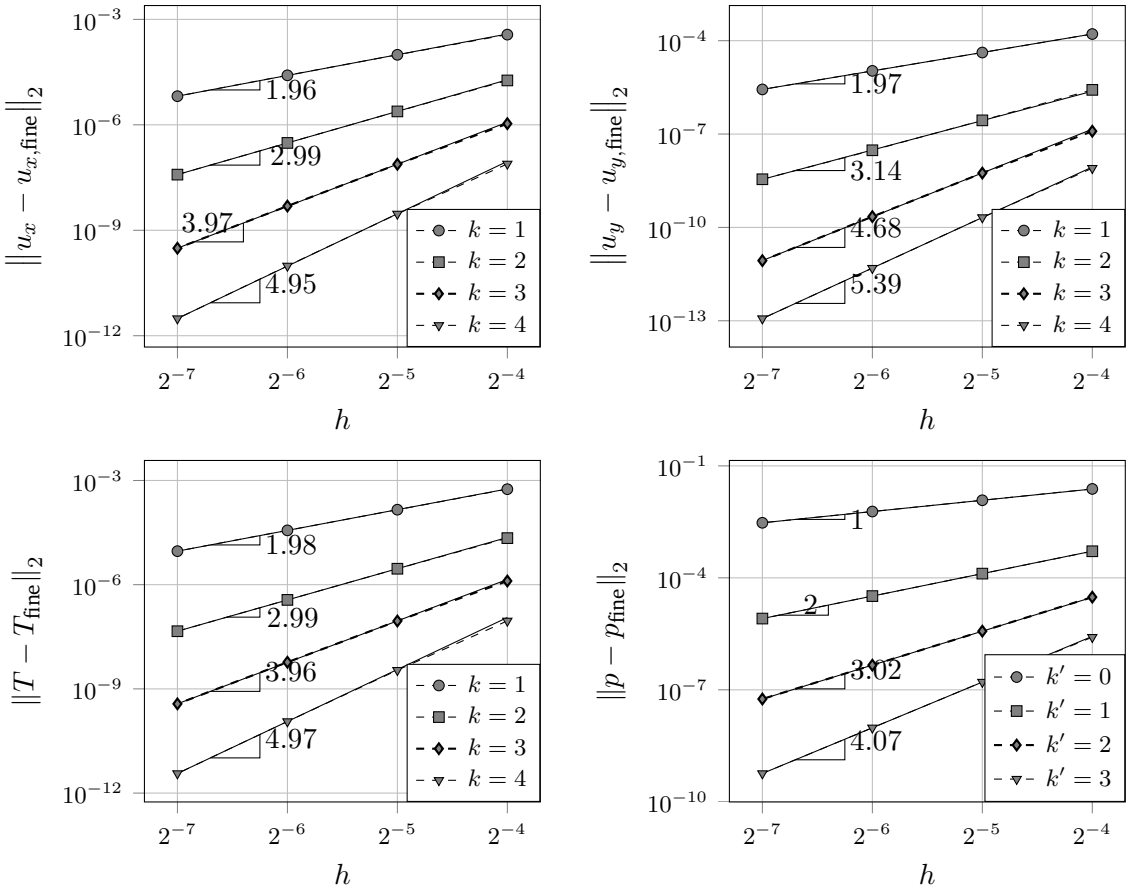


Figure 5.12: Convergence study of the Couette-flow with temperature difference. A power-law is used for the transport parameters.

fig:ConvergenceCFTD

Comparison with SIMPLE

As mentioned before, a solver for solving low-Mach number flows based on the SIMPLE algorithm presented in Klein et al. (2016) has already been developed and implemented within the BoSSS framework. Although the solver was validated and shown to be useful for a wide variety of test cases, there were also disadvantages inherent to the SIMPLE algorithm. For example, within the solution algorithm, Picard-type iterations are used to search for a solution. This usually requires some prior knowledge from the user in order to select suitable relaxation factor values that provide stability to the algorithm, but at the same time do not slow down the computation substantially. The intention of this subsection is to show a comparison of runtimes of the calculation of the Couette flow with vertical temperature gradient between the DG-SIMPLE algorithm (Klein et al., 2016) and the XNSEC solver. Calculations were performed on uniform Cartesian meshes with 16×16, 32×32, 64×64 and 128×128 elements, and with varying polynomial degrees between one and three for the velocity and temperature, and between zero and two for the pressure. All calculations where initialized with a zero velocity and pressure field, and with a temperature equal to one in the whole domain. Are calculations were performed single-core. The convergence criteria of the nonlinear solver is set to 10^{-8} for both solvers. The under-relaxation factors for the SIMPLE algorithm are set for all calculations

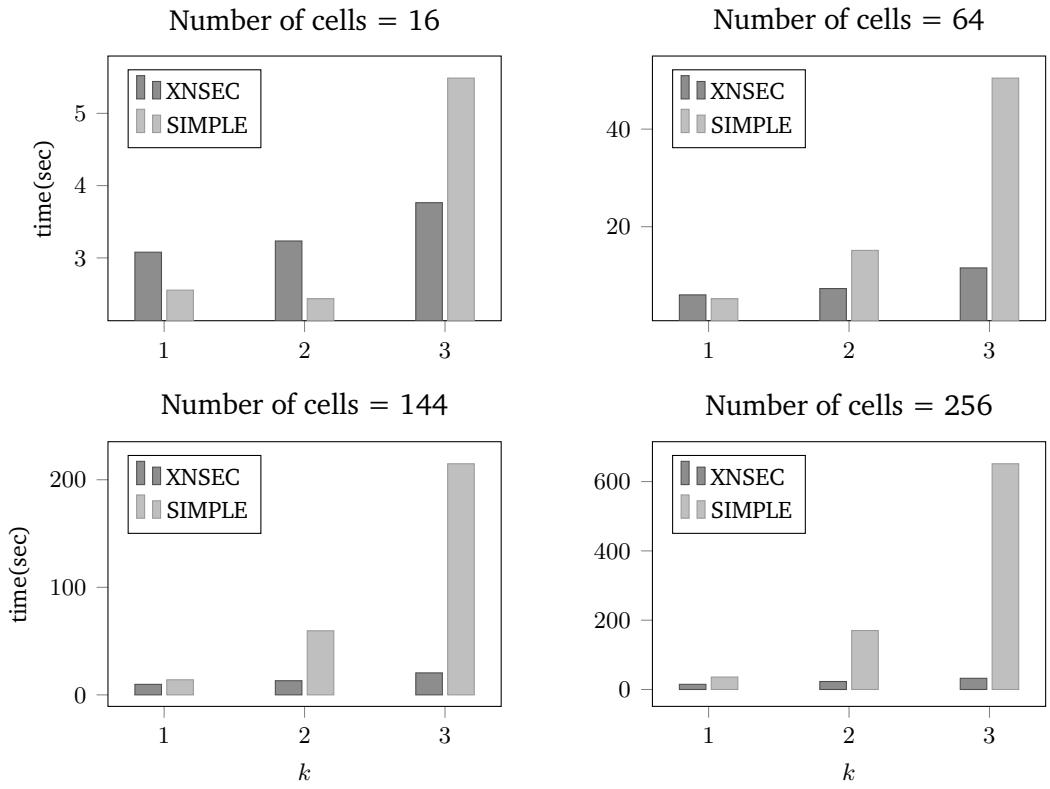


Figure 5.13: Runtime comparison of the DG-SIMPLE solver and the XNSEC solver for the Couette flow with vertical temperature gradient for different polynomial degrees k and number of cells.
Fig:RuntimeComparison

to 0.8, 0.5 and 1.0 for the velocity, pressure and temperature, respectively.

In Figure 5.13 a comparison of the runtimes from both solvers is shown. It is clearly appreciated how the runtimes of the SIMPLE algorithm are higher for almost all of the cases studied. Only for systems with a small number of cells the solver using the SIMPLE algorithm outperforms the XNSEC solver. It is also interesting to note how the runtime of the simulations with the XNSEC code seems to scale linearly with the polynomial degree. On the other hand, in the case of the SIMPLE solver, the scaling is much more unfavourable, and the runtime increases dramatically as the polynomial degree increases. Obviously the under-relaxation parameters of the SIMPLE algorithm have an influence on the calculation times and an appropriate selection of them could decrease the runtimes. This is a clear disadvantage, since the adequate selection of under-relaxation factors is highly problem dependent and requires previous expertise from the user. On the other hand, the globalized Newton method used by the XNSEC solver avoid this problem by using a more sophisticated method and heuristics in order to find a better path to the solution.

5.2.3 Differentially heated cavity problem

ss:DHC

The differentially heated cavity problem is a classical benchmark case that is often used to assess the ability of numerical codes to simulate variable density flows (Paillere et al., 2000; Vierendeels et al., 2003; Tyliszczak, 2014). The test case has the particularity that deals with

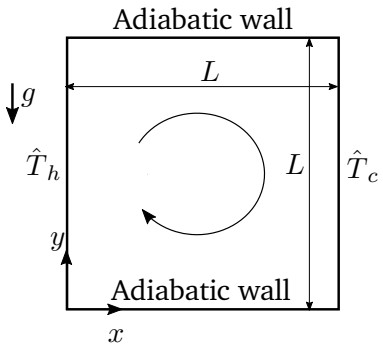


Figure 5.14: Schematic representation of the differentially heated cavity problem. DHCGeom

a closed system, where the thermodynamic pressure p_0 is a parameter that must be adjusted so that the mass is conserved. The thermodynamic pressure p_0 determines the density field, which in turn appears in the momentum equation and the energy equation, making it necessary to use an adequate algorithm to solve the system. This point presents a special difficulty for the solution, since the calculation of p_0 requires knowledge of the temperature field on the whole computational domain, inducing a global coupling of the variables.

The system is a fully enclosed two-dimensional square cavity filled with fluid. A sketch of the problem is shown in Figure 5.14. The left and right walls of the cavity have constant temperatures $\hat{T}_h$ and $\hat{T}_c$, respectively, with $\hat{T}_h > \hat{T}_c$, and the top and bottom walls are adiabatic. A gravity field induces fluid movement because of density differences caused by the difference in temperature between the hot and cold walls. The natural convection phenomenon is characterized by the Rayleigh number, defined as

$$\mathrm{Ra} = \mathrm{Pr}\frac{\hat{g}\hat{\rho}_{\mathrm{ref}}^2(\hat{T}_h - \hat{T}_c)\hat{L}_{\mathrm{ref}}^3}{\hat{T}_{\mathrm{ref}}\hat{\mu}_{\mathrm{ref}}^2}, \tag{5.9}$$ {eq:Rayleigh}

For small values of Ra, conduction dominates the heat transfer process, and a boundary layer covers the entire domain. On the other hand, large values of Ra represent a flow dominated by convection. When the number Ra increases, a thinner boundary layer is formed. Following Vierendeels et al. (2003), a reference velocity for buoyancy-driven flows can be defined as

$$\hat{u}_{\mathrm{ref}} = \frac{\sqrt{\mathrm{Ra}}\hat{\mu}_{\mathrm{ref}}}{\hat{\rho}_{\mathrm{ref}}\hat{L}_{\mathrm{ref}}}. \tag{5.10}$$

The Rayleigh number is then related to the Reynolds number according to

$$\mathrm{Re} = \sqrt{\mathrm{Ra}}. \tag{5.11}$$

Thus, it is sufficient to select a Re number in the simulation, fixing the value of the Ra number. The driving temperature difference $(\hat{T}_h - \hat{T}_c)$ appearing in Equation (5.9) can be represented as a nondimensional parameter:

$$\varepsilon = \frac{\hat{T}_h - \hat{T}_c}{2\hat{T}_{\mathrm{ref}}}. \tag{5.12}$$ {eq:nondimensionalTemperature}

Using these definitions, the Froude number can be calculated as

$$\mathrm{Fr} = \sqrt{\mathrm{Pr} 2\varepsilon}. \tag{5.13}$$

The results of the XNSEC solver are compared with those of the reference solution for $\hat{T}_{\mathrm{ref}} = 600\mathrm{K}$ and $\varepsilon = 0.6$. All calculations assume a constant Prandtl number equal to 0.71. The dependence of viscosity and heat conductivity on temperature is calculated using Sutherland's law (Equation (2.29)). The nondimensional length of the cavity is $L = 1$. The nondimensional temperatures T_h and T_c are set to 1.6 and 0.4, respectively. The nondimensional equation of state (Equation (2.26)) depends only on the temperature and reduces to

$$\rho = \frac{p_0}{T}. \tag{5.14}$$

The thermodynamic pressure p_0 in a closed system must be adjusted to ensure mass conservation. For a closed system is given by

$$p_0 = \frac{\int_\Omega \rho_0 \mathrm{d}V}{\int_\Omega \frac{1}{T} \mathrm{d}V} = \frac{m_0}{\int_\Omega \frac{1}{T} \mathrm{d}V}, \tag{5.15}$$ {eq:p0Condition}

where Ω represents the complete closed domain. The initial mass of the system m_0 is constant and is set $m_0 = 1.0$. Note that the thermodynamic pressure is a parameter with a dependence on the temperature of the entire domain. This makes necessary the use of an iterative solution algorithm, so that the solution obtained respects the conservation of mass. Within the solution algorithm of the XNSEC solver, Equation (5.15) is used to update the value of the thermodynamic pressure after each Newton iteration. The average Nusselt number is defined for a given wall Γ as

$$\mathrm{Nu}_\Gamma = \frac{1}{T_h - T_c} \int_\Gamma k \frac{\partial T}{\partial x} \mathrm{d}y. \tag{5.16}$$ {eq:Nusselt}

Comparison of results with the benchmark solution

Here a comparison of the results obtained with the XNSEC solver and the results presented in the work of Vierendeels et al. (2003) is made. Vierendeels et al. (2003) solved the fully compressible Navier-Stokes equations on a stretched grid with 1024×1024 using a finite-volume method with quadratic convergence, providing very accurate results that can be used as reference. The benchmark results are presented for $\mathrm{Ra} = \{10^2, 10^3, 10^4, 10^5, 10^6, 10^7\}$. In this range of Rayleigh numbers, the problem has a steady-state solution. The cavity is represented by the domain $[0, 1] \times [0, 1]$. For all calculations in this subsection, the simulations are done with a polynomial degree of four for both velocity components and temperature and three for the pressure. The mesh is in an equidistant 128×128 mesh.

Preliminary calculations showed that for cases up to $\mathrm{Ra} = 10^5$ the solution of the system using Newton's method presented in Section 4.2.2 is possible without further modifications, while for higher values the algorithm couldn't find a solution and stagnates after certain number of iterations. The homotopy strategy mentioned in Chapter 4 is used to overcome this problem and obtain solutions for higher Rayleigh numbers. Here, the Reynolds number is selected as the homotopy parameter and continuously increased until the desired value is reached.

In Figures 5.16 to 5.18 the temperature and velocity profiles across the cavity for different Rayleigh numbers are shown. The profiles calculated with the XNSEC solver agree closely to the benchmark solution. As expected, an increase of the acceleration of the fluid in the vicinity of the walls for increasing Rayleigh numbers is observed.

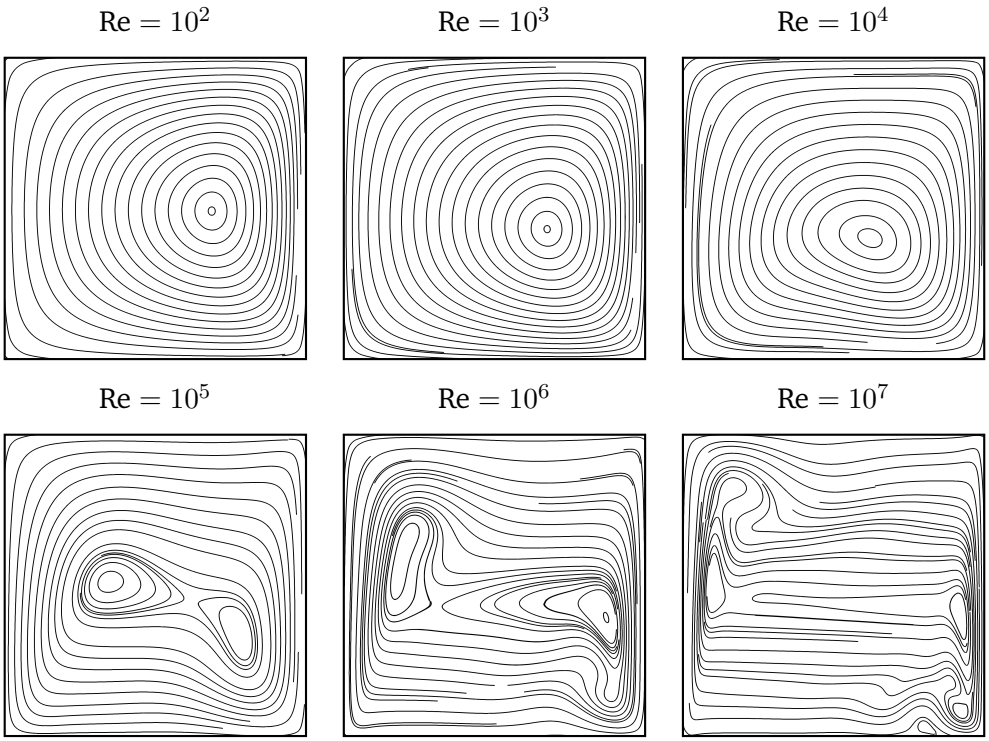


Figure 5.15: Streamlines of the heated cavity configuration with $\epsilon = 0.6$ for different Reynold numbers. fig:NSCStreamlines

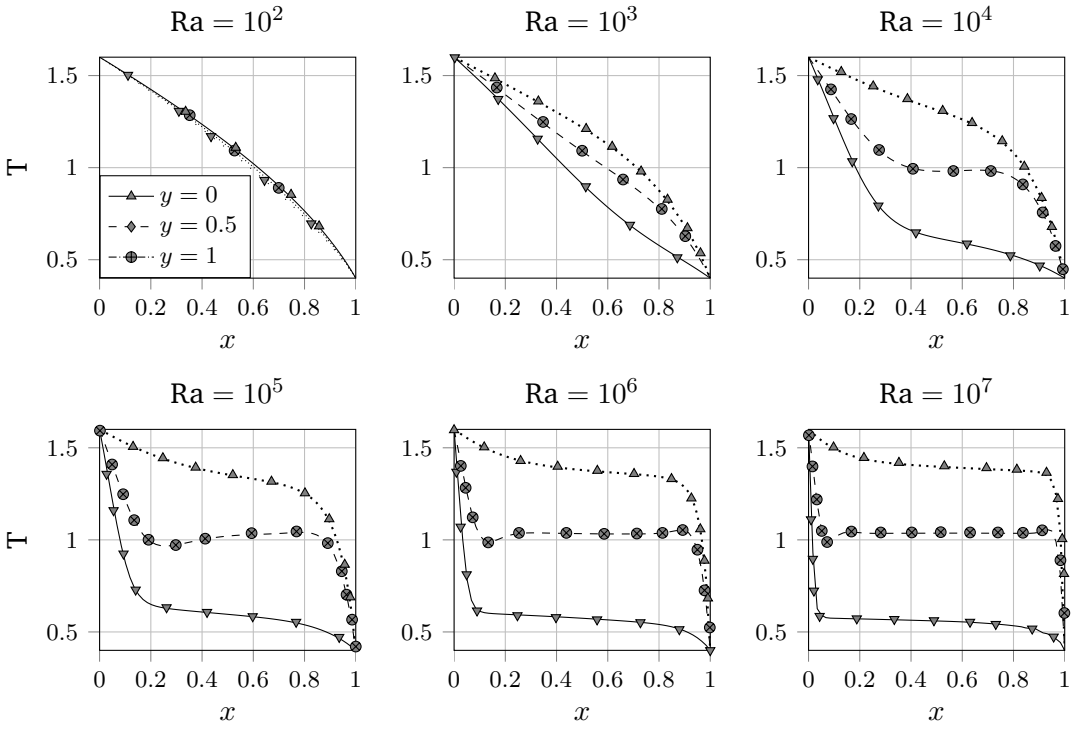


Figure 5.16: Temperature profiles for the differentially heated square cavity along different vertical levels. Solid lines represent the XNSEC solver solution and marks the benchmark solution.

fig:TempProfile

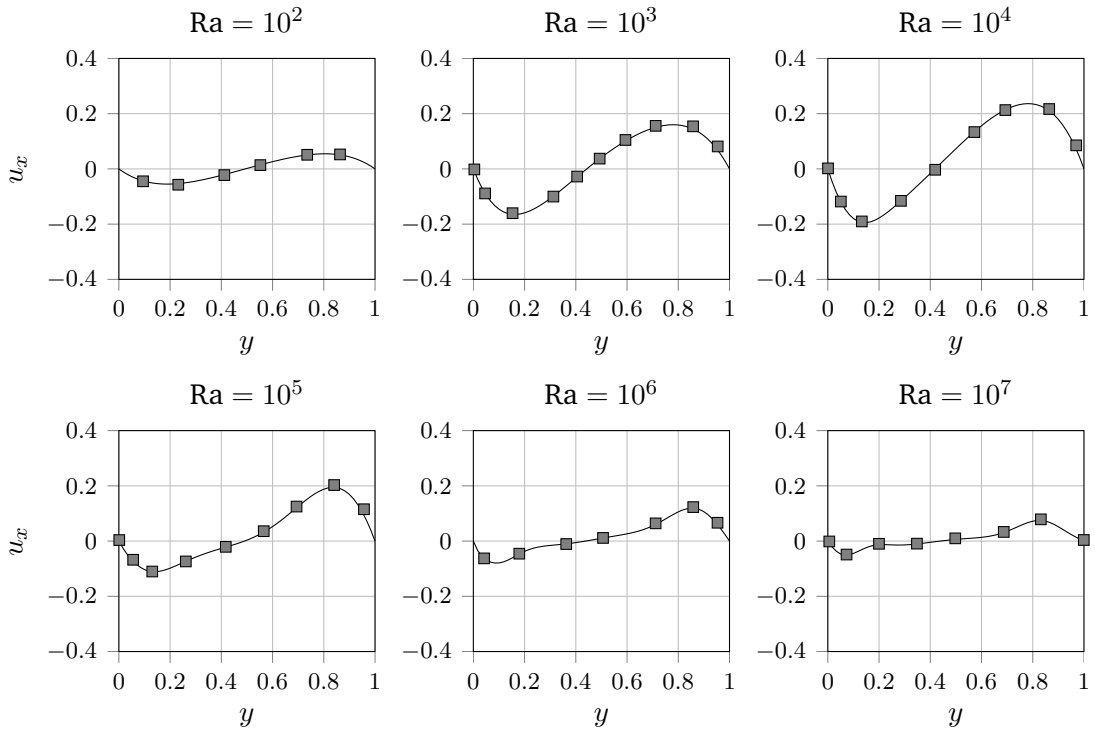


Figure 5.17: Profiles of the x-velocity component for the differentially heated square cavity along the vertical line $x = 0.5$. Solid lines represent the XNSEC solver solution and marks the benchmark solution.

