UNIVERSITY OF NOTTINGHAM, SCHOOL OF MATHEMATICAL SCIENCES

Numerical Modelling of Chemical Vapour Deposition Reactors

Nathan Sime

September 4, 2016

Abstract

In this thesis we study the chemical reactions and transport phenomena which occur in a microwave power assisted chemical vapour deposition (MPA-CVD) reactor which facilitates diamond growth. First we introduce a model of an underlying binary gas flow and its chemistry for a hydrogen gas mixture. This system is heated by incorporating a microwave frequency electric field, operating in a resonant mode in the CVD chamber. This heating facilitates the dissociation of hydrogen and the generation of a gas discharge plasma, a key component of carbon deposition in industrial diamond manufacture.

We then proceed to summarise the discontinuous Galerkin (DG) finite element discretisation of the standard hyperbolic and elliptic partial differential operators which typically occur in conservation laws of continuum models. Additionally, we summarise the non-stabilised discontinuous Galerkin formulation of the time harmonic Maxwell operator. These schemes are then used as the basis for the discretisation method employed for the numerical approximation of the MPA-CVD model equations.

The practical implementation of the resulting DG MPA-CVD model is an extremely challenging task, which is prone to human error. Thereby, we introduce a mathematical approach for the symbolic formulation and computation of the underlying finite element method, based on automatic code generation. We extend this idea further such that the DG finite element formulation is automatically computed following the user's specification of the convective and viscous flux terms of the underlying PDE system in this symbolic framework. We then devise a method for writing a library of automatically generated DG finite element formulations for a hierarchy of partial differential equations with automatic treatment of prescribed boundary conditions.

This toolbox for automatically computing DG finite element solutions is then applied to the DG MPA-CVD model. In particular, we consider reactor designs inspired by the AIXTRON and LIMHP reactors which are analysed extensively in the literature.

Acknowledgements

Primarily, I wish to thank my supervisor, Paul Houston, for his support, guidance, advice and encouragement throughout the past four years. I wish to further thank Andrew Cliffe for gently introducing me to the large task which lay ahead in the first year of my PhD. I would also like to thank Ian Friel for his insight into the rich science of industrial diamond manufacture.

I'd like to acknowledge Prof. John Billingham and Dr. Keith Hopkraft for helping me through musings on aspects of energy transport and plasma physics. I would further like to acknowledge Dr. John Brandon and Katharine Robertson whose feedback helped shape aspects of the code developed for this thesis.

Without the company of my fellow PhD students and the numerous post-doctoral researchers at the University of Nottingham, the coffee room would have been a lonely retreat. Special thanks are extended to Tom Wicks, Phillip Paine and Rosanna Cassidy for their continuing friendship.

I especially wish to extend my gratitude to my fellow students of the Scientific Computation group, Tom Bennison, Joe Collis and Scott Congreve. Their tolerance of my incessant questions regarding numerical mathematics shaped the foundation of this work. Furthermore, I would like to extend thanks to the newest member of the group, Jonathan Marsh, for lending his aid in the final weeks of writing this thesis.

Finally, I wish to thank my parents. Without their cultivation of scientific intrigue, financial support and life guidance, this thesis would not have been possible.

In fondest memory of Andrew Cliffe
"Gone on Ahead"

Contents

CHAPTER 1

Introduction

1.1 Synthetic Diamond

In 2012 the global industrial diamond output was estimated to be 4.52 billion carats, valued between $1.65 and $2.50 billion [112]. The first synthesis of artificially grown diamond is attributed to Howard Tracy Hall using the so-called high pressure high temperature (HPHT) process in 1954 [71]. This was achieved by building on work by Percy Bridgman in high pressure physics for which he was awarded the Nobel Prize in 1946. The largest of these first artificial diamonds measured 0.15 mm in diameter and were grown at pressures of 10 GPa and temperatures above 2000 °C. Further manufacturing methods have been developed since using chemical vapour deposition, explosive optics and ultrasound cavitation [129]. In this thesis we seek to model the chemical vapour deposition process of diamond manufacture.

The chemical vapour deposition (CVD) diamond market has seen strong growth since its introduction in the mid 1980s due to its properties of strength, durability, stiffness and thermal conductivity. Furthermore, research and development of CVD diamond manufacture has seen rapid expansion since its inception [129]. The advantages of the CVD diamond manufacture process owe to its rigidly controlled growth conditions. The remarkable optical, thermal, chemical and electronic properties of diamond, along with its extreme hardness and wear resistance, offer a material with great potential in scientific and engineering applications. Furthermore, diamond manufactured in chemical vapour deposition processes allows for the growth of the material to be tailored to specific applications.

The aim of this project is to formulate a mathematical model of the chemically reacting flows involved in the CVD diamond growth process, and to provide accurate and efficient numerical discretisation methods. Exploiting the results from these calculations will enable the growth of higher quality diamond products as a result of improved reactor design. The challenge lies in the amalgamation of modelling plasma physics, energy deposition, chemical reactions, and transport processes, as well as developing suitable computational methods. In this thesis, we present a model for the dissociation of molecular hydrogen in a CVD reactor, along with its discontinous Galerkin (DG) finite element method (FEM) approximation.

1.2 Synthetic Diamond Manufacture via Chemical Vapour Deposition

Industrial use of CVD diamond requires film purity, low defect content and a satisfactory growth rate. Employing a microwave plasma-assisted (MPA) resonant cavity system in the CVD reactor design facilitates the dissociation of required quantities of atomic hydrogen to meet these needs. A summary of the physical properties of diamond grown in CVD reactors, as well as an overview of the CVD growth process is given by Balmer et al. [13].

The first synthesis of diamond material using MPA-CVD was demonstrated in 1983 by Kamo et al. [88]. Employing a gas mixture of hydrogen and $\backsim$1% methane, the hydrogen component was dissociated via microwave discharge. Atomic hydrogen atoms stabilise the growth of diamond and discourage the growth of graphite. In a typical CVD manufacturing process, the diamond mounted on a substrate in the CVD reactor is heated to temperatures of 800 K to 1000 K and the gas is held at pressures between 15 torr to 300 torr. A typical MPA-CVD reactor microwave cavity design consists of a quartz window separating a vacuum chamber and an air filled cavity. The vacuum chamber contains the hydrogen methane mix in which the plasma is ignited and the diamond deposited. The shape and stability of the ignited plasma leads to optimum conditions for diamond growth. It is favourable that the plasma be flat against the substrate surface whilst being maintained for a period of hours to several days. A cross section of a typical MPA-CVD reactor design is show in Figure 1.1.

MPA-CVD reactor design can be optimised through trial and error based on an understanding of the underlying physical processes, although this is a time consuming