fig:VelocityXProfile

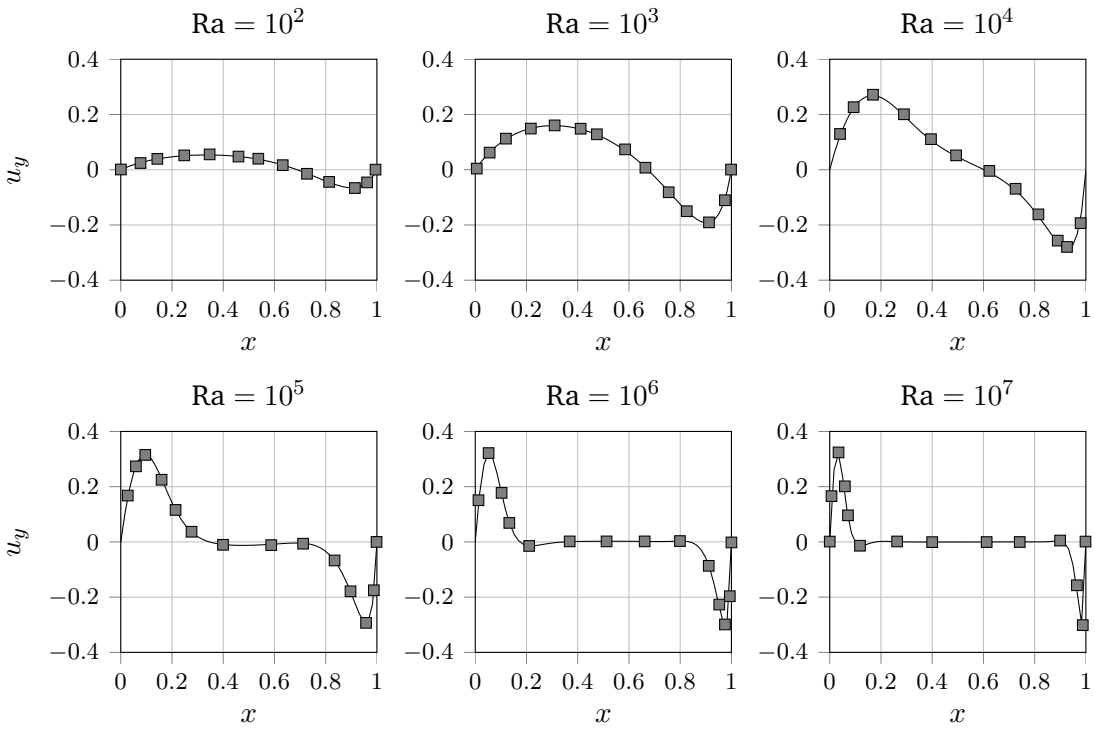


Figure 5.18: Profiles of the y-velocity component for the differentially heated square cavity along the horizontal line $y = 0.5$. Solid lines represents the XNSEC solver solution and marks the benchmark solution.

fig:VelocityYProfile

Rayleigh	p_0	$p_{0,\text{ref}}$	Nu_h	Nu_c	Nu_{ref}
10^2	0.9574	0.9573	0.9787	0.9787	0.9787
10^3	0.9381	0.9381	1.1077	1.1077	1.1077
10^4	0.9146	0.9146	2.2180	2.2174	2.2180
10^5	0.9220	0.9220	4.4801	4.4796	4.4800
10^6	0.9245	0.9245	8.6866	8.6791	8.6870
10^7	0.9225	0.9226	16.2411	16.1700	16.2400

Table 5.2: Comparison of calculated Nusselt numbers of the hot and cold wall and Thermodynamic pressure p_0 reported values by Vierendeels et al. (2003) for the differentially heated cavity. tab:p0_Nu_Results

A comparison of the thermodynamic pressure and the Nusselt numbers to the benchmark solution was also made. The results are shown in Table 5.2. The thermodynamic pressure is obtained from Equation (5.15), and the average Nusselt number is calculated with Equation (5.16). The results obtained with the XNSEC solver agree very well with the reference results, as can be seen for the thermodynamic pressure, which differs at most in the fourth decimal place. Note that the average Nusselt number of the heated wall Nu_h and the Nusselt number of the cold wall Nu_c are different. As the Rayleigh number grows, this discrepancy becomes larger, hinting that, at such Rayleigh numbers, the mesh used is not refined enough to adequately represent the thin boundary layer and more complex flow structures appearing in high-Rayleigh number cases. While for an energy conservative system Nu_h and Nu_c should be equal, for the DG-formulation this is not the case, since conservation is only ensured locally and the global values can differ. This discrepancy can be seen as a measure of the discretization error of the DG formulation and should decrease as the mesh resolution increases. This point will be discussed in the next section.

Convergence study

ssec:ConvStudyHeatedCavity

An $h-$convergence study of the XNSEC solver was conducted using the heated cavity configuration. Calculations were performed for polynomial degrees $k = 1, 2, 3, 4$ and equidistant regular meshes with, respectively, 8×8, 16×16, 32×32, 64×64, 128×128 and 256×256 elements. The L^2 -Norm was used to calculate errors against the solution in the finest mesh. The results of the h-convergence study for varying polynomial orders k are shown in Figure 5.19. It is observed how the convergence rates scale approximately as $k + 1$. Interestingly, for $k = 2$ the rates are higher than expected. On the other hand, some degeneration is observed in convergence rates for $k = 4$. This strange behavior can be explained if one considers that the heated cavity presents a singular behavior at the corners (similar to the problem previously exposed for the lid-driven cavity), which causes global pollution in the convergence behavior of the algorithm.

As discussed in the previous section, the difference in the average values of the Nusselt number on the hot wall Nu_h and the cold wall Nu_c is a direct consequence of the spatial discretization error and should decrease for finer meshes. In Figure 5.20 the convergence behavior of the Nusselt number is presented for different polynomial degrees k, different number of elements and for two Rayleigh numbers. As expected, it can be observed that this discrepancy is smaller when a larger number of elements is used. It can also be seen that

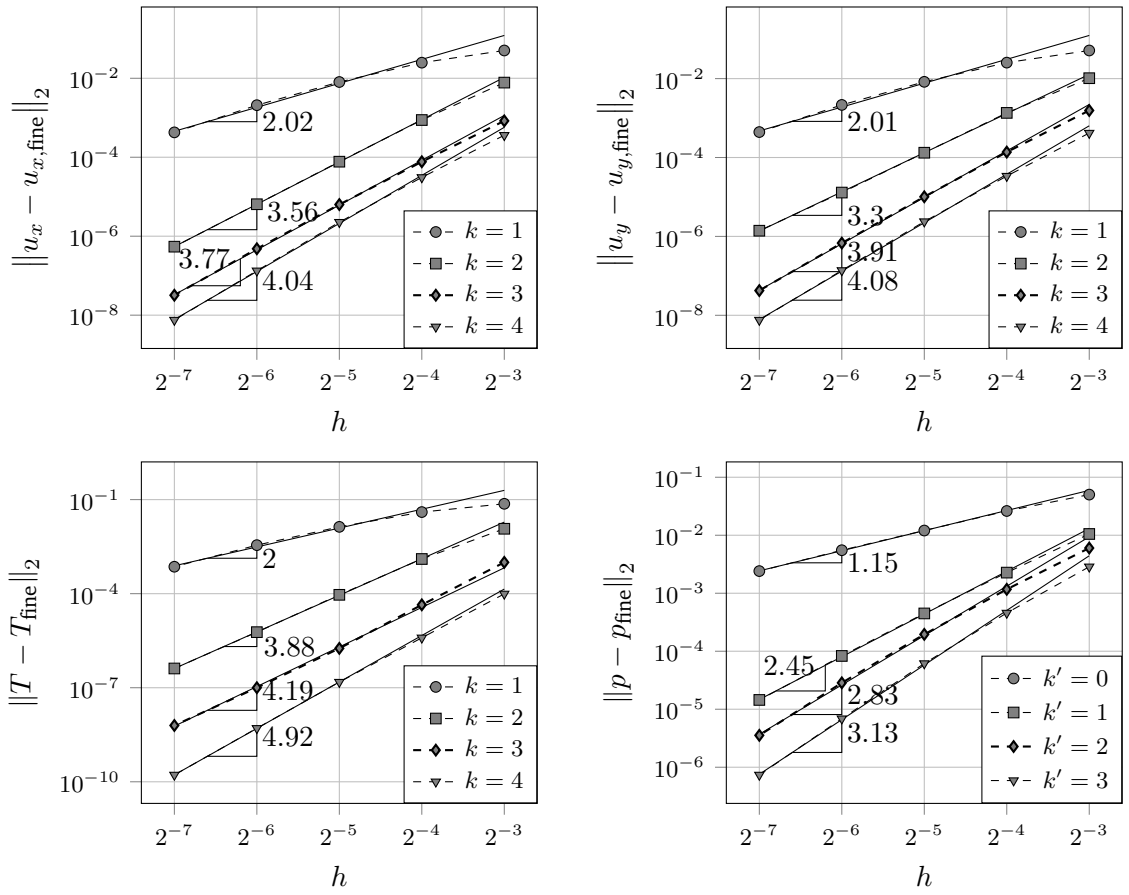


Figure 5.19: Convergence study of the differentially heated cavity problem for $\text{Ra} = 10^3$. fig:ConvergenceDHC

Nu_h reaches the expected solution of cells for a much smaller number of elements. This can be explained if one thinks that more complex phenomena take place near the cold wall (see Figure 5.15), which makes necessary a finer mesh resolution in that area.

The results presented in this section allow us to conclude that the implemented solver is capable of dealing with flows with variable densities, and in particular in closed spaces. Additionally, it was observed that even for this complex test, convergence properties close to those expected from the DG-method are obtained. In this section only systems with a steady state solution were treated. In the next section the ability of the solver to compute flows with varying densities in non-steady state will be shown.

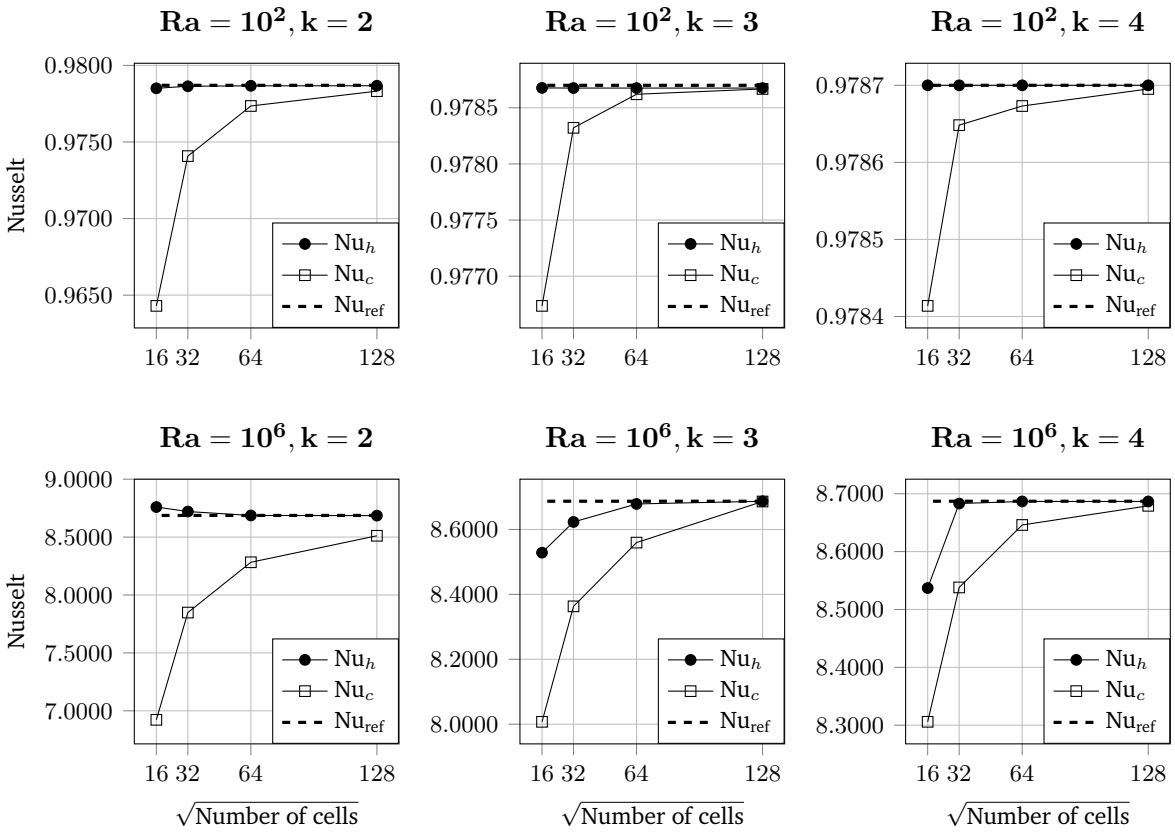


Figure 5.20: Nusselt numbers of the differentially heated square cavity at the hot wall (Nu_h) and the cold wall (Nu_c) for different number of cells and polynomial order k. The reference values from Vierendeels et al. (2003) are shown with dashed lines. fig:NusseltStudy

Influence of the penalty factor

A point not still discussed is the choice of the safety parameter η_0 of the penalty terms from the SIP discretization (see Section 3.3.3). Table 5.3 shows results obtained for $\text{Ra} = 10^3$, for different polynomial degrees and penalty safety factor. For the tests presented here, the penalty terms of the diffusive terms from the momentum and energy equations are considered equal. Furthermore, the number of elements in the mesh is selected in such a way that the number of degrees of freedom remains approximately constant for each simulation.

It is possible to see that the penalty safety factor (and therefore the penalty term) can have a great influence on the solution. If the value chosen is very small, as in the case of the table for $\eta_0 = 0.01$, the algorithm is not able to find a solution. On the other hand, if the chosen value is too high, the error also increases. It can be concluded that an optimal value for the penalty factor exists.

It is also noticeable that, maintaining a constant penalty safety factor, increasing the polynomial degree for an approximately constant number of DOFs gives an improvement in the results compared to the literature. Although for this testcase the effect of the penalty factor on the solution is not very large, the effect could be considerable, especially when dealing

η_0	DoFs	k	Nu_c	$\frac{\mathrm{Nu}_c - \mathrm{Nu}_{c,\mathrm{ref}}}{\mathrm{Nu}_{c,\mathrm{ref}}} \times 10^2$	p_0	$\frac{p_0 - p_{0,\mathbf{ref}}}{p_{0,\mathbf{ref}}} \times 10^4$
	2	6804	0.549483*	50.39424	0.899757*	408.7292
0.01	3	7056	0.722593*	34.76637	0.936085*	21.48436
	4	6655	-0.50954*	146	1.016691*	837.7683
	2	6804	1.090047	1.593667	0.938192	0.980674
1	3	7056	1.102072	0.508037	0.938057	0.453674
	4	6655	1.105225	0.22348	0.938046	0.570494
	2	6804	1.089332	1.65817	0.93843	3.521926
4	3	7056	1.102261	0.491027	0.938076	0.25384
	4	6655	1.105359	0.211372	0.938047	0.561709
	2	6804	1.08694	1.874166	0.939109	10.75641
16	3	7056	1.102266	0.490563	0.938124	0.251627
	4	6655	1.105439	0.204153	0.93805	0.537265

Table 5.3: Thermodynamic pressure and cold-side Nusselt number for different penalty safety factors in a heated cavity with $\mathrm{Ra} = 10^3$. Values marked with an asterisk are from problems not converged after 100 iterations.

fig:EtaInfluence

with more complex geometries and coarser meshes. The value $\eta_0 = 4$ has shown to be a value that gives stability to the scheme and is used for all simulations in this thesis, as already has been done in many works (Krause and Kummer, 2017; Kummer, 2017; Smuda, 2021) and is used for all calculations in this work.

5.2.4 Flow over a circular cylinder

ssec:FlowCircCyl

A further thest case of the XNSEC solver is the simulation of flow over a circular cylinder. The results shown section are based on the work by Miao (2022)

The simulation of a 2D flow over an obstacle exhibits different flows states depending on the Reynolds number. For low Reynolds numbers, the flow is laminar and stationary. When the Reynolds number reaches a critical value, the flow becomes non-stationary and eventually turbulent. In particular, for $50 \leq \mathrm{Re} \leq 160$ the behavior of the fluid is laminar with an non-stationary periodic character, and the well-known vortex shedding phenomenon appears (Sharma and Eswaran, 2004). In this section the ability of the XNSEC solver for dealing with isothermal and non-isothermal unsteady flow will be assessed. This is done by calculating the Strouhal number, average drag and lift coefficients and the Nusselt number and comparing them to reference solutions.

Set-up

Figure 5.21 shows the geometry of the problem. In the center of the domain is located a heated cylinder of diameter $d = 1$, which is subjected to a flow of constant velocity magnitude. The size of the computational domain is chosen as $D = 59d$, which is large enough so that the outlet boundary conditions do not influence the solution. The left half of the boundaries of the computational domain corresponds to a velocity inlet with constant velocity $(u, v) = (1, 0)$ and temperature $T = T_\infty$. The right half of the domain corresponds to a pressure outlet. The boundary condition corresponding to the heated cylinder is a no-slip wall defined with

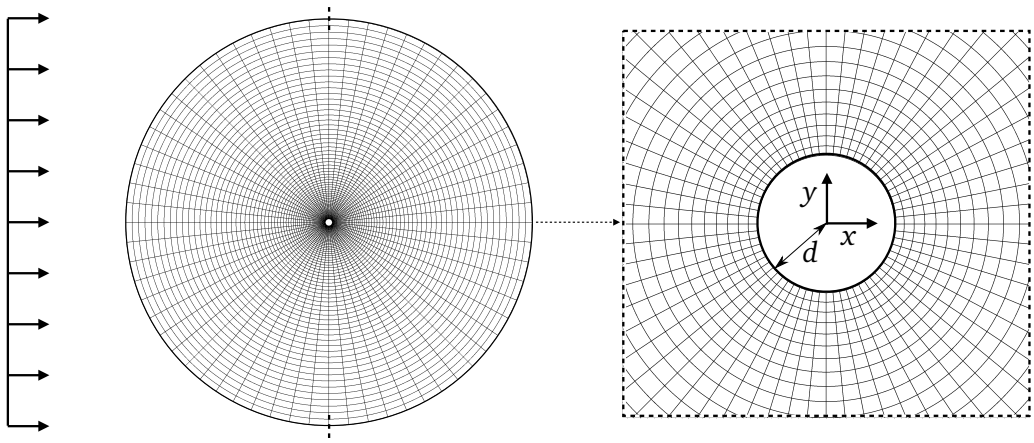


Figure 5.21: Schematic representation of the heated circular cylinder. Figure adapted from (Miao, 2022)
fig:CircularCylinderGeom

$(u, v) = (0, 0)$, and with a constant temperature $T = T_h$. The system is an open system, and the thermodynamic pressure has a constant value set to $p_0 = 1$.

As mentioned, a isothermal and non-isothermal system is considered in this section. A reference temperature of $\hat{T}_{\text{ref}} = 298.15\,\text{K}$ is selected. The conditions for the velocity inlet are for the isothermal case a dimensionless temperature of $T_h = 1.0$ and $T_\infty = 1$, while for the non-isothermal $T_h = 1.5$ and $T_\infty = 1.0$. A Reynolds number of 100 is used, for which the flow is known to be unsteady and periodic. Gravity effects are taken as negligible. The prandtl number has a constant value of $\text{Pr} = 0.71$ and the heat capacity ratio $\gamma = 1.4$. The transport parameters are calculated with Sutherlands law which is given by Equation (2.29) and the equation of state for the density is Equation (2.26).

For all calculations a polynomial degree of order three is used for both velocity components and the temperature, and two for pressure. In this simulation a curved mesh consisting of 64 elements in the radial direction and 64 elements in the angular direction is used. The use of a curved mesh for this problem was done primarily to adequately represent the geometry of the heated cylinder. Another possibility could have been to use a method such as a Immersed Boundary Method (which is also possible to do in the BoSSS-framework), but it is beyond the scope of this work.

One point to note regarding the use of curved meshes within the DG-method (and in all High-order methods), is that special care has to be taken in the representation of curved elements in order to use correctly cuadrature rules (Bassi and Rebay, 1997). This is critical to preserve the convergence properties of DG-methods. In particular, since the highest degree of the polynomials used in this simulation is three, elements of the bi-cubic type are used, where 16 nodes are used per element for the discretization for a two-dimensional element. Furthermore, the time discretization is performed with a BDF-3 scheme, and the time derivative of the continuity equation is calculated with a second order backward difference scheme (see Section 3.2.3). The simulation time corresponds to $t = 100$, and constant timesteps of $\Delta t = 0.2$ are used. The initial conditions are

$$u(t = 0) = 1 + u^{\text{vortex}}, \tag{5.17a}$$
$$v(t = 0) = 0 + v^{\text{vortex}}, \tag{5.17b}$$

$$T(t=0) = 1, \tag{5.17c}$$
$$p(t=0) = 0. \tag{5.17d}$$

Here, u^{vortex} and u^{vortex} are the velocity component of a vortex field of radius r, strength a and central point (x^o, y^o) defined by

$$u^{\text{vortex}}(x,y) = \begin{cases} -a(y-y^0) & \text{if} \quad \sqrt{(x-x^0)^2+(y-y^0)^2} \leq r \\ 0 & \text{if} \quad \sqrt{(x-x^0)^2+(y-y^0)^2} > r \end{cases} \tag{5.18a}$$

eq:VortexU

$$v^{\text{vortex}}(x,y) = \begin{cases} a(x-x^0) & \text{if} \quad \sqrt{(x-x^0)^2+(y-y^0)^2} \leq r \\ 0 & \text{if} \quad \sqrt{(x-x^0)^2+(y-y^0)^2} > r \end{cases} \tag{5.18b}$$

eq:VortexV

The reason for placing a vortex in the initial conditions is to include a perturbation in the system that triggers the vortex schedding phenomenon. The vortex moves with the flow and is eventually advected from the calculation domain. Specifically, for this simulation, a vortex of radius $r = 1$ and strength $a = 1$ is placed at $(x^o, y^o) = (2, 0)$. It is worth mentioning that the inclusion of the vortex in the initial conditions is not imperative to make the vortex-shedding phenomenon emerge, but it is a way to accelerate its appearance. For the range of Reynolds numbers mentioned, even instabilities of the numerical method should be enough to cause the phenomenon to arise, but in a much slower way.

The variables lift coefficient C_L, drag coefficient C_D, Strouhal number St, and Nusselt number Nu are calculated and compared with reference results. These characteristic quantities are defined as

$$C_L = \frac{2F_L}{\rho_\infty u_\infty^2 d} \tag{5.19}$$

$$C_D = \frac{2F_D}{\rho_\infty u_\infty^2 d} \tag{5.20}$$

$$\text{St} = \frac{fd}{u_\infty} \tag{5.21}$$

$$\text{Nu} = \frac{d}{\Delta T} \frac{1}{\|\partial S\|_{\textbf{leb}}} \oint_{\partial S} \nabla T \cdot \mathbf{n} dS \tag{5.22}$$

F_L and F_D are the lift and drag force respectively, f is the vortex shedding frequency and $\|\partial S\|_{\textbf{leb}} = \pi d$ is the circunference of the cylinder.

Isothermal case.

The isothermal case results are compared with the work of Sharma and Eswaran (2004). They reported results for the case $\text{Re} = 100$ and $T_h = T_\infty$. The Strouhal number and average drag coefficient calculated with the XNSEC solver are $\text{St} = 0.1639$ and $C_{D,\textbf{avg}} = 1.3103$, while the reference values $\text{St} = 0.164$ and $C_{D,\textbf{avg}} = 1.3183$, which means a error of less than a 0.6% on both quantities. Its worth noting that preliminary calculations with a implicit Euler scheme for the discretization of the temporal terms didn't cause the vortex shedding phenomenon to appear, making necessary a BDF-3 scheme.

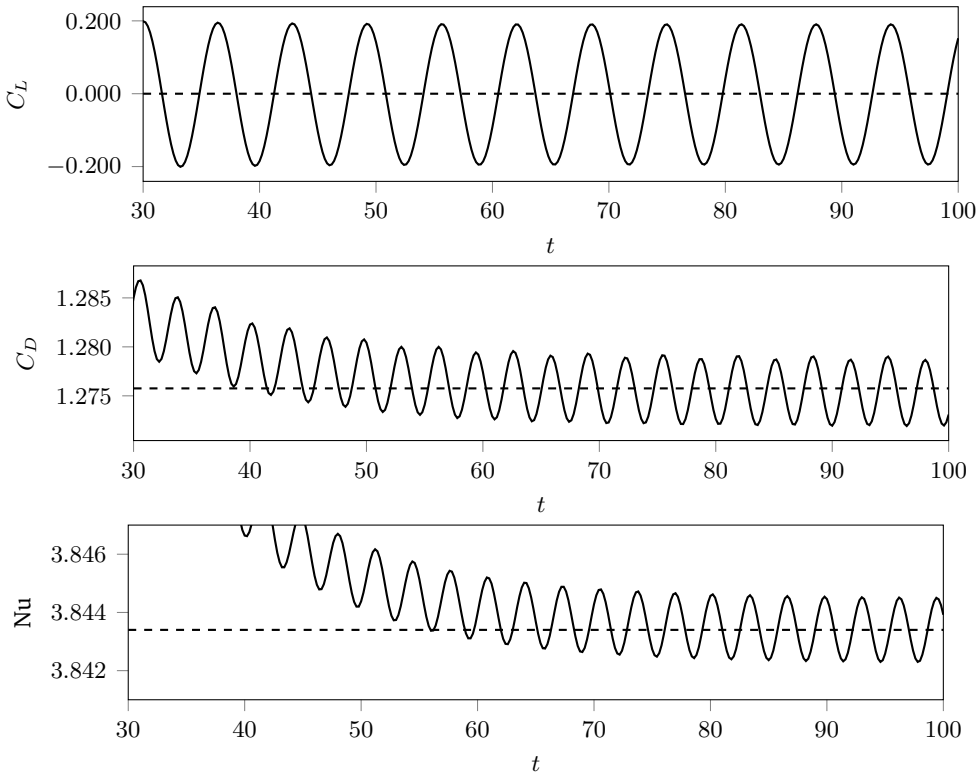


Figure 5.22: Temporal evolution of average Nusselt number, lift coefficient and drag coefficient of the heated cylinder

fig:HeatedCylinderResults

Non-isothermal case.

A comparison of the calculation results for the non-isothermal case was performed based on the results reported in Shi et al. (2004), Wang et al. (2000) and Hennink (2022). The temporal evolution of the characteristic quantities C_D, C_L and Nu is shown in Figure 5.22. The results agree very good with the references. The average Nusselt number is Nu $= 3.8434$, while the reference value from Hennink (2022) is Nu $= 3.804$, which corresponds to a difference of 1.04%. The Strouhal number is St $= 0.1538$ and the references report values of St $= 0.152$ and St $= 0.1536$, a difference of approximately 1%. It is possible then to conclude that the XNSEC solver allows to simulate adequately unsteady non-isothermal flows, where the fluid properties do present a scalar dependence (in this case, on the temperature).

It is worth noting that in this test-case the temperature changes causes only a moderate variation in the density. As will be seen later, the temporal term appearing in the continuity equation, as implemented in the present solver, is problematic in the case of larger density differences (Knikker, 2011), as it is a cause of numerical instabilities.

5.2.5 Rayleigh-Bénard Convection

ssec:RayBer

The Rayleigh-Bénard convection is a configuration similar to the heated cavity (see Section 5.2.3) as the flow is also induced by bouyancy effects. The configuration consists of a fluid located between two horizontal plates maintained at different temperatures, with the

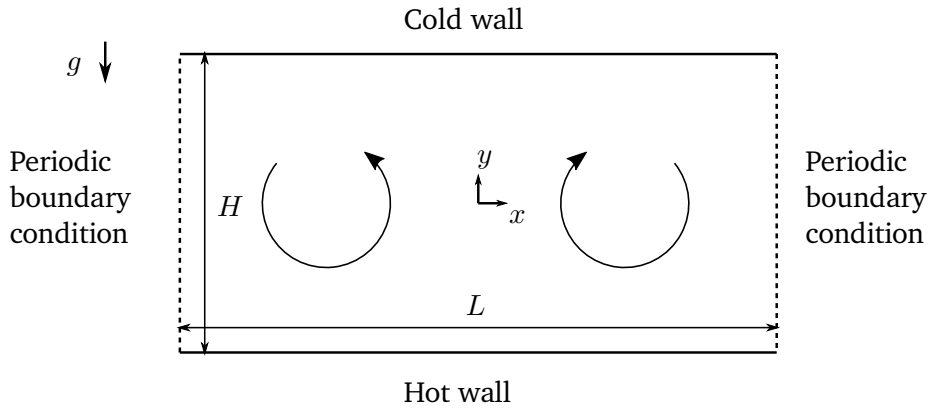


Figure 5.23: Geometry of the Rayleigh-Bénard convection problem. Convection rolls are sketched. fig:RayBenGeometryPeriodic

temperature of the lower plate being higher than that of the upper plate. It is known that under a certain Rayleigh number the flow presents a macroscopically stable behavior and the heat is simply transferred by conduction. When the Rayleigh number exceeds a certain critical value Ra_{crit}, the system becomes unstable, which causes a perturbation of the system to give rise to fluid motion, creating the so-called convection cells, also usually referred to as Bénard cells.

In the following sections two aspects of this situation will be discussed by making use of different types of boundary conditions on the sides of the domain. First in Section 5.2.5 periodic boundary conditions are used, which allows studying a single pair of convection rolls and to calculate numerically the critical value Ra_{crit}, which is compared with theoretical values. Subsequently, in Section 5.2.5 a transient simulation of this configuration is performed by using pressure-outlet boundary conditions. The results shown in this section are taken from the work by Miao (2022).

Periodic boundary conditions

ssec:SingleCellConv

In this section the fluid behavior will be analysed by studying a system where a single convection cell appears. Making use of linear stability theory for the governing equations under the Boussinesq assumption, it is possible to determine that for a system with rigid boundaries the critical Rayleigh number is $\text{Ra}_{\text{crit}} = 1707.762$ (Chandrasekhar, 1961). Moreover, the dimensionless wave number is $a_c = 3.117$, which implies that the convective rolls develop at an aspect ratio of $2\pi/a_c = 2.016$. For the simulations presented in this section an aspect ratio of $L/H = 2$ is used, as is done by Kao and Yang (2007). The geometry and boundary conditions of the analysed problem can be found in Figure 5.23. The upper wall corresponds to a no-slip wall that is maintained at a constant temperature T_c. Similarly, the bottom wall is also a no-slip wall with a temperature T_h, where $T_h > T_c$. The boundary conditions on the left and right of the computational domain are periodic boundary conditions. Gravity has only one component in the negative direction of y.

For the range of Rayleigh numbers treated here, the simulations are steady state. For all simulations in this section a regular Cartesian grid consisting of 32×64 cells, with dimensionless lengths $L = 2$ and $H = 1$ was used. The polynomial degree of both velocity components and

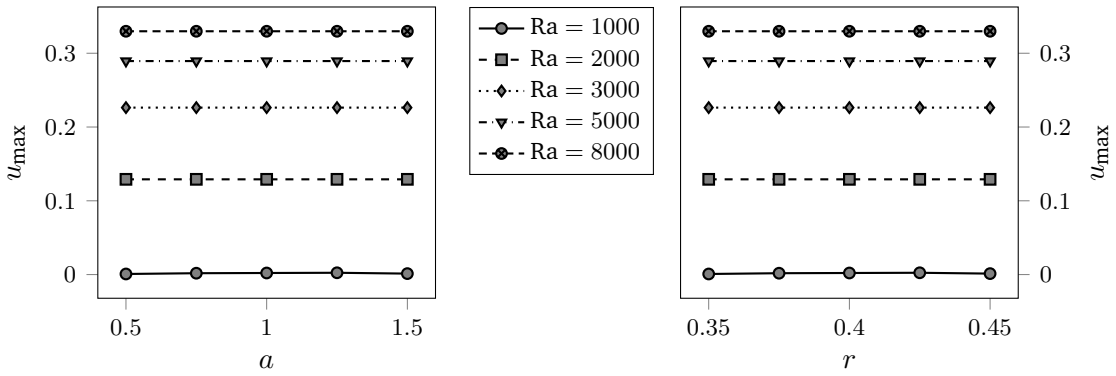


Figure 5.24: Maximum x-velocity in the Rayleigh-Bénard convection configuration for different a and r.

fig:RayBerMaxVel

temperature is four, and for pressure it is three. An open system is assumed, and $p_0 = 1.0$ throughout the simulation.

First, the stability of the XNSEC solver is studied by determining the critical value $\mathrm{Ra}_{\mathrm{crit}}$. As mentioned above, the theoretical value $\mathrm{Ra}_{\mathrm{crit}} = 1707.762$ was determined using the Boussinesq assumption in the governing equations, which, unlike the low-Mach equations, is only valid if the variation of temperature within the system is very small. For this reason, an analysis is performed for a dimensionless temperature difference $\epsilon = (T_h - T_c)/(T_h + T_c) = 0.0001$, meaning $T_h = 1.0001$ and $T_c = 0.9999$. The reference temperature is chosen to be $\hat{T}_{\mathrm{ref}} = 600\,\mathrm{K}$. Since the XNSEC solver is based on the low-Mach equations, this choice of low temperature difference allows comparing the present results with those obtained analytically using the Boussinesq approximation. The definition of the dimensionless numbers as well as the reference velocity are exactly the same as those mentioned in Section 5.2.3. The transport coefficients are calculated with Sutherland's law.

Similarly to the case discussed in Section 5.2.4, velocity fields are initialized with vortices, which play the role within the simulation of a trigger for the inherent instabilities of the problem. In particular, two vortices with opposite directions of rotation are added. The velocity components of these are again given by Equation (5.18). The coordinates of the first vortex are $(x^o, y^o) = (-0.5, 0)$ and it has a strength $a = 1$ and a radius $r = 0.4$. For the second vortex $(x^o, y^o) = (0.5, 0)$, $a = -1$ and $r = 0.4$ are selected. The initial conditions are

$$u(t=0) = 1 + u^{\text{vortex-left}} + u^{\text{vortex-right}}, \tag{5.23a}$$

$$v(t=0) = 0 + v^{\text{vortex-left}} + v^{\text{vortex-right}}, \tag{5.23b}$$

$$T(t=0) = 1, \tag{5.23c}$$

$$p(t=0) = -\frac{\rho y}{\mathrm{Fr}^2}. \tag{5.23d}$$

It is worth noting that the initialization of the solver without the vortices leads the solver to a stationary solution where there is no fluid motion. Clearly, this is a solution of the equations, but, as already mentioned, it is an unstable solution.

First, a study was performed to demonstrate the independence of the steady-state solution from the chosen initial conditions. This is demonstrated in Figure 5.24, where for a given Rayleigh number the maximum velocity obtained inside the system is shown, varying the strength a or the radius r of the vortices. It is apparent that the case $\mathrm{Ra} = 1000$ does not

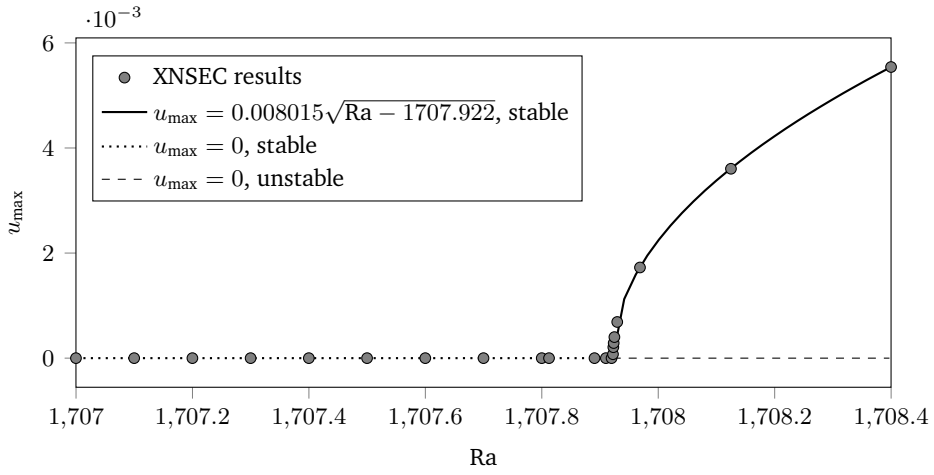


Figure 5.25: Stability behavior of the Rayleigh-Bénard convection with $\epsilon = 0.0001$. fig:ReyBerCritRa

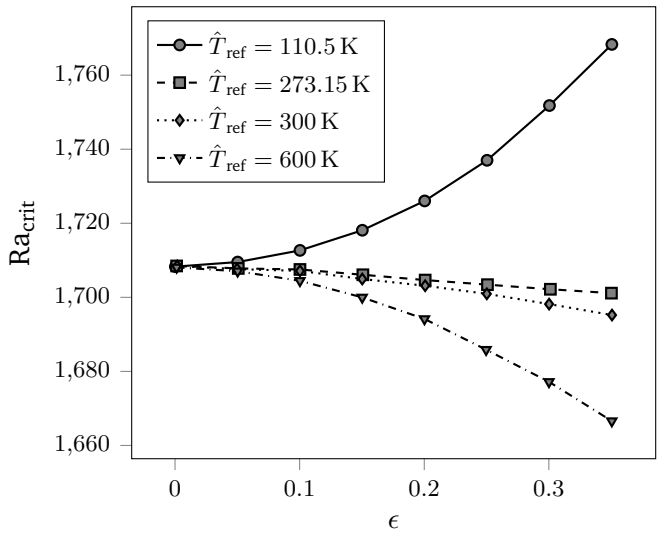


Figure 5.26: Critical Rayleigh number at different temperatures and different reference temperatures. fig:RayBenardTemperatureRaPlot

exhibit a macroscopic fluid motion, while $\mathrm{Ra} = 2000$ does. This points to the fact that the critical value is effectively in this range.

Subsequently, a series of simulations were performed with $a = 0.5$ and $r = 0.4$, and a bisection algorithm was used to find the critical Rayleigh value, obtaining $\mathrm{Ra} = 1707.922$, as shown in Figure 5.25. Compared to the theoretical value $\mathrm{Ra} = 1707.762$, it presents a difference of only 0.009%. In addition, the proportionality $u \propto \sqrt{\mathrm{Ra} - \mathrm{Ra}_{\mathrm{Ra}}}$ that is expected by analytical arguments is also obtained, in particular, $u_{\max} = 0.008015\sqrt{\mathrm{Ra} - 1707.922}$. It is possible to conclude that, at least for small temperature differences, the low-Mach approximation has a behavior similar to that of the equations with the Boussinesq approximation.

In relation to this, it is interesting to analyse the influence of the temperature difference within the system on the critical Rayleigh value. This is demonstrated in Figure 5.26, where the calculation is done for different values of the reference temperature. Clearly, for low ϵ the critical value is very close to the theoretical value obtained under the Boussinesq approximation.

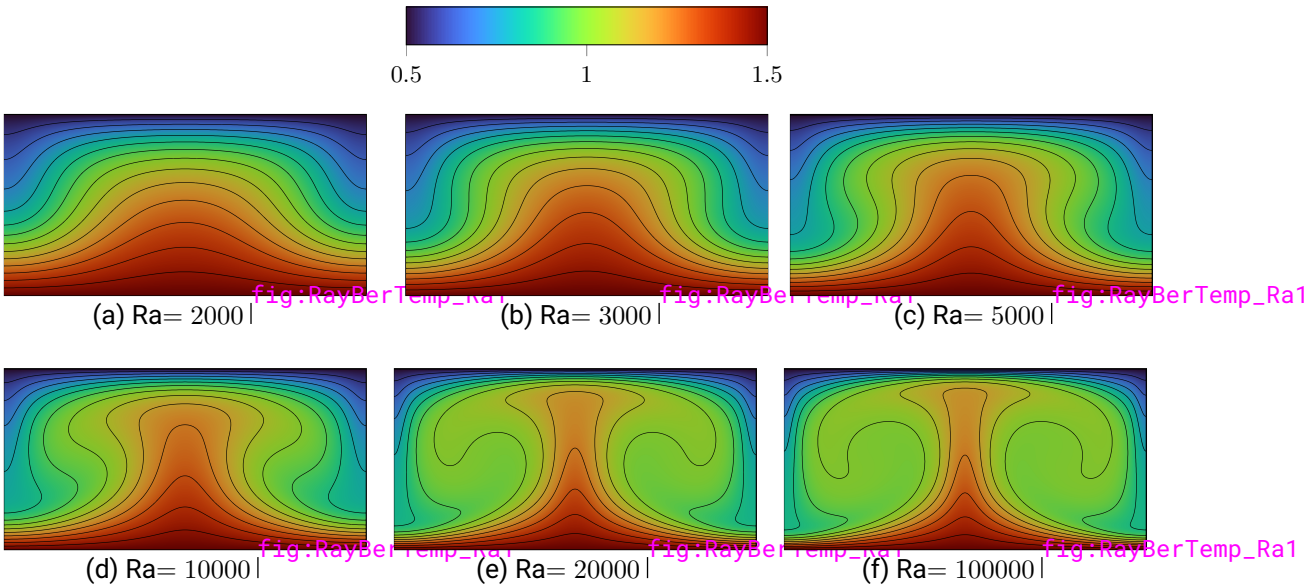


Figure 5.27: Temperature field and contours of a Rayleigh-Bénard convection roll

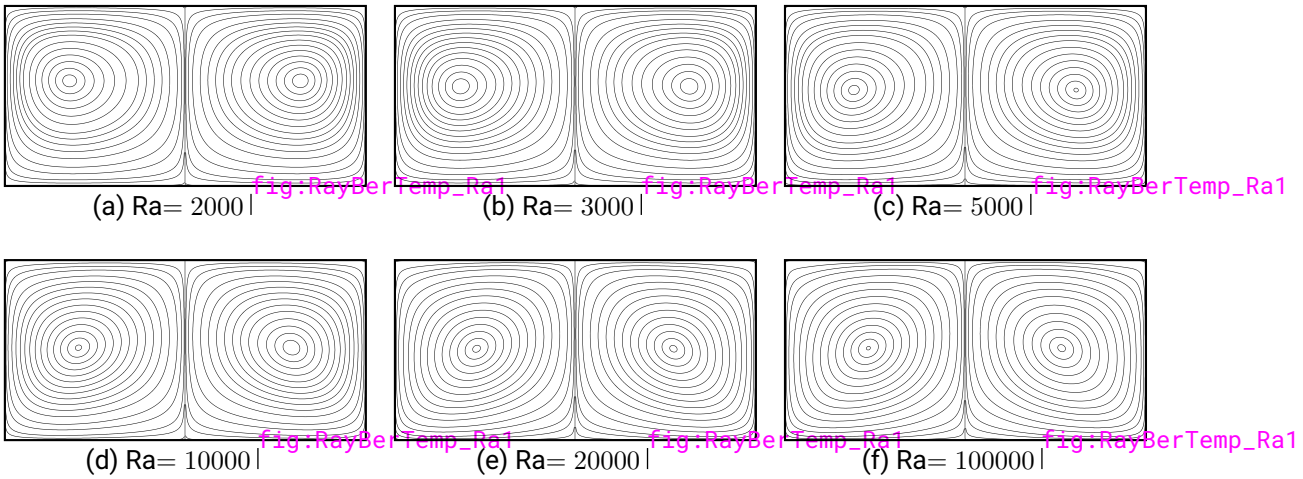


Figure 5.28: Streamlines of a Rayleigh-Bénard convection roll

As ϵ increases, a larger deviation becomes apparent. This can be attributed to the dependence of the viscosity, thermal conductivity, and density coefficients on the temperature.

Finally, in Figure 5.27 and Figure 5.28 the temperature fields and streamlines obtained with the XNSEC solver for $\epsilon = 0.5$ and $\hat{T}_{\text{ref}} = 600\,\text{K}$ for different Rayleigh numbers are shown. An interesting point to mention is the difference between the results obtained using the Boussinesq approximation and the low-Mach approximation. Comparing the results reported here with publications that use the Boussinesq approximation (Shishkina (2021) and Zhou et al. (2004)), a clear difference in the streamlines is observed. In particular, the centers of the streamlines obtained with the low-Mach solver are slightly shifted towards the colder zones, while the streamlines obtained with the Boussinesq approximation do not present this deviation and remain close to the center. A possible explanation for this is that with

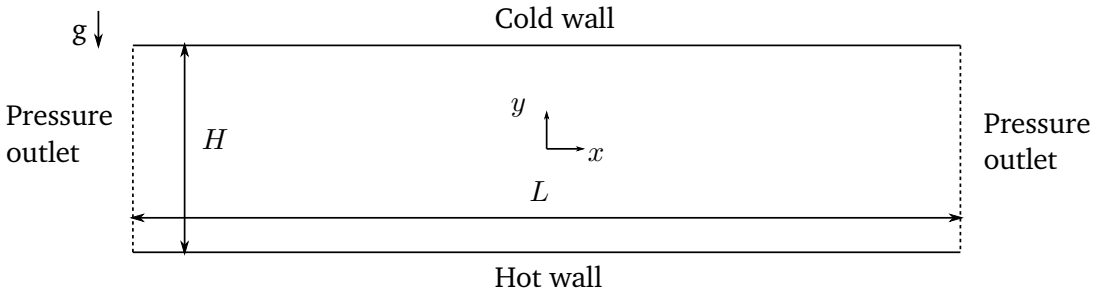


Figure 5.29: Geometry of the Rayleigh-Bénard convection with pressure outlet boundary conditions. fig:RayBenGeometry

the Boussinesq approximation the expansion effects of the fluid are ignored, whereas in the low-Mach equations this is not the case. When the latter is used, the hot fluid moves the center of the streamline towards the cold fluid.

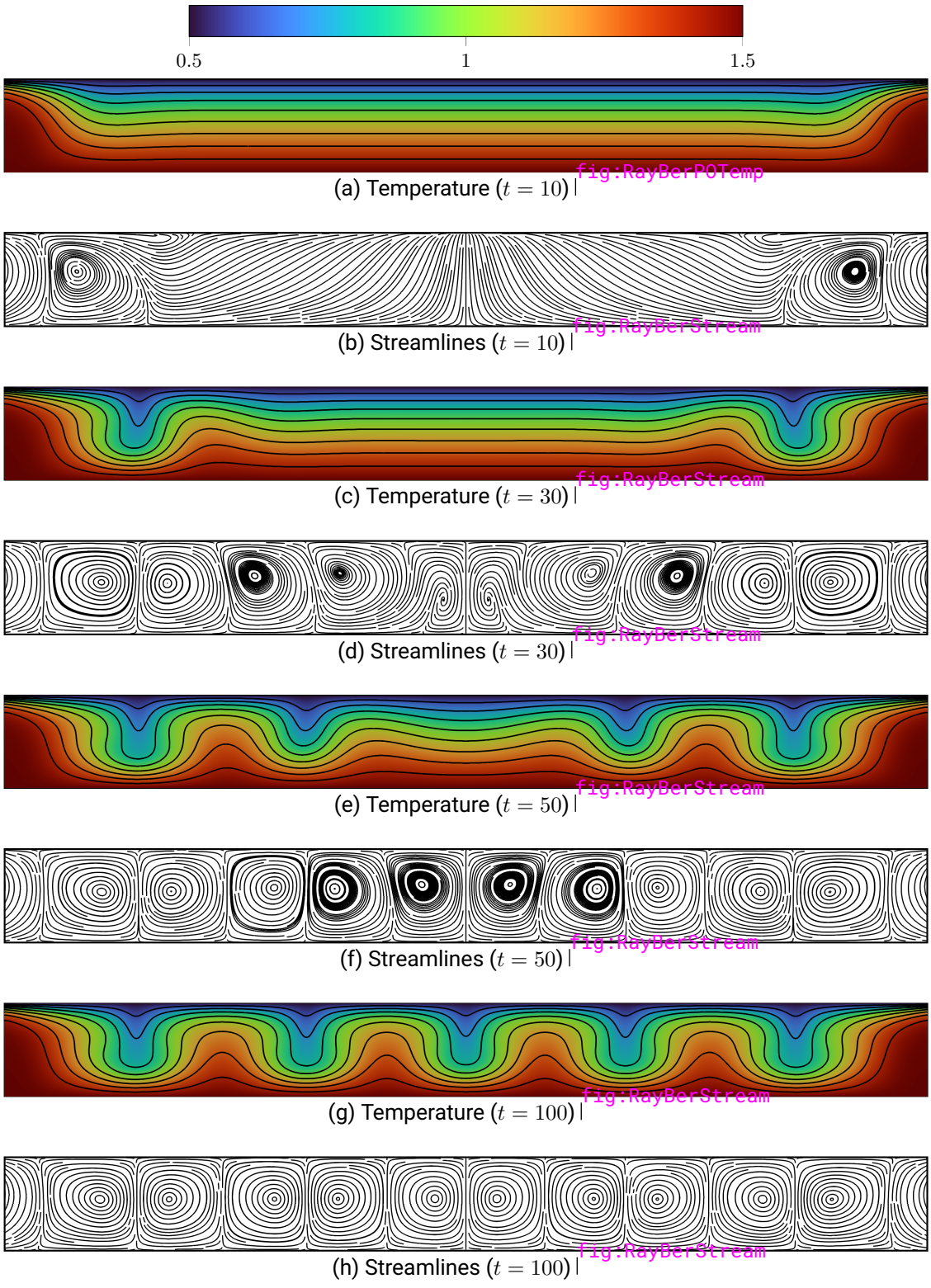
Pressure outlet boundary condition

ssec:MultipleCellConv

Now a transient calculation of the Rayleigh-Bénard configuration is done. The configuration is similar to the one described for the case with periodic boundary conditions, with the difference that the boundaries of the left and right sides are now pressure outlet boundary conditions and the length L of the system is chosen to be considerably longer. A sketch of the system is shown in Figure 5.29. Theoretically, adequately representing the convection rolls would require an infinitely long system, because then, the influence of the boundary conditions would be negligible The nondimensional lengths are chosen to be $H = 1$ and $L = 10$. A grid with 32×320 cells is used. The polynomial degrees for the velocity components and temperature are set to four and for the hydrodynamic pressure to three. The time discretization is done again with a BDF-3 scheme and the calculation time is 150, using timesteps of $\Delta t = 0.5$. The temperatures are set to $T_h = 1.5$ and $T_c = 0.5$, with a reference temperature of $\hat{T}_{\text{ref}} = 600\,\text{K}$. The Rayleigh number is $\text{Ra} = 5659$, which is above the critical value. The initial conditions are chosen to be

$$u(t=0) = 0, \tag{5.24a}$$
$$v(t=0) = 0, \tag{5.24b}$$
$$T(t=0) = 1, \tag{5.24c}$$
$$p(t=0) = -\frac{\rho y}{\text{Fr}^2}. \tag{5.24d}$$

Note that no vortex is included in the initial conditions and that only a fluid at rest is considered. A perturbation-like effect caused by the pressure outlet boundaries triggers the movement of the fluid. At the first stages of the simulation the perturbations induce a vortex-like structure close to the left and right boundaries, and start gradually filling the whole domain, finally reaching a steady solution. This can be seen in Figure 5.30. Clearly, in the center of the domain (far away from the outlet boundary conditions), the structure of the solution is very similar to the ones shown in the last subsection. In fact, if the domain length is chosen sufficiently large, they should be equal.



(a) Temperature ($t = 10$) | fig:RayBerPOTemp

(b) Streamlines ($t = 10$) | fig:RayBerStream

(c) Temperature ($t = 30$) | fig:RayBerStream

(d) Streamlines ($t = 30$) | fig:RayBerStream

(e) Temperature ($t = 50$) | fig:RayBerStream

(f) Streamlines ($t = 50$) | fig:RayBerStream

(g) Temperature ($t = 100$) | fig:RayBerStream

(h) Streamlines ($t = 100$) | fig:RayBerStream

Figure 5.30: Temperature and streamlines of the Rayleigh-Bénard flow with pressure outlets. fig:RayBerUnsteadySol

5.3 Multi-component non-isothermal cases

sec:MultCompNonIsothermCase

Finally in this section, test cases where the whole operator presented in Chapter 3 is used are shown. The balance equations of continuity, momentum, energy and species are considered and solved in a coupled manner, together with a equation of state and expressions for the transport parameters. For all the cases treated here, the solution of the flame sheet problem described in Section 2.2 is calculated first, and then this solution is used as an initial estimate for the solution of the finite chemistry rate problem (see Section 2.1.2). For all test cases presented in this section, a smoothing parameter $\sigma = 40$ is used (see Section 4.4).

As already mentioned before, for the resolution of the linear solver using a direct solver such as *PARDISO* was not possible, because the memory requirements are too high. In all calculations of this section the multigrid orthonormalization method commented in Section 4.2.3 is used.

The combustion model used is the one-step model shown in Section 2.1. All calculations assume methane as fuel, therefore, the relevant chemical components are CH_4, O_2, CO_2, H_2O and N_2, thus $N = 5$ and $\mathbf{Y}' = \left(Y_{CH_4}, Y_{O_2}, Y_{CO_2}, Y_{H_2O}\right)$. The nitrogen mass fraction Y_{N_2} is calculated according to Equation (2.23).

First in Section 5.3.1 a coflow laminar diffusion flame configuration is calculated and some remarks about the convergence behavior of the XNSEC solver using are made. Next in Section 5.3.2 a diffusion flame in a two-dimensional counterflow configuration is simulated and compared with results from a one-dimensional configuration solved using a fourth-order finite difference solver. Finally in Section 5.3.3 the convergence rates of the fully coupled solver are investigated by studing a pseudo-one-dimensional diffusion flame configuration.

5.3.1 Coflow laminar diffusion flame

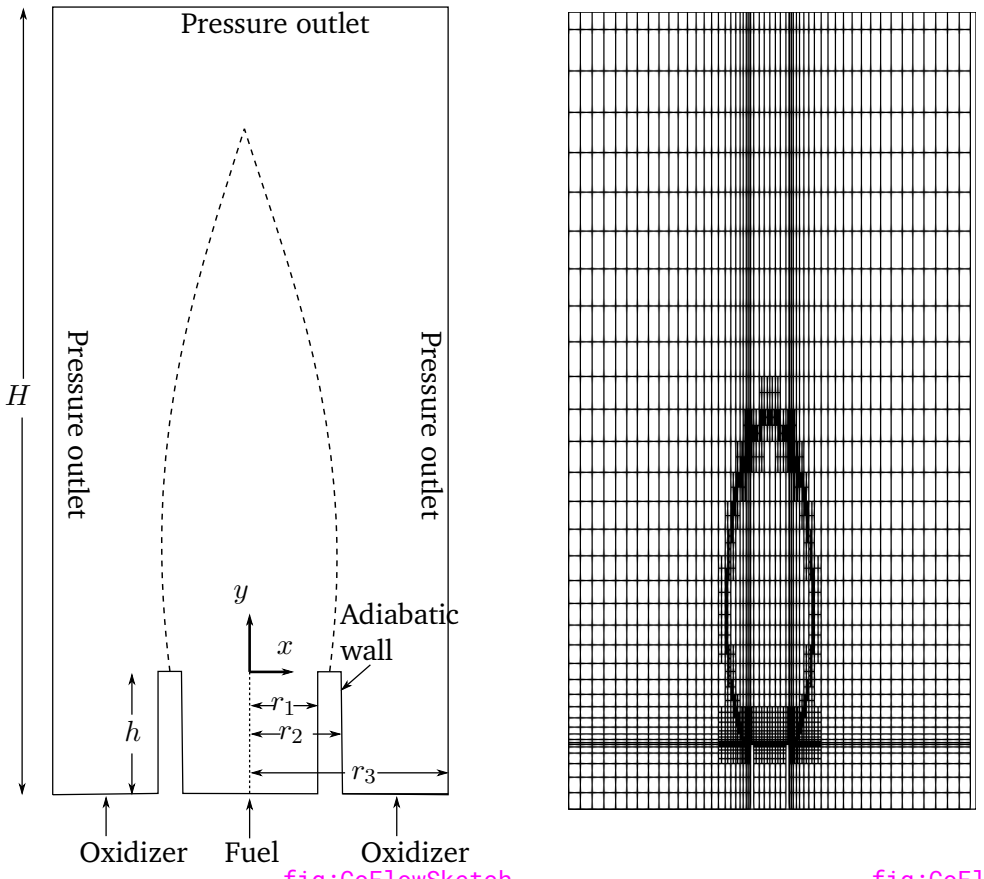
ssec:coflowFlame

The coflowing flame configuration is used as a first test to assess the behavior of the solver for reactive flows applications. It consists of two concentric ducts that emit fuel and oxidant into the system, which after ignition forms a flame. This configuration has been widely studied, starting with the seminal work of Burke and Schumann (1928) and followed by many others (see, for example, Smooke and Giovangigli (1992), Smooke et al. (1986), and Braack et al. (1997)). In particular, the work from Smooke and Giovangigli (1992) and later publications solved this configuration using a 2D-axisymmetric system and also used the flame sheet estimates to find adequate initial conditions for their Newton algorithm.

Since the solution of the axisymmetric system of equations presents numerical difficulties that are not the main concern of the present work, in this section an infinitely long slot burner configuration is considered. For that kind of configuration, cartesian coordinates describe the problem naturally.

Set-up

A schematic diagram of the configuration can be seen in Figure 5.31a. The system consists of a fuel inlet with two oxygen inlets on its sides. These inlets are separated by a finite wall thickness. The inclusion of this separation is observed to be necessary to be able to obtain a converged solution, which makes sense, since a finite separation between both inlets smooths the big gradients due to strong mixing and reaction in that area. Although the system is clearly



(a) Sketch (b) Refined mesh

Figure 5.31: Geometry of a coflowing flame configuration (not to scale).

symmetric around the axis $x = 0$, no symmetry assumption is made and the whole domain is considered for the simulation.

The lengths depicted in Figure 5.31a are set as $r_1 = 1$, $r_2 = 1.2$, and $r_3 = 11.763$. Aditionally $h = 4$ and $H = 63$. The lengths r_3 and H are set with arbitrarily large values in order to avoid influence of the outer boundary conditions on the solution of the flame zone. Setting a higher value of r_3 or L did not significantly affect the results. The inlet boundary conditions are set as:

- Oxidizer inlet: $\{\forall(x, y) : y = -h \wedge x \in [-r_3, -r_2] \cup [r_2, r_3]\}$

$$u = 0, \qquad v = v_O, \qquad T = T^O, \qquad \mathbf{Y}' = (0, Y^O_{\mathrm{O_2}}, 0, 0)$$

- Fuel Inlet: $\{\forall(x, y) : y = -h \wedge x \in [-r_1, r_1]\}$

$$u = 0, \qquad v = v^F(x), \qquad T = T^F, \qquad \mathbf{Y}' = (Y^F_{\mathrm{CH_4}}, 0, 0, 0)$$

The oxidizer enters the system as a plug flow with a constant velocity of $v_O = 1$. The inlet

velocity of the fuel stream v_F is a a parabolic profile given by

$$v^F(x) = \left[1 - \left(\frac{x}{X_1}\right)^2\right] v_m^F \tag{5.25}$$

with $v_m^F = 0.592$. The inlet temperatures of both streams is $T^F = T^O = 1$. Combustion of diluted methane on air is considered, with $Y^F_{\mathrm{CH_4}} = 0.2$ and $Y^F_{\mathrm{N_2}} = 0.8$ for the fuel stream, and $Y^O_{\mathrm{O_2}} = 0.23$ and $Y^O_{\mathrm{N_2}} = 0.77$ for the oxidyzer stream. The superscripts F and O represent the fuel and oxidizer inlet respectively. The pressure outlet boundary condition is the same as Equation (2.37c). Finally, the boundary conditions at the tips correspond to adiabatic walls, which are defined as in Equation (2.37b), with $\mathbf{u}_{\mathrm{D}} = (0, 0)$.

The variables are nondimensionalized in the usual way. The reference length is set $\hat{L}_{\mathrm{ref}} = 0.635\,\mathrm{cm}$ and the reference velocity $\hat{u}_{\mathrm{ref}} = 8.19\,\mathrm{cm\,s^{-1}}$. The reference temperature is $\hat{T}_{\mathrm{ref}} = 300\,\mathrm{K}$. All derived variables are nondimensionalized using the air stream as a reference condition, i.e. $\hat{\rho}_{\mathrm{ref}} = 1.17\,\mathrm{kg\,m^{-3}}$, $\hat{\mu}_{\mathrm{ref}} = 1.85 \times 10^{-5}\,\mathrm{kg\,m^{-1}\,s^{-1}}$ and $\hat{W}_{\mathrm{ref}} = 28.82\,\mathrm{kg\,kmol^{-1}}$, giving the nondimensional numbers $\mathrm{Re} = 33.02$ and $\mathrm{Da} = 2.17 \cdot 10^9$. The Prandtl number is assumed to be constant with $\mathrm{Pr} = 0.71$. The reference heat capacity is set $\hat{c}_{p,\mathrm{ref}} = 1.3\,\mathrm{kJ\,kg^{-1}\,K^{-1}}$, which is also the constant value used for the flame sheet calculation (i.e. $c_p = 1$). The choice of this value for the heat capacity is important because it gives a solution of the flame sheet which is similar to the actual solution of the full problem. Gravity effects are not taken into account. The transport parameters are calculated using Sutherland's law with $\hat{S} = 110.5\,\mathrm{K}$. The mixture heat capacity c_p is calculated with Equation (2.28) and using NASA polynomials for the heat capacity of each component. Finally, a nonunity but constant Lewis number formulation is used, with $\mathrm{Le}_{\mathrm{CH_4}} = 0.97$, $\mathrm{Le}_{\mathrm{O_2}} = 1.11$, $\mathrm{Le}_{\mathrm{H_2O}} = 0.83$ and $\mathrm{Le}_{\mathrm{CO_2}} = 1.39$ (Smooke and Giovangigli, 1991)

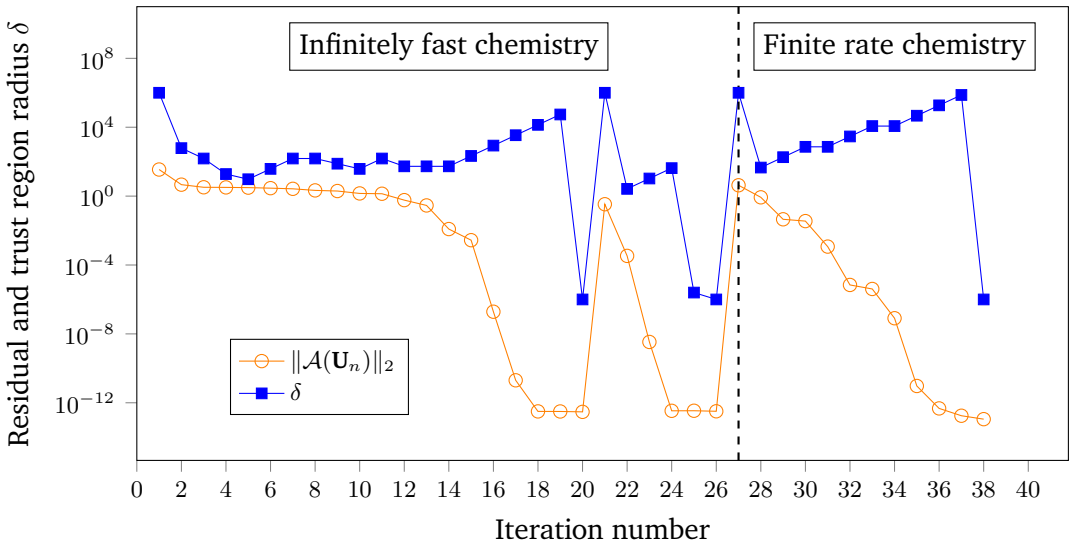
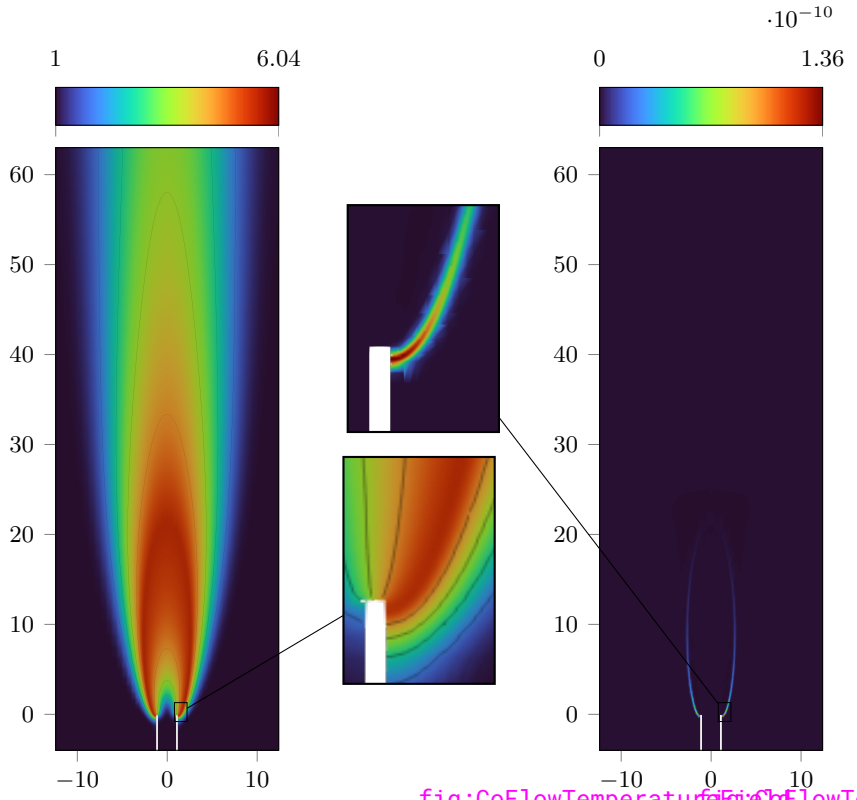


Figure 5.32: Typical convergence history of a diffusion flame in the coflowing flame configuration.A mesh refinement is done at iteration 21. fig:CoFlow_ConvergenceStory



(a) Dimensionless temperature (b) Reaction rate

fig:CoFlowTemperatureFig:CoFlowTemperatureField

Figure 5.33: Temperature and reaction rate fields of the coflow configuration.

Fig:CoFlowFlameFig

Numerical results

The purpose of simulating this case is to test the XNSEC solver in a real-world application, with realistic physical parameter values. In particular, it is intended to demonstrate that the strategy of using the flame sheet solution as the initial estimate is adequate for obtaining a converged solution, even for a case where the physical properties of the system (ρ, μ, c_p) are composition and temperature dependent, and for nonunity Lewis numbers. Numerical experiments using the XNSEC solver showed that the solution of the problem is highly mesh-dependent. The presence of very high gradients in some areas requires a higher density of cells to obtain a well-resolved solution. In Figure 5.31b the actual mesh used for the solution of the full problem is shown. A base mesh with smaller elements in the vicinity of the inlets and larger elements further away from them is used. It is observed that the complex mixing and combustion phenomena that occur in the vicinity of the inlets have a critical effect on the convergence of the solution. For this reason, a special refinement of the base mesh is done in the vicinity of the tips. Furthermore, an adequate mesh resolution at the flame location is critical. In order to avoid over-solving, an adaptive mesh refinement strategy in a pseudo-time-stepping framework is used. After obtaining a steady-state solution, the mesh is refined and the calculation is started again (see Section 4.4). In particular, for this case, three pseudo-timesteps were performed for the flame sheet calculation. After each pseudo-timestep, the mesh is refined in the vicinity of the flame sheet, that is, in the cells where $z = z_{\text{st}}$. Obviously, a finer grid in

the vicinity of the surface $z = z_{\text{st}}$ will be beneficial for the solution of the finite reaction rate problem if the same conditions used to derive the flame sheet equations (namely constant c_p and unity Lewis number) are assumed. However, experiments with the XNSEC solver have shown that this refinement strategy is still beneficial for the convergence of the complete problem even with non-constant c_p and nonunity Lewis numbers.

In Figure 5.32 the convergence history using the Newton algorithm presented in Section 4.2.2 is shown. The flame sheet calculation requires 20 Newton iterations to find a solution. It is clearly seen that the residuals $\|\mathcal{A}(\mathbf{U}_n)\|_2$ decrease very slowly for about the first 14 iterations, while the parameter δ of the globalised Newton method is adapted to find an optimal value to reduce the residuals. Around iteration 14 the algorithm starts to increase δ, leading to a faster reduction of the residuals. A solution to the problem according to the termination criterion exposed in Section 4.2.5 is found in iteration number 20. In iteration 21 mesh refinement based on the strategy mentioned above is used and now only 6 iterations are required to find a converged solution. Finally, in iteration number 27 the flame sheet solution is used as the initial estimate for the full problem, which requires only 11 iterations to find a solution.

For the flame sheet calculation a polynomial degree $k = 2$ is chosen, resulting in a rather small system with 49,140 degrees of freedom. For the finite rate calculation $k = 4$ is used, which resulted in a system with 482,310 degrees of freedom. This highlights another advantage of the approach of using the flame sheet calculation for two-dimensional simulations can be highlighted. The initial estimate can be found relatively easily for a system with few degrees of freedom. Using the solution found as the initial estimate has the consequence that Newton's algorithm for the complete problem (which has many more degrees of freedom) only needs a few iterations to find a solution.

In Figure 5.33 the obtained temperature and reaction rate fields are shown. Since for the selected inlet velocities the flame corresponds to an overventilated one, the typical jet form is observed. The maximum dimensionless temperature T reached corresponds to 6.04 (meaning $1812\,\text{K}$). Magnified plots show that the bottom part of the flame sits on the outside part of the tips. The high reaction rates appearing in the area close to the inlets are also worth highlighting. This could explain why the mesh refinement in the vicinity of the inlets is crucial to obtain a converged solution.

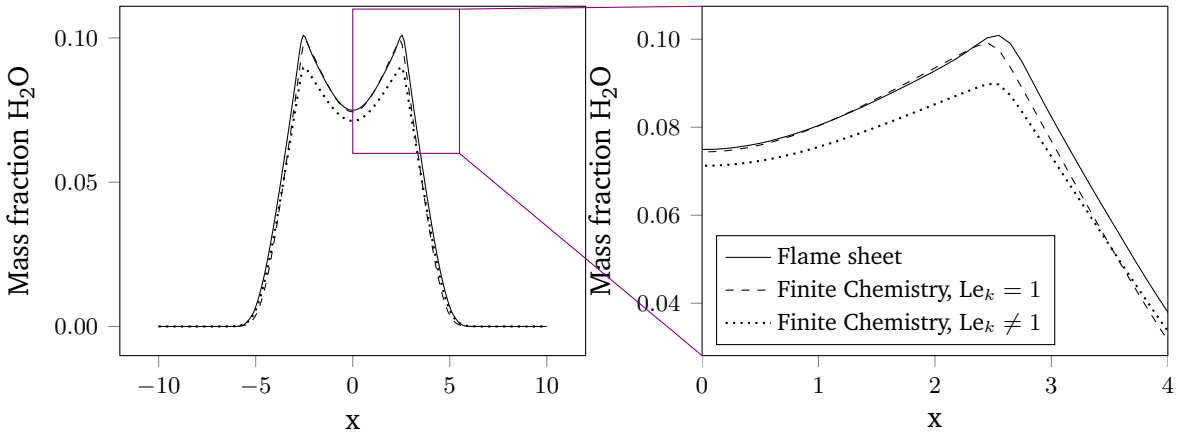


Figure 5.34: Mass fraction field of H_2O over the line $x = 10$. fig:CoFlowMF3_infiniteFinite

A question to consider is whether indeed the initialization of the finite reaction rate problem with the flame sheet provides adequate estimates for the Newton algorithm, in particular for more complex problems that do not fulfil the assumptions made to obtain the equations of the mixture fraction problem such as systems with nonconstant heat capacity c_p or with nonunity Lewis numbers. For illustrative purposes, the simulation of the coflowing flame for unity and nonunity lewis numbers was performed. In Figure 5.34 the solution obtained for the mass fraction field of H_2O along the line $y = 10$ is shown. As expected, the solution obtained for the case with unity Lewis number is very close to the one obtained using the flame sheet. Furthermore. the case with nonunity Lewis number also presents a solution very similar to that of the flame sheet, with a small deviation in the vicinity of the zones where the chemical reaction occurs. Both finite chemistry calculations use the same initial estimate and both calculations find the converged solution after 11 Newton iterations. In all the simulations shown in this thesis, the use of the flame sheet estimation served as a way to initialise the finite-reaction rate problem.

The simulation of the coflowing flame shows that the strategy of using the flame sheet as an initial condition offers an efficient way to obtain steady-state solutions of combustion problems. Since the flame sheet is only used an estimate for the ignited solution, it is possible to perform the calculations on relatively coarse grids, and use low-order polynomial degrees. The obtained solution can be used as an estimate for calculations with higher polynomial degrees to find a more accurate solution of the full flame problem in a few iterations. A disadvantage, as already mentioned in Section 4.4, is the requirement to choose the c_p parameter, since an inappropriate choice gives a solution of the flame sheet problem far away from the solution of the full problem. However, the user's experience and access to experimental information allows estimating this value relatively easily. Note that the value chosen in this section ($\hat{c}_{p,\text{ref}} = 1.3\,\text{kJ}\,\text{kg}^{-1}\,\text{K}^{-1}$) served as an adequate estimate for all the simulations presented in this thesis.

5.3.2 Counterflow diffusion flame *

ss:CDF

The counterflow diffusion flame is a canonical configuration used to study the structure of nonpremixed flames. This simple configuration has been a subject of study for decades because it provides a simple way of creating a strained diffusion flame, which proves to be useful when studying the flame structure, extinction limits or production of pollutants of flames (Pandya and Weinberg, 1964; Spalding, 1961; Keyes and Smooke, 1987; Lee et al., 2000).

The counterflow diffusion flame consists of two oppositely situated jets. The fuel (possibly mixed with some inert component, such as nitrogen) is fed into the system by one of the jets, while the other jet feeds oxidyzer to the system, thereby establishing a stagnation point flow. On contact and after ignition, the reactants produce a flame that is located in the vicinity of the stagnation plane. A diagram of the setup can be seen in Figure 5.35. In this section, the solution of a steady two-dimensional flame formed in an infinitely long slot burner will be treated. Simirlarly to the coflow configuration treated before, the infintely long slot burner configuration can be calculated naturally using cartesian coordinates.

First, as a means of verifying the solver for combustion applications, the results obtained with the XNSEC solver for steady two-dimensional counterflow diffusion flame are compared with the solution of a simplified system of equations for a steady and quasi one-dimensional

*Modified version from Gutiérrez-Jorquera and Kummer (2022)

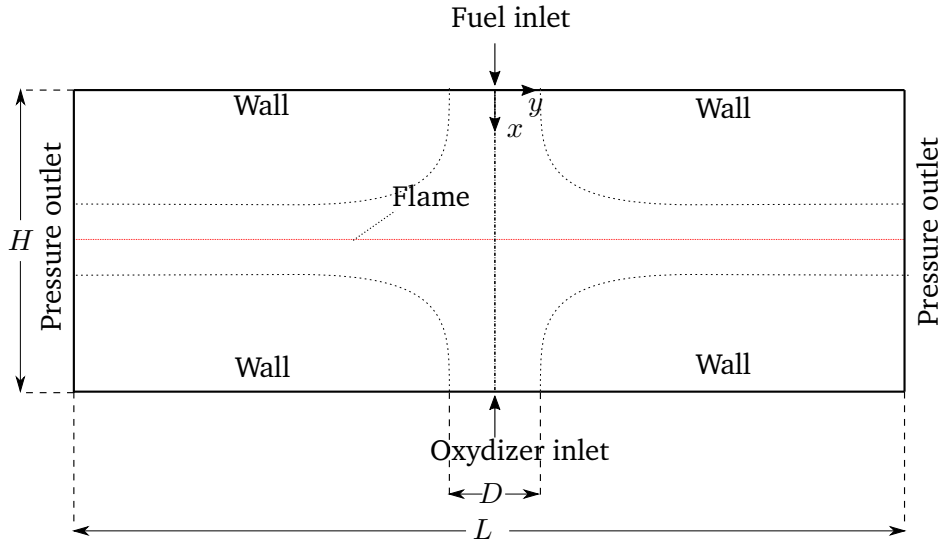


Figure 5.35: Schematic representation (not to scale) of the counterflow diffusion flame configuration. fig:CDFScheme

flame. Later, the influence of the inlet velocities on the maxmimum temperature is studied and finally some remarks concerning the convergence behavior are given.

The one-dimensional diffusion flame

By assuming an infinite injector diameter, a self-similar solution and by neglecting the radial gradients of the scalar variables along the axis of symmetry, it is possible to reduce the three-dimensional governing equations to a one-dimensional formulation along the stagnation streamline $y = 0$ (see the textbook from Kee et al. (2003) for the derivation). The governing equations for a steady planar stagnation flow reduce to

$$\frac{\partial \hat{\rho}\hat{v}}{\partial \hat{x}} + \hat{\rho}\hat{U} = 0, \tag{5.26a}$$ {eq:OneDimCont}

$$\hat{\rho}\hat{v}\frac{\partial \hat{U}}{\partial \hat{x}} + \hat{\rho}\hat{U}^2 = -\hat{\Lambda} + \frac{\partial}{\partial \hat{x}}\left(\hat{\mu}\frac{\partial \hat{U}}{\partial \hat{x}}\right), \tag{5.26b}$$ {eq:OneDimMom}

$$\hat{\rho}\hat{c}_p\hat{v}\frac{\partial \hat{T}}{\partial \hat{x}} = \frac{\partial}{\partial \hat{x}}\left(\hat{\lambda}\frac{\partial \hat{T}}{\partial \hat{x}}\right) + \hat{\mathrm{Q}}\ \hat{\mathcal{Q}}, \tag{5.26c}$$ {eq:OneDimTemp}

$$\hat{\rho}\hat{v}\frac{\partial Y_k}{\partial \hat{x}} = \frac{\partial}{\partial \hat{x}}\left(\hat{\rho}\hat{D}\frac{\partial Y_k}{\partial \hat{x}}\right) + \hat{W}_k\nu_k\hat{\mathcal{Q}}, \quad (k = 1, \ldots, N-1) \tag{5.26d}$$ {eq:OneDimMF}

eqs:OneDimEquations

where U is the scaled velocity and $\hat{\Lambda}$ is the radial pressure curvature, which is an eigenvalue independent of $\hat{x}$. Again, the hat sign represents dimensional variables. The equations are written assuming Fick's law and an one-step combustion model. The system of equations needs to be solved for $\hat{v}$, $\hat{U}$, $\hat{T}$ and for Y_k with $(k = 1, \ldots, N-1)$. In addition, an equation of state and expressions for the heat capacity $\hat{c}_p$ and the transport parameters $\hat{\mu}, \hat{\lambda}, (\hat{\rho}\hat{D})$ are needed. This formulation is very well known and is often used for analysis of flame structure and determination of extinction points, just to mention a few.

In order to assess the ability of the XNSEC solver to simulate such a system, the solution obtained for a two-dimensional configuration is compared with the solution of the quasi one-dimensional equations solved with BVP4, a fourth order finite difference boundary value problem solver provided by MATLAB (Kierzenka and Shampine, 2001). The BVP4 solver provides automatic meshing and error control based on the residuals of the solution, allowing the development with relatively low effort a code that solves the one-dimensional equations.

It is important to mention some points regarding the solution of these equations using the BVP4 solver. Analogous to the problem mentioned in Section 4.4, the solution of the system of Equations (5.26a) to (5.26d) has multiple solutions. One of them is the cold solution and the other is the burning one (see Figure 2.1). The same idea mentioned in Section 4.4 is also valid for the quasi-one-dimensional configuration. In particular, this means that a first step for finding a converged solution of Equations (5.26a) to (5.26d) is to solve the system

$$\frac{\partial \hat{\rho}\hat{v}}{\partial \hat{x}} + \hat{\rho}\hat{U} = 0 \tag{5.27a}$$

$$\hat{\rho}\hat{v}\frac{\partial \hat{U}}{\partial \hat{x}} + \hat{\rho}\hat{U}^2 = -\hat{\Lambda} + \frac{\partial}{\partial \hat{x}}\left(\hat{\mu}\frac{\partial \hat{U}}{\partial \hat{x}}\right) \tag{5.27b}$$

{eq:OneDimCont2}

$$\hat{\rho}\hat{v}\frac{\partial Z}{\partial \hat{x}} = \frac{\partial}{\partial \hat{x}}\left(\hat{\rho}\hat{D}\frac{\partial Z}{\partial \hat{x}}\right) \tag{5.27c}$$

{eq:OneDimMom2}

eqs:OneDimEquationsMixtureFraction

together with the equation of state (2.26) and expressions for the transport parameters. The dependency of the temperature and mass fractions on the mixture fraction Z is given by the Burke-Schumann limit (see Section 2.2). This solution can be used as an initial estimate for the solution of Equations (5.26a) to (5.26d). It is observed that if the constant value of $\hat{c}_p$ for calculation of the flame sheet is chosen too big, the solver yields the solutions without a flame. For the calculations considered here $\hat{c}_p = 1.3\,\mathrm{kJ\,kg^{-1}\,K}$ is an adequate value that delivers the ignited solution.

However, it is observed that this flame sheet solution is not directly useful as an initial estimation for the solution of the full system of equations. In order to help the BVP4 solver find a converged solution, an intermediate step is necessary. First, the flame sheet solution is used as an initial estimate for the solution of Equation (5.26) using a constant heat capacity and unity Lewis number. Once the algorithm has found a solution, it can be used for solving the same system but with a variable heat capacity according to Equation (2.28) and user defined Lewis numbers.

Set-up of the two-dimensional counterflow diffusion flame

In this part, the simulation of combustion of diluted methane with air in a two-dimensional infinitely long slot burner configuration using the XNSEC solver is studied. The solution is obtained by solving Equations (2.22a) to (2.22d), making use of the flame sheet solution as initial estimates. The transport parameters are calculated using Sutherland law with $\hat{S} = 110.5\,\mathrm{K}$. Gravity effects are not taken into account. The mixture heat capacity c_p is calculated with Equation (2.28) and using NASA polynomials for the heat capacity of each component.

For comparison with the quasi-one-dimensional model, three pairs of inlet velocities are considered. They are shown in Table 5.4. Both streams enter at a temperature $\hat{T}^O = \hat{T}^F =$

$300\,\mathrm{K}$. The mass composition of the fuel inlet is assumed to be $Y^F_{\mathrm{CH_4}} = 0.2$ and $Y^F_{\mathrm{N_2}} = 0.8$, and the oxidizer inlet is air with $Y^O_{\mathrm{O_2}} = 0.23$ and $Y^O_{\mathrm{N_2}} = 0.77$.

Counterflow diffusion flames are usually characterized by the strain rate a. Many different definitions for it can be found in the literature (Fiala and Sattelmayer, 2014). In this work the definition of the strain rate based on the maximum axial velocity gradient is used. The strain ratesfor the three cases mentioned above are $36.04\,\mathrm{s}^{-1}$, $84.03\,\mathrm{s}^{-1}$ and $176.71\,\mathrm{s}^{-1}$, respectively.

The lengths described in Figure 5.35 are $\hat{D} = 2\,\mathrm{cm}$, $\hat{H} = 2\,\mathrm{cm}$ and $\hat{L} = 12\,\mathrm{cm}$. The variables are nondimensionalized using $\hat{L}_{\mathrm{ref}} = 2\,\mathrm{cm}$, $\hat{T}_{\mathrm{ref}} = 300\,\mathrm{K}$ and $\hat{p}_{\mathrm{ref}} = 101\,325\,\mathrm{Pa}$. For each case, the reference velocity is set to $\hat{u}_{\mathrm{ref}} = \hat{v}^O$. Again, all derived variables are nondimensionalized using the air stream as a reference condition, i.e. $\hat{\rho}_{\mathrm{ref}} = 1.17\,\mathrm{kg\,m^{-3}}$, $\hat{\mu}_{\mathrm{ref}} = 1.85 \times 10^{-5}\,\mathrm{kg\,m^{-1}\,s^{-1}}$ and $\hat{W}_{\mathrm{ref}} = 28.82\,\mathrm{kg\,kmol^{-1}}$. The reference heat capacity is set $\hat{c}_{p,\mathrm{ref}} = 1.3\,\mathrm{kJ\,kg^{-1}\,K^{-1}}$.

Under this conditions, the Reynolds numbers are $\mathrm{Re} = 156$, $\mathrm{Re} = 390$ and $\mathrm{Re} = 858$, for the low, medium and high inlet velocities respectively. The Prandtl number is assumed to be constant with $\mathrm{Pr} = 0.75$. A nonunity but constant Lewis number formulation is used, with $\mathrm{Le}_{\mathrm{CH_4}} = 0.97$, $\mathrm{Le}_{\mathrm{O_2}} = 1.11$, $\mathrm{Le}_{\mathrm{H_2O}} = 0.83$ and $\mathrm{Le}_{\mathrm{CO_2}} = 1.39$ (Smooke and Giovangigli, 1991). The system is considered open, the thermodynamic pressure is constant and set to $p_0 = 1$.

The boundary condition of the inlets are

- Oxidizer inlet: $\{\forall(x, y) : y = 0 \wedge x \in [-D/2, D/2]\}$

$$u = 0, \qquad v = v^O(y), \qquad T = 1.0, \qquad \mathbf{Y}' = (0, Y^O_{\mathrm{O_2}}, 0, 0).$$

- Fuel Inlet: $\{\forall(x, y) : y = H \wedge x \in [-D/2, D/2]\}$

$$u = 0, \qquad v = v^F(y), \qquad T = 1.0, \qquad \mathbf{Y}' = (Y^F_{\mathrm{CH_4}}, 0, 0, 0).$$

The pressure outlet boundary condition is the same as Equation (2.37c). The pressure outlet boundaries are placed far away from the center of the domain, to decrease the effect on the centerline. Placing the boundary further away did not appreciably change the results. Finally the boundary conditions at the walls are defined as in Equation (2.37b), with $\mathbf{u}_{\mathrm{D}} = (0, 0)$ and a constant temperature $T = 1.0$.

In Figure 5.36 the solution profiles for case (a) are shown. The used mesh was obtained by a process of mesh refinement. The base mesh is initially created with a larger concentration of elements in the center of the domain. The points of intersection from the velocity inlet and wall boundary conditions are also refined, which was observed to improve the robustness of the algorithm. Similarly to the coflowing flame (Section 5.3.1), during the solution algorithm of the flame sheet problem, the mesh is additionally refined around the flame sheet making use of a pseudo-time-stepping approach. As expected, a stagnation flow develops and a flame

	$\hat{v}^F_m\,(\mathrm{cm\,s^{-1}})$	$\hat{v}^O_m\,(\mathrm{cm\,s^{-1}})$	$a(\mathrm{s^{-1}})$	$\hat{T}^F(\mathrm{K})$	$\hat{T}^O(\mathrm{K})$
case(a)	4.85	12.29	36.04	300	300
case(b)	12.13	30.73	84.03	300	300
case(c)	26.69	67.62	176.71	300	300

Table 5.4: Maximum inlet velocity, strain and temperatures used for the counterflow diffusion flame calculations.

tab:cdf_velocities

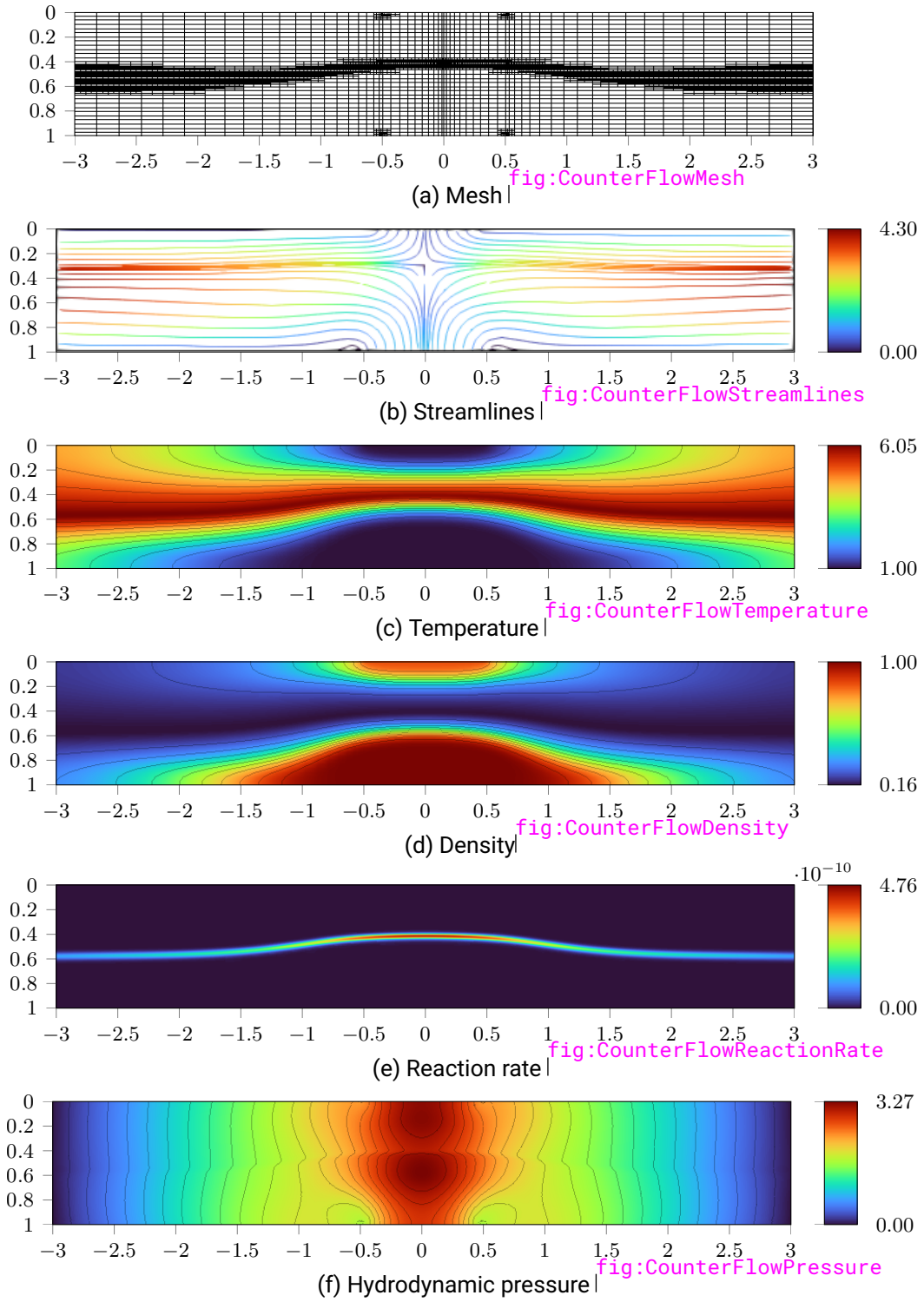


Figure 5.36: Nondimensional solution and derived fields of the counterflow flame configuration for case (a).

fig:CounterFlowFlameFig1

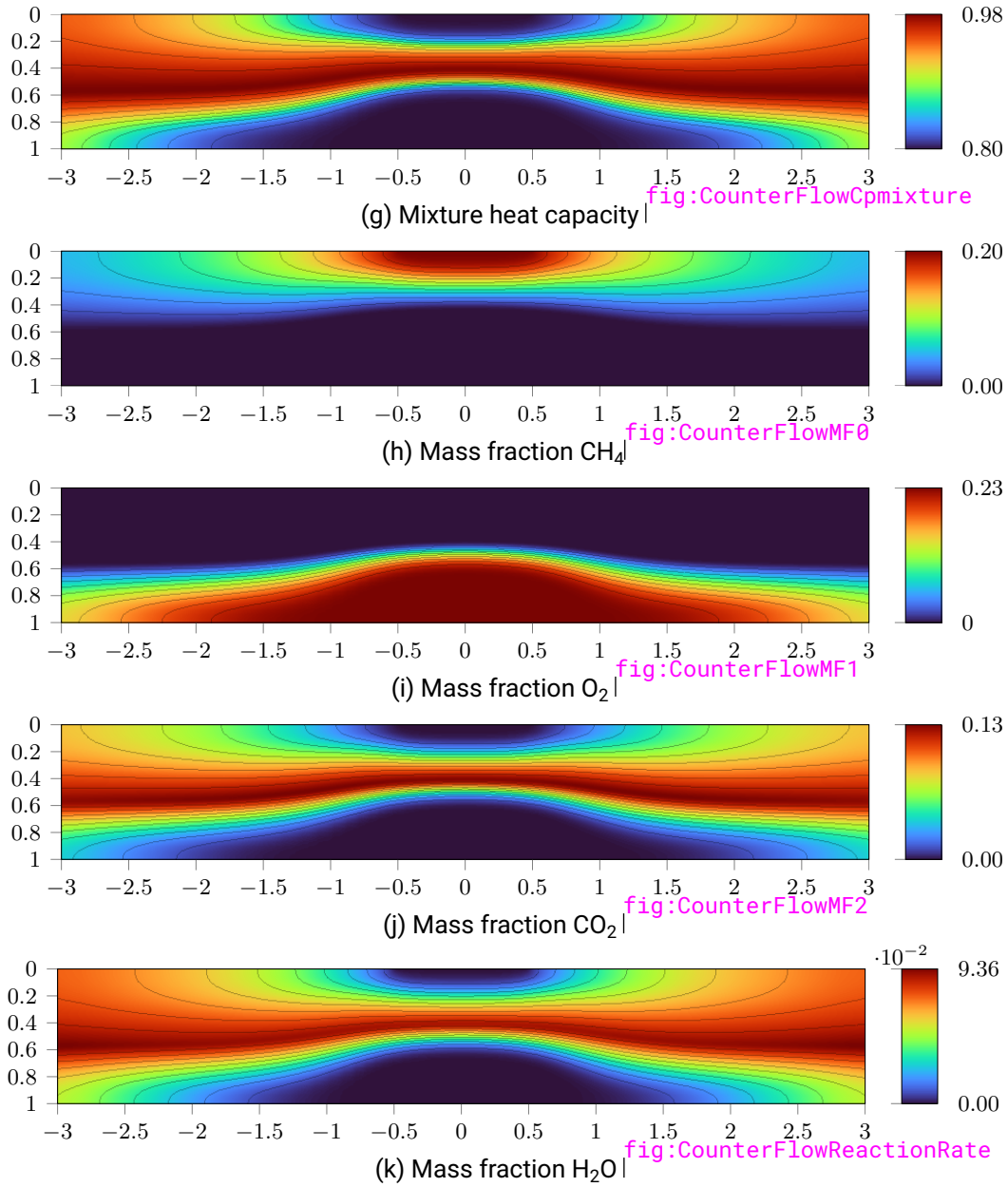


Figure 5.36: Nondimensional solution and derived fields of the counterflow flame configuration for case (a) (continued).

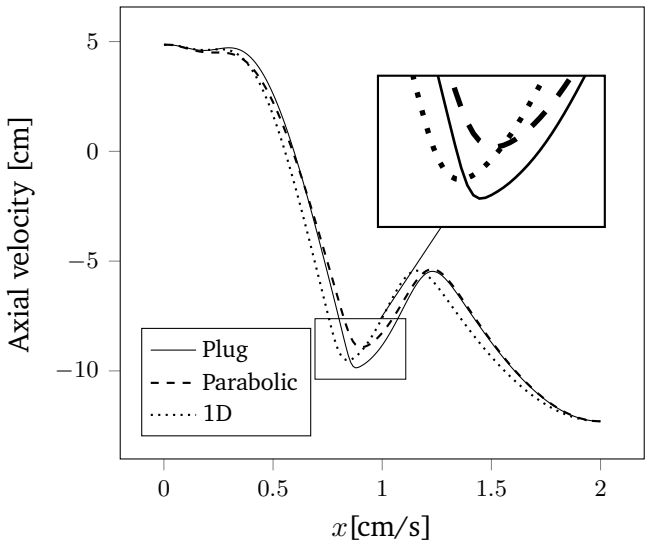


Figure 5.37: Velocity profiles of the counterflow diffusion flame for parabolic and plug inlet boundary conditions. fig:CounterFlowFlame_DifferentBoundaryConditions

forms close to the stagnation streamline. For this strain rate, a maximum dimensionless temperature of $T = 6.05$ is obtained (1815 K). This big increase in the temperature is also reflected in a big decrease in the density field, where a decrease of almost six times on the density values is observed. This change of density also causes the acceleration of fluid, as observed in Figure 5.36b.

In Figure 5.36k the reaction rate given by Equation (2.24) is plotted. It is interesting to see that the actual reacting zone is very small, which clearly demonstrates why adequate meshing is necessary to capture the steep gradients resulting from the strong and highly localized heat sources. Finally, and as expected, the fuel and oxidizers fields seem to only be found on either side of the flame. Although it cannot be seen here, some reactant leaking occurs, meaning that there exists a small zone where both species coexist. This point will be addressed later.

Comparison of two-dimensional and the quasi one-dimensional counterflow flames

In this section a comparison of the results obtained with the XNSEC solver for a two-dimensional counterflow diffusion flame, and the results obtained with the BVP4 solver for a quasi one-dimensional flame is made. This comparison is made along the centerline of the domain (see Figure 5.35). In this section, only dimensional variables will be considered. The transport parameters, chemical model and the equation of state are exactly the same for both formulations. For all calculations in this section done with the XNSEC solver, a polynomial degree of four is used for the velocity components, temperature and mass fractions. A polynomial degree of three is used for the pressure. This results in systems with approximately 439,000 degrees of freedom.

The choice of the type of velocity boundary conditions for the inlets requires some attention. Different possibilities exist to describe the velocity profiles. One possibility is to characterize the velocity boundary conditions by assuming a Hiemenz potential flow, where a single parameter defines the flow field. Other possibilities are also a constant velocity value (plug flow) or a

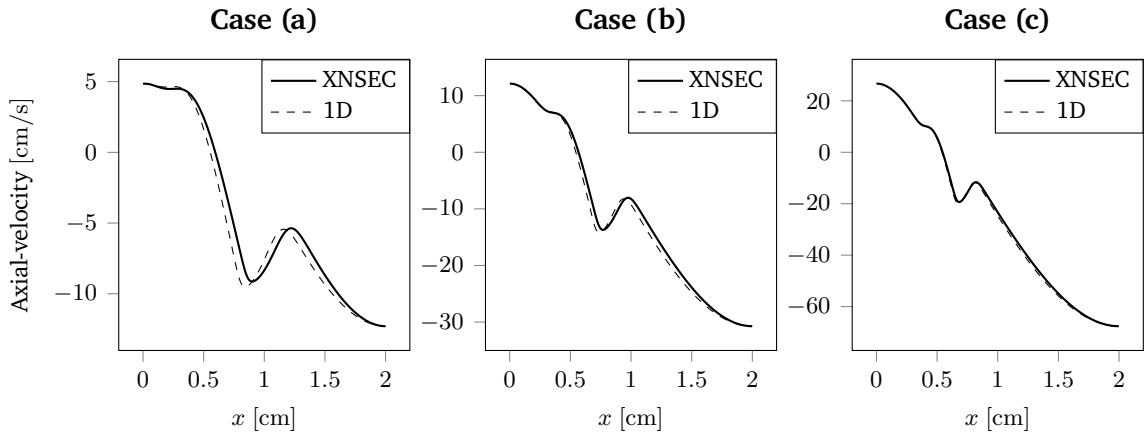


Figure 5.38: Comparison of the axial velocity calculated with the XNSEC solver and the one-dimensional approximation. fig:BoSSS_1D_Comparison_velocity

parabolic profile, which allows defining different velocity values for each jet inlet. The effect of boundary conditions on the flame structure has been studied by Chelliah et al. (1991) and Johnson et al. (2015), where it is concluded that both plug and potential flow are able to adequately describe experimental data.

The question of whether a plug or parabolic flow profile allows a better representation of the quasi one-dimensional equations was treated in the work from Frouzakis et al. (1998). There it is stated that the one and two-dimensional formulations yield very similar results, provided that the inlets of the two-dimensional configurations are uniform. Furthermore, preliminary calculations with the XNSEC solver showed that the selection of a plug flow or parabolic have an influence on the solution, as shown in Figure 5.37. Based on these results, the plug flow boundary condition is adopted for all following test cases.

In Figure 5.38 a comparison of the axial velocities calculated with the XNSEC solver and the one-dimensional solution is shown. While for the high strain case the results agree closely, for lower strains a discrepancy is observed. Recall that the derivation of the one-dimensional approximation assumes a constant velocity field incoming to the flame zone in order to obtain a self-similar solution. In the case of the two-dimensional configuration presented here, the border effects do have an influence on the centerline, which disrupts the self-similarity. This effect is more pronounced for low velocities, which explains the discrepancy between curves.

In Figure 5.39 the temperature and mass fraction fields are presented. Again, a discrepancy is observed for low strains, but results show a good agreement for higher inlet velocities. It can also be observed that, as expected, at higher strains a significant leakage of oxygen across the flame is present. This is a typical behavior of a flame that is getting closer to its extinction point (Fernandez-Tarrazo et al., 2006).

A drawback from the usual one-step models with constant activation temperature is that they tend to over predict fuel leakage. This behavior is not observed in the one-step model with variable activation temperature used here. In Figure 5.40 the comparison is shown for the configuration (c) between the mass fractions fields obtained using a chemical model with variable kinetic parameters given by Equations (2.19) and (2.20) and with constant kinetic parameters using $\hat{T}_a = \hat{T}_{a0}$ and $\hat{Q} = \hat{Q}_0$. The oxygen leakage obtained by using the chemical model with variable parameters is evident, demonstrating that the chemical model is capable

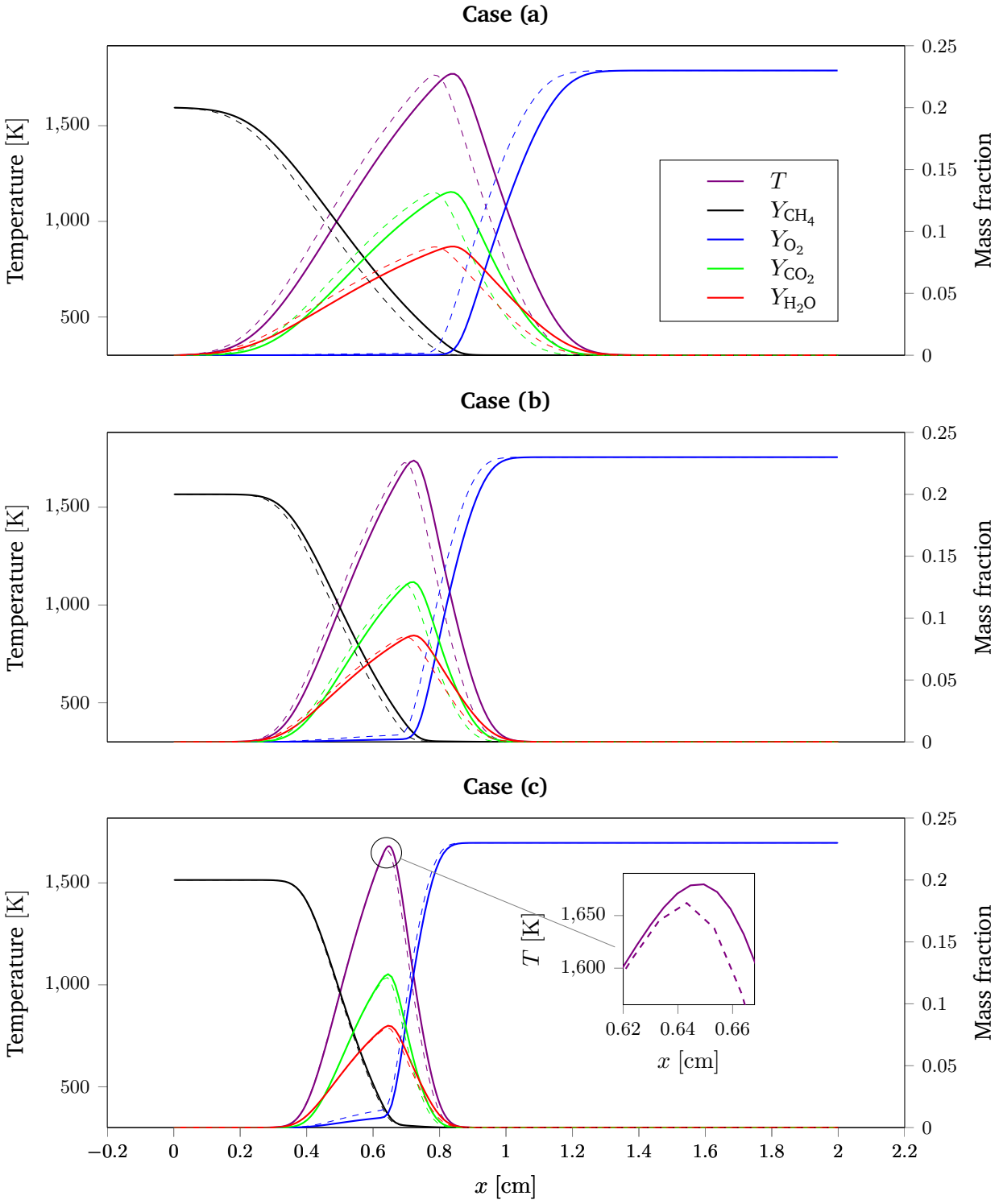


Figure 5.39: Comparison of temperature and mass fraction fields obtained with the XNSEC solver (solid lines) and the one-dimensional approximation (dashed lines). fig:BoSS_1D_Comparison

of appropriately modeling the leakage phenomenon.

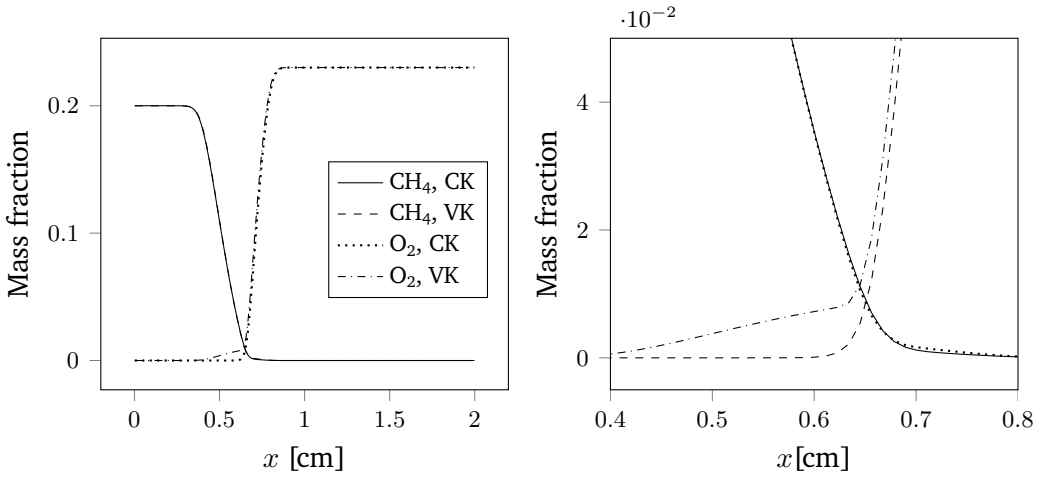


Figure 5.40: Oxygen leakage in the counterflow diffusion flame configuration.Fuel and oxidizer mass fraction profiles using variable kinetic parameters (VK) and constant kinetic parameters (CK) are shown. Right picture is zoomed in near the flame zone. fig:VarParams

In Figure 5.41 the maximum temperature obtained at the centerline for different strain rates is plotted. Qualitatively speaking, the solution obtained with the XNSEC solver agrees with the expectations. As the strain rate increases the residence time decreases, the system moves away from equilibrium and the maximum temperature decreases (see Figure 2.1). On the other hand, the comparison of values obtained with the XNSEC solver and those of the quasi-one-dimensional approximation clearly shows a discrepancy in the results. For low strain rates, this discrepancy is small, being only $10\,\mathrm{K}$ for $a = 20\,\mathrm{s}^{-1}$, approximately a difference of 0.5%. As the strain rate increases so does the discrepancy. For $a = 200\,\mathrm{s}^{-1}$ the difference is almost 50K, which is a 9% disagreement. A similar behavior is also reported in Frouzakis et al. (1998), where a difference of 50K was obtained between the results of a two-dimensional axisymmetric configuration and a quasi-one dimensional configuration.

It is worth noting that the XNSEC solver was not able to find a converged solution for $a > 202\,\mathrm{s}^{-1}$, and the Newton algorithm stagnates. This is most probably a sign of under-resolution of the mesh, and that the used refinement strategy did not help for such high strain rates. A better mesh refinement strategy is necessary for calculating the flame at conditions near the extinction point. Moreover, for high strain rates, the flame will be far from the thermochemical equilibrium, and it is likely that the solution obtained for the flame sheet will be far from the solution with finite reaction rates. A possibility would be to use one of the well-known continuation methods to progressively move in the direction of the extinction point (see, for example, Nishioka et al. (1996)). The homotopy methodology presented in Section 4.3 can be viewed as one of those methods and would be useful when looking for solutions of systems that are close to the extinction point, by gradually increasing the velocity of the inlets. A complexity that arises is how to create in a robust way a dynamical mesh that is suitable for obtaining the intermediate solutions while searching for the final result. This issue is beyond the scope of this thesis and may be the subject of future research.

The difference between the results obtained for the two-dimensional configuration and the quasi-one dimensional approximation could be explained by some condition within the 1D

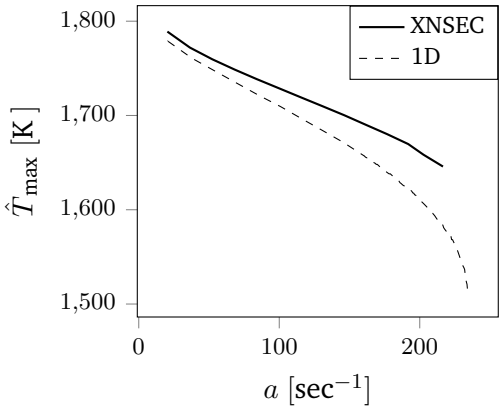


Figure 5.41: Maximum centerline temperature of a counterflow flame for different strains. fig:TemperatureStrainPlot

system assumptions being violated in the set-up of the 2D configuration. It is known that in addition to the boundary conditions, the ratio between the width of the slot and the separation between the two slots (here $D/H = 1.0$) also has an influence on the solution and that a high ratio is desirable (Frouzakis et al., 1998). Experiments with the XNSEC solver showed that increasing the ratio to $D/H = 1.5$ or decreasing it to $D/H = 0.75$ did not change the results appreciably.

Another point that was not addressed here is whether the boundary conditions chosen for the cold walls have an effect on the solution along the centerline. Other possibilities could have been using outlet boundary conditions or an adiabatic wall. However, it is expected that its effect on the centerline would not be big. The points mentioned here should be addressed in future work.

Temperature convergence study

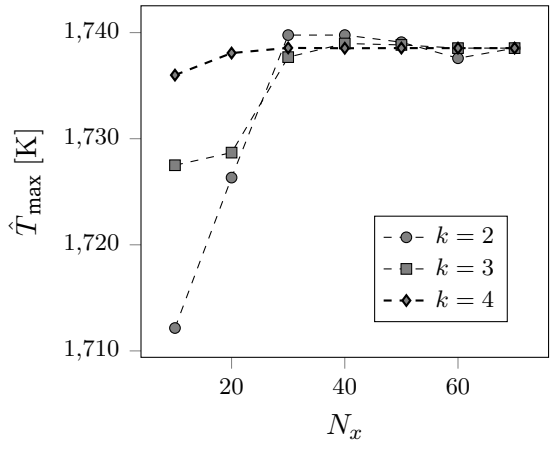


Figure 5.42: Convergence study of the maximum value of the temperature for the counterflow diffusion flame configuration. fig:TemperatureConvergenceDiffFlame

Similarly to problems presented in earlier sections, the presence of singularities caused by non-consistent boundary conditions causes a degenerative effect on the global error values,

making a global convergence study for this configuration problematic. However, it is still possible to study the behavior of some characteristic point value under different conditions to prove the mesh independence of the solution.

In Figure 5.42 it is shown how the maximum temperature along the centerline obtained for the case (b) behaves under different number of elements in the x direction (N_x) and polynomial degrees. The values for $k = 1$ are not shown, because for this range of cell elements, the maximum temperature value was of the order of 60K higher than the ones depicted here. The temperature tends to a limit value, and it is possible to observe how this value is reached already for coarse meshes when using a polynomial degree of three or four. For $k = 2$ the temperature also tends to a limit value, but at a slower rate compared to $k = 3$ or $k = 4$.

In the next section a simplified one-dimensional flame configuration will be used in order to be able to realize a global h-convergence study of the whole system operator.

5.3.3 Chambered diffusion flame

ss:UDF

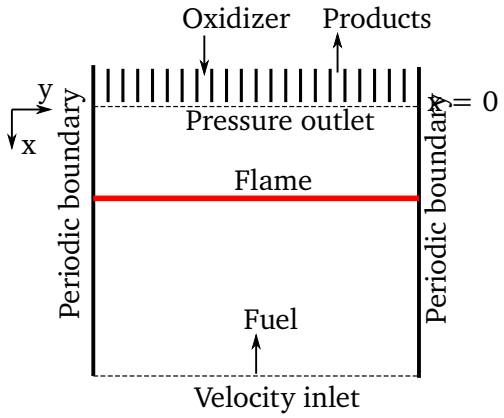


Figure 5.43: Schematic representation of the chambered diffusion flame configuration. fig:chamberedDiffFlame

In this chapter an h-convergence study for a quasi-one-dimensional configuration is shown. This is done by using a planar unstrained diffusion flame in the so-called chambered diffusion flame. This configuration has served as a model for many theoretical studies related to diffusion flames (Matalon et al. (1980), Rameau and Schmidt-Lainé (1985), and Matalon and Metzener (2010)).

A sketch of the configuration can be seen in Figure 5.43. Fuel is injected at a constant rate into the bottom of a small insulated chamber, while oxidant diffuses into the system against the direction of flow. Constant conditions at the outlet of the chamber are achieved in an experimental setting by a rapid renewal of the flow of the oxidant. Under these conditions, an unstrained planar flame is formed.

The fuel inlet into the chamber is modeled with a constant velocity inlet boundary condition Equation (2.37a), while the flow outlet at the top is considered an outlet as given by Equation (2.37d). Since the interest is in the flame far away from the container walls, it is sufficient to set the remaining boundary conditions as periodic boundaries. This effectively transforms the problem into a pseudo-two-dimensional configuration.

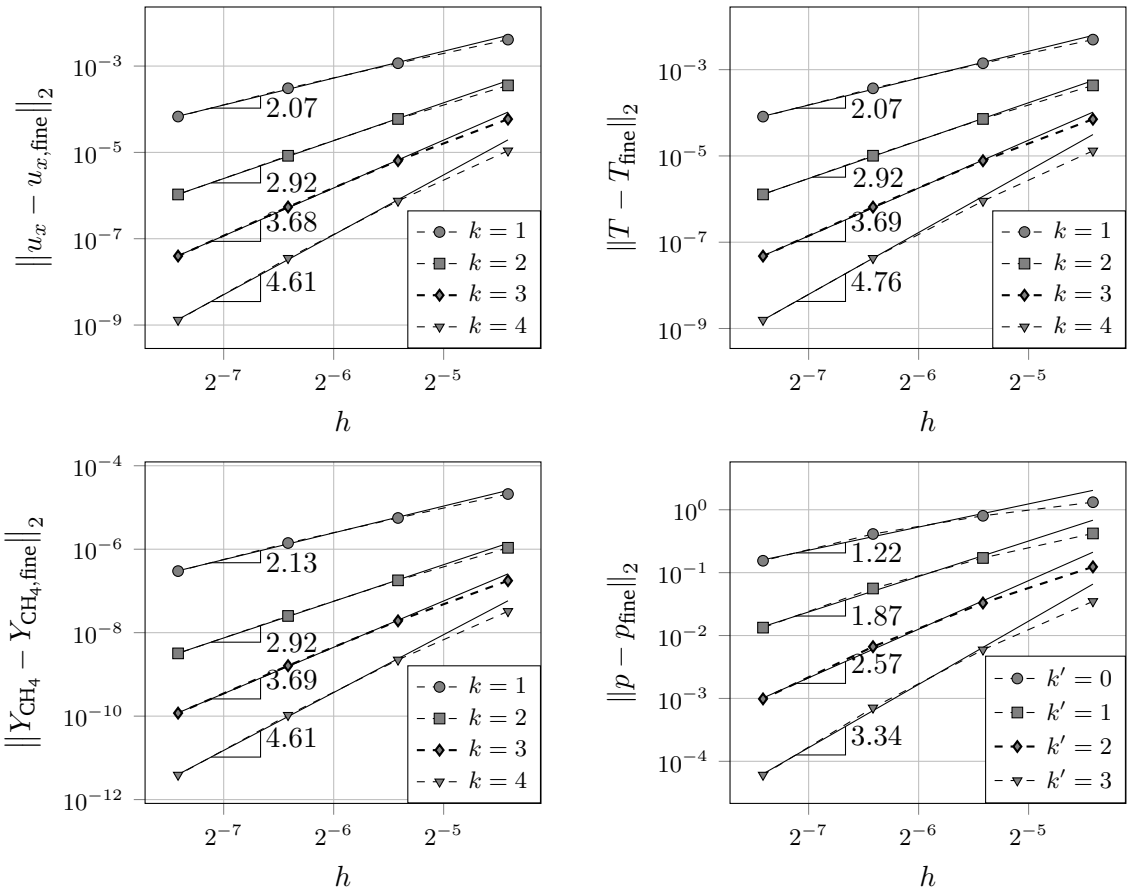


Figure 5.44: Convergence study for the chambered diffusion flame configuration. ConvergenceDiffFlame

The inlet velocity of the fuel jet is set to $2.5\,\mathrm{cm\,s^{-1}}$ and its mass composition is $Y^0_{\mathrm{CH_4}} = 0.2$ and $Y^0_{\mathrm{N_2}} = 0.8$ while air has a composition $Y^0_{\mathrm{O_2}} = 0.23$ and $Y^0_{\mathrm{N_2}} = 0.77$. The temperature of the fuel and air feed streams is $300\,\mathrm{K}$. The length of the system L is equal to $0.015\,\mathrm{m}$. The Reynolds number is $\mathrm{Re} = 2$

For this configuration, an h-convergence study is conducted, where uniform Cartesian meshes with 5×2^6, 5×2^7, 5×2^8, 5×2^9 and 5×2^{10} cells are used. The polynomial degrees are varied from one to four for velocity, temperature and mass fractions, and from zero to three for pressure. Errors are calculated using the finest mesh as a reference solution.

The results are shown in Figure 5.44 for variables u, T, $Y_{\mathrm{CH_4}}$ and p. The convergence results for other variables are similar and not shown here. The expected convergence rates characteristic for the DG-method are observed. For low polynomial degrees the orders of convergence are very close to the theoretical values. However for higher polynomial degrees a slight deterioration of the convergence rate is observed.

5.3.4 Combustion over a square cylinder

As a last testcase of the XNSEC solver, a configuration very similar to the one previously shown for the flow over a cylinder was simulated, but now extending the case to a system where

combustion exists. Unlike Section 5.2.4, for this test case the flow over a square cylinder is considered. The simulation is non-stationary. As will be explained later, this was not possible with the current solver implementation, and only some simplified cases will be shown. First, results for a non-reactive steady state solution are shown. Later, an unsteady case with combustion present is analyzed.

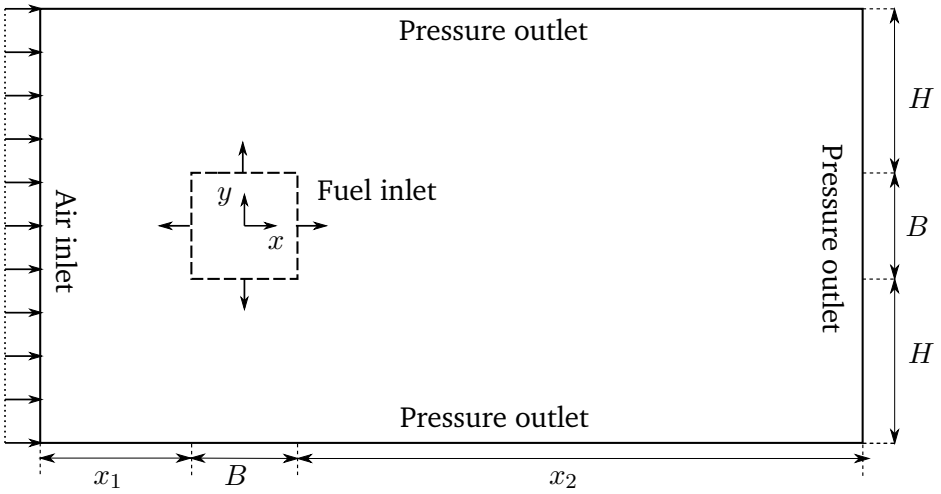


Figure 5.45: Temperature field and mesh of the unsteady combustion over a square cylinder. fig:CombustionSquareCylinder

Flow over a heated square cylinder

In the work from Miao (2022) the XNSEC solver was used for calculating the flow around a heated square cylinder. The results were compared with the data published by Sharma and Eswaran (2004). Simulations were calculated for the test case with Reynolds numbers ranging from Re $= 5$ to Re $= 40$. The square cylinder is modeled by no-slip walls with $(u, v) = (0, 0)$ and the incoming flow field (air inlet in Figure 5.45) is $(u, v) = (1, 0)$. The fluid considered is air. In Figure 5.46 a comparison of the normalized recirculation length L_r/B obtained with the XNSEC solver and the reference is shown. The results agree very well, deviating slightly at higher Reynolds numbers.

Unsteady combustion over a square cylinder

The last test of the XNSEC solver was aimed at calculating a non-stationary case with combustion. It is well known that there is a critical Reynolds value (at least for non-reactive systems) from which the system becomes non-stationary. According to Sharma and Eswaran (2004), for Re > 50 the flow has a non-stationary periodic character. In this section results from two simulations are shown. For the first one, a constant density is assumed, and in the second one the equation of state is used. A sketch of the configuration is shown in Figure 5.45. Fuel is expelled at a constant rate and homogeneously through the cylinder. A constant flow of air in the horizontal direction comes in contact with the fuel, which eventually forms a flame.

The lengths depicted in Figure 5.45 are set to $B = 1$, $x_1 = 4$, $x_2 = 22$ and $H = 8$. The air enters with a uniform velocity to the system $(u, v) = (1, 0)$, and a uniform temperature

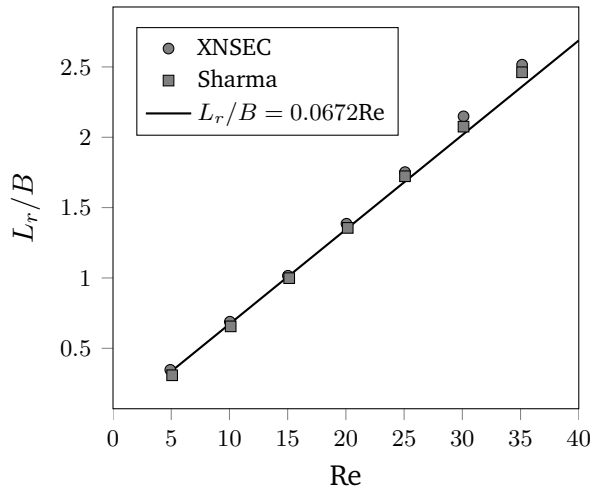


Figure 5.46: Recirculation lengths for different Reynolds numbers. Fig.RecirculationLength

$T = 1$. Its composition is $Y^O_{O_2} = 0.23$ and $Y^O_{N_2} = 0.77$. The fuel inlet enters with a velocity field $(u, v) = (0.2x/B, 0.2y/B)$, has a uniform temperature $T = 1$ and composition $Y^F_{CH_4} = 0.2$ and $Y^F_{N_2} = 0.8$. A Reynolds number of Re $= 300$, for which in the non-reactive case the vortex shedding phenomenon occurs. The Prandtl number is set to Pr $= 0.75$.

For this simulations only the infinite-reaction rate equations are calculated and advanced in time. This is mainly done in order to reduce the calculation time, but also because the goal is to test the capability of the solver to calculate flow fields in an unsteady state that are greatly influenced by the temperature.

The initial conditions used are similar to those of the circular cylinder in section Section 5.2.4. Again, an initial vortex is used to trigger the vortex shedding according to

$$u(t = 0) = 1 + u^{\text{vortex}}, \tag{5.28a}$$
$$v(t = 0) = 0 + v^{\text{vortex}}, \tag{5.28b}$$
$$T(t = 0) = 1, \tag{5.28c}$$
$$p(t = 0) = 0. \tag{5.28d}$$

where the vortices u^{vortex} and u^{vortex} are given by Equation (5.18). The strength of the vortex is $a = 1$ and its center is initially located at $(x^o, y^o) = (-2.5, 0)$. A Cartesian base mesh with 54×32 elements is used, which is refined or coarsened as necessary during the simulation. The calculation time is set to $t = 100$, with constant time steps of $\Delta t = 0.05$. A BDF-2 scheme is used for the temporal discretization.

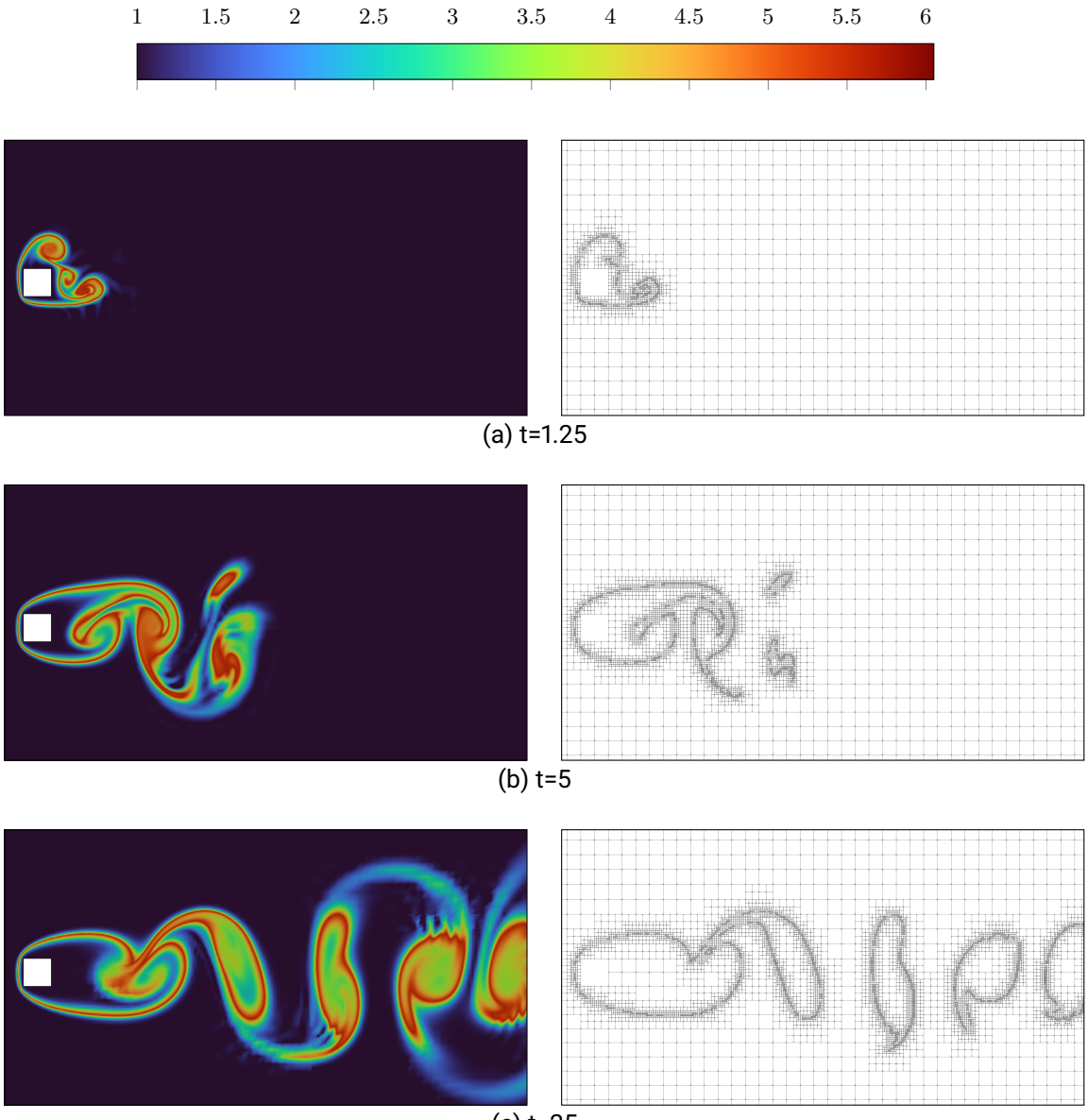
The time-dependent simulation of a combustion phenomenon proved to be very challenging and the temporal discretization described in Section 3.2.3 did not allow obtaining solutions of this problem. It is well known that the inclusion of the $\partial\rho/\partial t$ term of the continuity equation in the source term, as done in the present work, is a source of numerical instability. In the work of Nicoud (2000) it is reported that obtaining solutions for density ratios greater than three becomes difficult. It was previously shown in Section 5.2.4 and Section 5.2.5 that the XNSEC solver was able to compute unsteady non-isothermic flows. However, those test cases presented moderate density ratios, the largest of them being 1.5. On the other hand, for a combustion process like the one presented here, the density ratios are much higher, even higher than six

for typical combustion cases. In Rauwoens et al. (2009) a similar destabilization effect is also reported for high density ratios.

Nevertheless, simulations of this test case ignoring the $\partial\rho/\partial t$ term were performed. This is a big and nonphysical approximation, but is nonetheless used to showcase the capability of the solver for calculating unsteady reacting flow.

First, the problem was calculated with the assumption of a constant density $\rho = 1$. By doing this, the term $\partial\rho/\partial t$ is automatically equal to zero, and the momentum equation is only slightly coupled to the mixture fraction equation, since the viscosity still depends on the temperature. In Figure 5.47 the temperature field is shown at different times. Obviously, since this is a simplified case, using the flame sheet circumvents the need to simulate the ignition of the system, and from the instant $t = 0$ onward the system already has points where the temperature reaches the adiabatic temperature. It can be seen that the vortex-shedding phenomenon appears, as expected for a number Re $= 300$. It is interesting to see thatthe vortex shedding separation point moves downwards (compared with a square cylinder without outflow) since the incoming air flow is pushed by the fuel exiting the cylinder. The mesh obtained from the mesh refinement process is also shown. The mesh is refined or coarsened after each time-step by using as a criterion the location of the flame sheet, meaning that cells where $z = z_{\text{st}}$ are refined.

Finally, the configuration using a variable density was calculated. The temperature fields and meshes are shown in Figure 5.48. It is interesting to observe the great effect that the big density variations caused by the combustion have on the flow structure. The zones near the flame sheet are greatly accelerated because of the big reduction in the density. The refinement strategy mentioned before was used, and was observed to be critical for finding solutions of the system. Interestingly the vortex shedding doesn't seem to appear in this case, at least for the times calculated here.

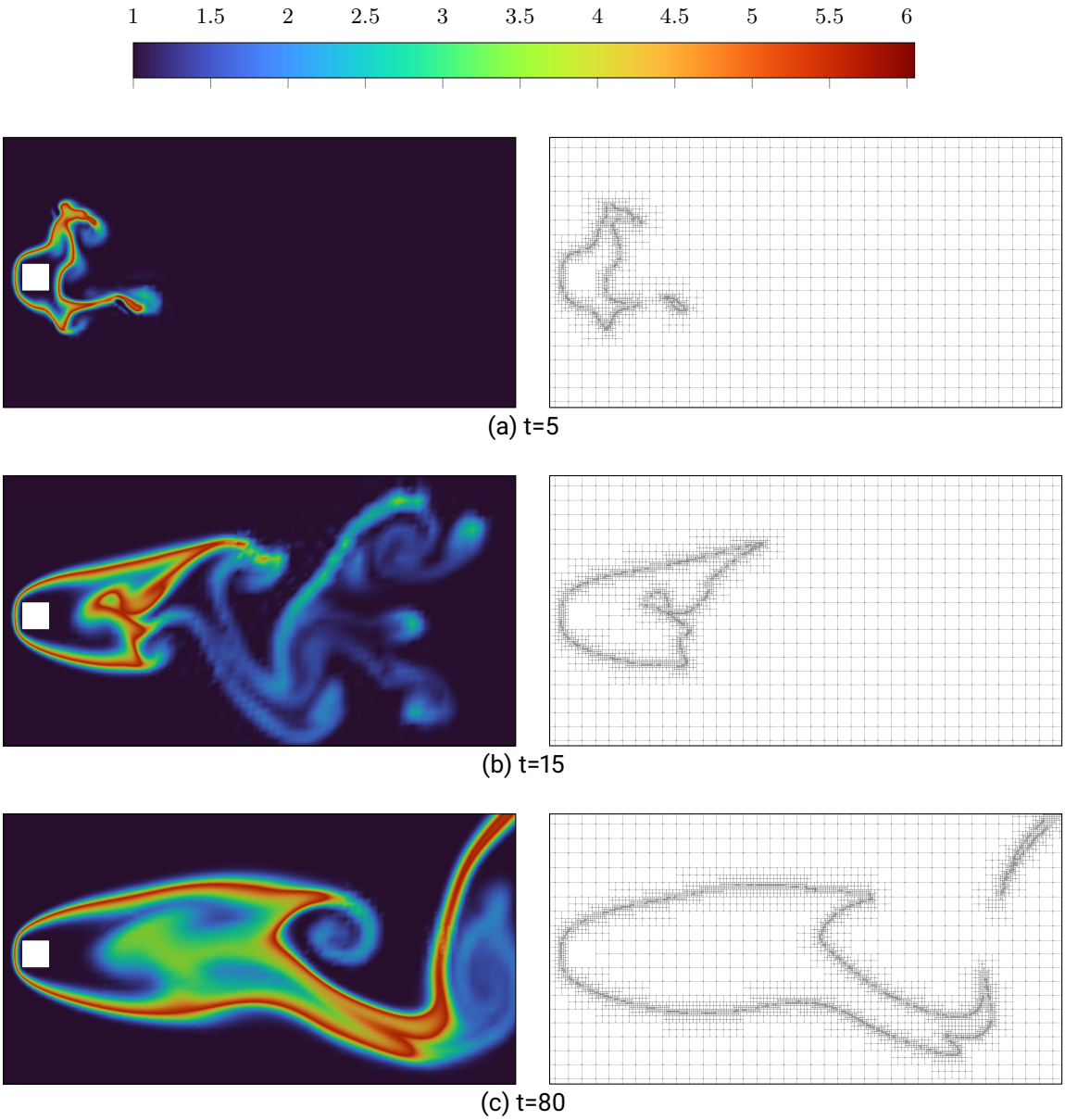


(a) t=1.25

(b) t=5

(c) t=25

Figure 5.47: Temperature field and mesh calculated with the Burke-Schumann solution at different times, assuming a constant density. fig:CombustionOverCylinder_CD



(a) t=5

(b) t=15

(c) t=80

Figure 5.48: Temperature field calculated with the Burke-Schumann solution at different times. fig:CombustionOverCylinder

6 Conclusion

ch:conclusion

In the present work we have shown the discretization using the DG method for the reactive low-Mach equations. A mixed order formulation has been used, where velocity, temperature and mass fractions are represented by polynomials of degree k, and pressure by polynomials of degree $k-1$. The system obtained from the discretization was solved by means of a Newton-Dogleg type method. Additionally, a homotopy strategy for solving highly nonlinear systems was introduced and used for some of the presented test cases . For the reactive test cases the concept of the flame sheet estimate was demonstrated to be a useful and computationally cheap way of initializing the finite reaction rate combustion calculations. The solver was used to calculate three different benchmark configurations, one without combustion and two cases with combustion. The first of these is the differentially heated cavity. This benchmark case was solved for varying Rayleigh numbers, spanning from Ra $= 10^2$ to Ra $= 10^7$. It was observed that for cases with Rayleigh number larger than 10^5 the use of our homotopy strategy was necessary to ensure convergence. Velocity and temperature profiles as well as thermodynamic pressure and Nusselt number were compared with a reference solution, obtaining very satisfactory results, thus validating our implementation of the method for variable density systems. An h-convergence study was presented for the differentially heated cavity, obtaining the expected convergence rates up to $k = 4$, where some degeneration on the rates was observed. Furthermore the counterflow diffusion flame configuration was analyzed, with which we intended to test the implemented chemistry model in conjunction with our solver. The results obtained with the XNSEC solver were compared with results obtained by solving the self-similar one-dimensional representation of the counter diffusion flame configuration for varying strain rates. While for low strains some discrepancies between results were observed, the difference narrowed for high strains, which can be explained by the influence of the border effects on the centerline results for a two-dimensional configuration. Finally, the chambered diffusion flame configuration was used to study the convergence rates of our method. It was shown that, as expected, the convergence rates increase when using higher degree polynomials, but with some deterioration compared to the theoretical expected convergence. In future work the implemented solver is intended to be used in conjunction with our extended-DG solver Kummer (2017), Kummer et al. (2021), and Krause and Kummer (2017)in order to study multiphase reactive systems such as droplets.

Future Work: The governing equations treated in this work were based on some strong assumptions. In particular, the one-step model chemical model (esto no es tan malo). El modelo de difusion utilizado es altamente simplificado, y se espera que en ciertos sistemas con combustion pueda arrojar resultados con un error apreciable, en particular en sistemas que no se encuentren altamente diluidos. La implementacion de un modelo de difusion más complejo, como por ejemplo la Hirschfelder and Curtiss approximation sería una forma simple y eficiente de solucionar el problema. En este trabajo se trató principalmente con sistemas de combustion en estado estacionario. El uso de la flame sheet solution como estimado inicial

probó ser una manera eficiente de encontrar la burning solution. Con esta estrategia, se circumvent la necesidad de simular el proceso de inición de la flama puesto que solo se está interesado en el steady state solution. La simulación del proceso de ignicion es un topico abierto que debe ser tratado en futuros trabajos. Si bien en el trabajo se demostró que el fully coupled approach funcionó muy bien para una gran variedad de problemas, para sistemas más complejos, como por ejemplo procesos de combustion no estacionarios, los tiempos de calculo podrían resultar prohibitivos. Una mayor paralelización computacional, en particular de los linear-solvers, podrían brindar aceleración a los calculos. Este es un topico de actual estudio en el departamento de mecanica de fluidos FDY, en donde distintos metodos para resolucion de sistemas de ecuaciones están siendo estudiados, y en particular con aplicaciones en la mecánica de fluidos.

Las simulaciones con combustion ocupan combustible diluido. Simulaciones con combustion

A Appendix

ch:appendix

A.1 Main Counterflow flame 1D

```
clc; clear all; close all;
%% Velocity multiplier which defines the strain
velMult = 3;
%% Problem geometry description
C.L = 0.02; % Domain length, m
C.cp = 1.3; % Heat capacity;
C.initialCellNumber = 100;
C.chemActive = true;
C.variableKineticParameters = true;
%% Boundary conditions
C.vLeft = 0.0243 * velMult; % Velocity left (fuel)
C.vRight = -0.0243 * velMult * 3; % Velocity right (oxidizer)
C.TF0 = 300; % Temperature left (fuel)
C.TO0 = 300; % Temperature right (oxidizer)
C.CompositionFuel = [0.2, 0.0, 0.0, 0.0, 0.8];
C.CompositionOx = [0.0, 0.23, 0.0, 0.0, 0.77];
%% Physical parameters
C.p0 = 101325; % Pressure, Pa
C.viscosity0 = 1.716e-5; % kg/( m s) ==> viscosity at T = 273.15
   for air
C.Tref = 273; % Reference temperature from powerlaw
C.a = 2 / 3; % PowerLaw exponent
C.Coef_Stoic = [-1, -2, 1, 2, 0]; % Chemical reaction 1CH4 + 2O2
   -> 1CO2 + 2H2O
C.MM = [16, 32, 44, 18, 28];
C.s = (C.Coef_Stoic (2) * C.MM(2)) / (C.Coef_Stoic(1) * C.MM(1));
C.phi = C.s * C.CompositionFuel(1) / C.CompositionOx(2);
C.zst = 1.0 / (1.0 + C.phi);
C.Q = 50100;
C.R = 8.314 * 1000; % gas constant, kg m^2 / (s^2 K mol)
%% Calculate flame sheet (infinite reaction rate)
x = zeros(1);
FlameSheetSolution = zeros(1);
lambdaOut = 100;
[sol, lambdaOut] = CounterFlowFlame_MixtureFraction(C);
x = sol(1, :); % x coordinates
```

```
dummyZero = zeros(1, length(x));
FlameSheetSolution = [sol(2, :); sol(3, :); sol(4, :); sol(5, :);
    ...
     dummyZero; sol(6, :); dummyZero; sol(7, :); dummyZero; sol(8,
        :); ...
     dummyZero; sol(9, :); dummyZero; sol(10, :); dummyZero];
%% Finite reaction rate calculation
[xint, Sxint, calculatedStrain, calculatedMaxTemperature,
   lambdaOut] = CounterFlowFlame_FullChemistry(C,
   FlameSheetSolution, x, lambdaOut, true);

% Export results for comparison with two dimensional formulation
```

A.2 Matlab code for Flame Sheet problem

```
function [solutionArray, lambdaOut] =
   CounterFlowFlame_MixtureFraction(myconfig)
options = bvpset('stats', 'off', 'NMax', 10000, 'AbsTol', 1e-8);

%% Solver configuration
lambda = -100; %Initial estimation for dpdz
TF0 = myconfig.TF0;
TO0 = myconfig.TO0;
Q = myconfig.Q;
cp = myconfig.cp;
zst = myconfig.zst;
L = myconfig.L;
YF0 = myconfig.fuelInletConcentration(1);
YO0 = myconfig.oxidizerInletConcentration(2);
Coef_Stoic = myconfig.Coef_Stoic;
MM = myconfig.MM;
Tref = myconfig.Tref;
a = myconfig.a;
Prandtl = 1;

%% Solve system
sol = bvpinit(linspace(0, L, myconfig.initialCellNumber), @
   mat4init, lambda);
sol = bvp4c(@mat4ode, @mat4bc, sol, options);
Sxint = deval(sol, linspace(0, L, 200));
lambdaOut = sol.parameters;
Sxint = deval(sol, sol.x);

%% Recover primitive variables from Z
ZArray = Sxint(4, :);
for i = 1:length(ZArray)
```

```
    z_T(i) = getTemperatureFromZ(ZArray(i));
    z_Y1(i) = getY1FromZ(ZArray(i));
    z_Y2(i) = getY2FromZ(ZArray(i));
    z_Y3(i) = getY3FromZ(ZArray(i));
    z_Y4(i) = getY4FromZ(ZArray(i));
    z_Y5(i) = getY5FromZ(ZArray(i));
end
solutionArray = [sol.x; Sxint(1, :); Sxint(2, :); Sxint(3, :); z_T
   ; z_Y1; z_Y2; z_Y3; z_Y4; z_Y5; Sxint(4, :)];

%% ========================
% Note: the system to be solved is of first order on v, second
   order on U,
% and second order on the scalars (T, Y1,Y2...)
% Second order equations are brought to a first order by
   introducing a
% transformation
function dydx = mat4ode(~, y, lambda) % equation being solved
    v = y(1);
    U1 = y(2);
    U2 = y(3);
    Z1 = y(4); % Z
    Z2 = y(5); % dZ/dy
    if (Z1 > 1) %Limit values
        Z1 = 1;
    end
    if (Z1 < 0)
        Z1 = 0;%Limit values
    end

    rho_ = GetRhoFromZ(Z1);
    mu_ = GetMuFromZ(Z1) / Prandtl;
    % Calculation of density and viscosity derivatives is
       problematic,
    % because the derivative at zSt is not defined.
    %Is simply ignored (seems to still give an adequate starting
       solution )
    rho_p = 0.0;
    mu_p = 0.0;

    dydx = [(-rho_ * U1 - rho_p * v) / rho_; ... %Conti
        U2; ... %Mom
        (lambda + rho_ * v * U2 + rho_ * U1^2 - mu_p * U2) * 1.0 /
            mu_; ... %Mom
        Z2; ... .%MassFraction
        (rho_ * v * Z2 - mu_p * Z2) / mu_; ... %MassFraction
```

```
        ];
end


function res = mat4bc(ya, yb, lambda) % boundary conditions
    va = ya(1);
    U1a = ya(2);
    Za = ya(4);

    vb = yb(1);
    U1b = yb(2);
    Zb = yb(4);

    res = [va - myconfig.vLeft; ... % v(0) = v0
        vb - myconfig.vRight; ... % v(L) = vl
        U1a - 0; ... % U(0) = 0
        U1b - 0; ... % U(L) = 0
        Za - 1.0; ... % Z(0) = 1.0
        Zb - 0.0; ... %Z(L) = TL
        ];
end

function yinit = mat4init(x) % initial guess function.
    yinit = [0.0; ... %v
        0.0; ... %U1
        0.0; ... %U2
        0.5; ... % Z1
        0.0; ... % Z2
        ];
end
% Definition of helper functions for density and transport
   parameters
function viscosity = mu(T)
    viscosity = myconfig.viscosity0 * (T / Tref)^(a);
end
function density = rho(T, Yk)
    mult = 0.0;
    for c = 1:(length(Yk))
        mult = mult + Yk(c) / MM(c);
    end
    density = myconfig.p0 / (myconfig.R * T * mult); % kg / m^3
end
%=============================================
%Transformations from Z

function Temperature_fromZ = getTemperatureFromZ(Z)
```

```
    if (Z >= zst)
        Temperature_fromZ = Z * TF0 + (1 - Z) * TO0 + Q * YF0 / cp
            * zst * (1 - Z) / (1 - zst);
    else
        Temperature_fromZ = Z * TF0 + (1 - Z) * TO0 + Q * YF0 / cp
            * Z;
    end
end

function Y1_fromZ = getY1FromZ(Z)
    if (Z >= zst)
        Y1_fromZ = YF0 * (Z - zst) / (1 - zst);
    else
        Y1_fromZ = 0.0;
    end
end

function Y2_fromZ = getY2FromZ(Z)
    if (Z >= zst)
        Y2_fromZ = 0;
    else
        Y2_fromZ = YO0 * (1 - Z / zst);
    end
end

function Y3_fromZ = getY3FromZ(Z)
    nu_CO2 = Coef_Stoic(3);
    nu_O2 = Coef_Stoic(2);
    nu_CH4 = Coef_Stoic(1);
    MW_CO2 = MM(3);
    MW_O2 = MM(2);
    MW_CH4 = MM(1);
    if (Z >= zst)
        Y3_fromZ = -YO0 * (nu_CO2 * MW_CO2) / (nu_O2 * MW_O2) * (1
            - Z);
    else
        Y3_fromZ = -YF0 * (nu_CO2 * MW_CO2) / (nu_CH4 * MW_CH4) *
           Z;
    end
end

function Y4_fromZ = getY4FromZ(Z)
    nu_H2O = Coef_Stoic(4);
    nu_O2 = Coef_Stoic(2);
    nu_CH4 = Coef_Stoic(1);
    MW_H2O = MM(4);
```

```
    MW_O2 = MM(2);
    MW_CH4 = MM(1);
    if (Z >= zst)
        Y4_fromZ = -YO0 * (nu_H2O * MW_H2O) / (nu_O2 * MW_O2) * (1
            - Z);
    else
        Y4_fromZ = -YF0 * (nu_H2O * MW_H2O) / (nu_CH4 * MW_CH4) *
           Z;
    end
end

function Y5_fromZ = getY5FromZ(Z)
    if (Z >= zst)
        YNOxi0 = 1.0 - YO0;
        YNFuel0 = 1.0 - YF0;
        Y5_fromZ = YNOxi0 * (1 - Z) + YNFuel0 * Z;
    else
        YNOxi0 = 1.0 - YO0;
        YNFuel0 = 1.0 - YF0;
        Y5_fromZ = YNOxi0 * (1 - Z) + YNFuel0 * Z;
    end
end

function MuFromZ = GetMuFromZ(z)
    MuFromZ = mu(getTemperatureFromZ(z));
end

function rhoFromZ = GetRhoFromZ(Z)
    rhoFromZ = rho(getTemperatureFromZ(Z), [getY1FromZ(Z),
       getY2FromZ(Z), getY3FromZ(Z), getY4FromZ(Z), getY5FromZ(Z)
       ]);
end
end
```

A.3 Matlab code for Finite reaction rate problem

```
%% Solves the one-dimensional counterflow diffusion flame in
   cartesian coordinates
function [xint, Sxint, calculatedStrain, maxTemperature, lambdaOut
   ] = CounterFlowFlame_FullChemistry2(C, flameSheetSol, xinit,
   lambda)

%% Solver configuration
variableKineticParameters = C.variableKineticParameters;
Prandtl = 0.75;
```

```
%% Problem description
L = C.L; % Domain length, m
cp = C.cp; % Heat capacity;
s = C.s;
p0 = C.p0; % Pressure, Pa
R = C.R; % gas constant, kg m^2 / (s^2 K mol)
MM = C.MM;
v0 = C.vLeft;
vl = C.vRight;
nu = C.nu;
Tref = C.Tref;
B = 6.9e11; %  Pre-exponential factor 6.9e14 cm3/(mol s) =>  6.9
   e11m3/(kmol s)
Ta0 = 15900; %K
massHeatRelease = C.Q * MM(1);
viscosity0 = C.viscosity0;

%% Boundary Condition values
TF0 = C.TF0; % Temperature left (fuel)
TO0 = C.TO0; % Temperature right (oxidizer)
YLeft = C.fuelInletConcentration;
YRight = C.oxidizerInletConcentration;
YF0 = YLeft(1);
YO0 = YRight(2);
sol = bvpinit(xinit, @mat4init, lambda);
sol.y = flameSheetSol;

%% ==================================
% First, solve system for a Constant Cp and constant Lewis numbers
cpconstant = true;
Le = [1.0, 1.0, 1.0, 1.0, 1.0];
options = bvpset('stats', 'true', 'Vectorized', 'off', 'NMax',
   10000, 'AbsTol', 1e-6, 'RelTol', 1e-3);
sol = bvp4c(@mat4ode, @mat4bc, sol, options);
lambda = sol.parameters;

%% ==================================
% Now, turn on variable Cp and non unity Lewis numbers and solve
   system.
cpconstant = false;
Le = [0.97, 1.11, 1.39, 0.83, 1.0];
sol = bvp4c(@mat4ode, @mat4bc, sol, options);
Sxint = deval(sol, linspace(0, L, 1000));
calculatedStrain = max(abs(Sxint(2, :)));
fprintf('Strain (biggest du/dy magnitude): %7.3f.\n',
   calculatedStrain);
```

```
maxTemperature = max(Sxint(4, :));
fprintf('Maximum temperature: %7.3f.\n', maxTemperature);

xint = linspace(0, L, 200);
Sxint = deval(sol, xint);
lambdaOut = sol.parameters;

%%
% Note: the system to be solved is of first order on v, second
   order on U,
% and second order on the scalars (T, Y1,Y2...)
% Second order equations are brought to a first order by
   introducing a
% transformation
    function dydx = mat4ode(x, y, lambda) % equation being solved
        v = y(1);
        U1 = y(2);
        U2 = y(3);
        T1 = y(4);
        T2 = y(5);
        Y1_1 = y(6);
        Y1_2 = y(7);
        Y2_1 = y(8);
        Y2_2 = y(9);
        Y3_1 = y(10);
        Y3_2 = y(11);
        Y4_1 = y(12);
        Y4_2 = y(13);
        Y5_1 = y(14);
        Y5_2 = y(15);
        Yk_1 = [Y1_1; Y2_1; Y3_1; Y4_1; Y5_1];
        Yk_2 = [Y1_2, Y2_2, Y3_2, Y4_2, Y5_2]; % Array of
           derivatives

        T1(T1 < 300) = 300; % "repair" values
        Yk_1(Yk_1 < 0) = 0;
        Yk_1(Yk_1 > 1) = 1;

        %% Density and transport parameters
        rho_ = rho(T1, Yk_1);
        rho_p = drhody(T1, T2, Yk_1, Yk_2);
        mu_ = mu(T1);
        mu_p = dmu_dy(T1, T2);
        if cpconstant
            cp = C.cp;
        else
```

```
        cp = getMixtureCp(T1, Yk_1, ["CH4", "O2", "CO2", "H2O
            ", "N2"]);
    end
    k_cp = mu_ / (Prandtl); % k/cp = mu / pr
    k_cp_p = mu_p / (Prandtl);
    rhoD1 = mu_ / (Prandtl * Le(1));
    rhoD2 = mu_ / (Prandtl * Le(2));
    rhoD3 = mu_ / (Prandtl * Le(3));
    rhoD4 = mu_ / (Prandtl * Le(4));
    rhoD5 = mu_ / (Prandtl * Le(5));
    rhoD1_p = mu_p / (Prandtl * Le(1));
    rhoD2_p = mu_p / (Prandtl * Le(2));
    rhoD3_p = mu_p / (Prandtl * Le(3));
    rhoD4_p = mu_p / (Prandtl * Le(4));
    rhoD5_p = mu_p / (Prandtl * Le(5));
    Q = getHeatRelease(Yk_1);
    omega = getReactionRate(T1, Yk_1);
    %
    dydx = [(-rho_ * U1 - rho_p * v) / rho_; ... %Conti
        U2; ... %Mom
        (lambda + rho_ * v * U2 + rho_ * U1^2 - mu_p * U2) *
            1.0 / mu_; ... %Mom
        T2; ... .%Energy
        (rho_ * v * T2 - k_cp_p * T2 - Q * omega / cp) / (k_cp
            ); ... %Energy
        Y1_2; ... .%MassFraction1
        (rho_ * v * Y1_2 - rhoD1_p * Y1_2 - omega * nu(1) * MM
            (1)) / (rhoD1); ... %MassFraction1
        Y2_2; ... .%MassFraction2
        (rho_ * v * Y2_2 - rhoD2_p * Y2_2 - omega * nu(2) * MM
            (2)) / (rhoD2); ... %MassFraction2
        Y3_2; ... .%MassFraction3
        (rho_ * v * Y3_2 - rhoD3_p * Y3_2 - omega * nu(3) * MM
            (3)) / (rhoD3); ... %MassFraction3
        Y4_2; ... .%MassFraction4
        (rho_ * v * Y4_2 - rhoD4_p * Y4_2 - omega * nu(4) * MM
            (4)) / (rhoD4); ... %MassFraction4
        Y5_2; ... .%MassFraction5
        (rho_ * v * Y5_2 - rhoD5_p * Y5_2 - omega * nu(5) * MM
            (5)) / (rhoD5); ... %MassFraction5
        ];

end
function res = mat4bc(ya, yb, lambda) % boundary conditions
    va = ya(1);
    U1a = ya(2);
```

```
    TO0a = ya(4);

    Y1_1a = ya(6);
    Y2_1a = ya(8);
    Y3_1a = ya(10);
    Y4_1a = ya(12);
    Y5_1a = ya(14);

    vb = yb(1);
    U1b = yb(2);
    TO0b = yb(4);

    Y1_1b = yb(6);
    Y2_1b = yb(8);
    Y3_1b = yb(10);
    Y4_1b = yb(12);
    Y5_1b = yb(14);
    res = [va - v0; ... % v(0) = v0
        vb - vl; ... % v(L) = vl
        U1a - 0; ... % U(0) = 0
        U1b - 0; ... % U(L) = 0
        TO0a - TF0; ... % T(0) = TF0
        TO0b - TO0; ... %T(L) = TO0
        Y1_1a - YLeft(1); ... %
        Y1_1b - YRight(1); ... %
        Y2_1a - YLeft(2); ... %
        Y2_1b - YRight(2); ... %
        Y3_1a - YLeft(3); ... %
        Y3_1b - YRight(3); ... %
        Y4_1a - YLeft(4); ... %
        Y4_1b - YRight(4); ... %
        Y5_1a - YLeft(5); ... %
        Y5_1b - YRight(5); ... %
        ];


end
function yinit = mat4init(x) % initial guess function.
    yinit = [0.0; ... %v
        0.0; ... %U1
        0.0; ... %U2
        300; ... % TO0
        0.0; ... % T2
        1.0; ... % Y1_1
        0.0; ... % Y1_2
        0.0; ... % Y2_1
```

```
            0.0; ... % Y2_2
            0.0; ... % Y3_1
            0.0; ... % Y3_2
            0.0; ... % Y4_1
            0.0; ... % Y4_2
            0.0; ... % Y5_1
            0.0; ... % Y5_2
            ];
    end
%% Definition of helper functions for density and transport
   parameters
    function MW = getAverageMolecularWeight(Yk)
        mult = 0;
        for c = 1:5
            mult = mult + Yk(c) / MM(c);
        end
        MW = 1.0 / mult;
    end

    function density = rho(T, Yk)
        MW = getAverageMolecularWeight(Yk);
        density = p0 * MW / (R * T); % kg / m^3
    end

    function dRho_dy = drhody(T, dTdy, Yk, dYkdy)
        dRho_dy = -p0 * dTdy / (R * T^2 * (Yk(1) / MM(1) + Yk(2) /
            MM(2) + Yk(3) / MM(3) + Yk(4) / MM(4) + Yk(5) / MM(5))
           ) - ...
            p0 * (dYkdy(1) / MM(1) + dYkdy(2) / MM(2) + dYkdy(3) /
                MM(3) + dYkdy(4) / MM(4) + dYkdy(5) / MM(5)) ...
            / (R * T * (Yk(1) / MM(1) + Yk(2) / MM(2) + Yk(3) / MM
               (3) + Yk(4) / MM(4) + Yk(5) / MM(5))^2);
    end

    function viscosity = mu(T)
        S = 110.4;
        viscosity = viscosity0 * (T / Tref)^(3 / 2) * (Tref + S) /
            (T + S);
    end

    function dViscosity_dy = dmu_dy(T, dTdy)
        mu0 = viscosity0;
        S = 110.4;
        dViscosity_dT = 0.3e1 / 0.2e1 * mu0 * sqrt(T/Tref) * (Tref
            + S) / (T + S) / Tref - ...
            mu0 * (T / Tref)^(0.3e1 / 0.2e1) * (Tref + S) / (T + S
```

```
            )^2;
        dViscosity_dy = dViscosity_dT * dTdy;
    end

    function reactionRate = getReactionRate(T, Yk)
        Yf = Yk(1);
        Yo = Yk(2);
        rho_ = rho(T, Yk);
        Ta = getActivationTemperature(T, Yk);
        reactionRate = B * exp(-Ta./T) .* (rho_ .* Yf ./ MM(1)) .*
            (rho_ .* Yo ./ MM(2));
    end

    function heatRelease = getHeatRelease(Yk)
        if variableKineticParameters
            phi = GetPhi(Yk(1), Yk(2));
            if phi > 1.5
                phi = 1.5;
            end
            alpha = 0.21;
            if phi > 1
                heatRelease = (1.0 - alpha * (phi - 1)) *
                   massHeatRelease;
            else
                heatRelease = massHeatRelease;
            end
        else
            heatRelease = massHeatRelease;
        end
    end

    function activationTemperature = getActivationTemperature(T,
       Yk)
        if variableKineticParameters
            phi = GetPhi(Yk(1), Yk(2));
            if (phi >= 1.07)
                activationTemperature = (1 + 4.443 * (phi - 1.07)
                   ^2) * Ta0;
            elseif (phi <= 1.07 && phi > 0.64)
                activationTemperature = Ta0;
            else
                activationTemperature = (1 + 8.25 * (phi - 0.64)
                   ^2) * Ta0;
            end
        else
            activationTemperature = Ta0;
```

```
        end
    end

    function phi = GetPhi(YF, YO)
        phi = (s * YF0 / YO0) * (s * YF - YO + YO0) / (s * (YF0 -
           YF) + YO);
    end
end
```

Curriculum vitae

Vor- und Nachname

Persönliche Daten

Geburtsdatum:	-
Geburtsort:	-
Staatsangehörigkeit:	-

Schulbildung

Jahr - Jahr	-
Jahr - Jahr	-

Studium

Jahr - Jahr	-

Wissenschaftliche Tätigkeit

Jahr - Jahr	Wissenschaftlicher Mitarbeiter am Fachgebiet für Strömungsdynamik im Fachbereich Maschinenbau der TU Darmstadt, Promotion und Lehrtätigkeit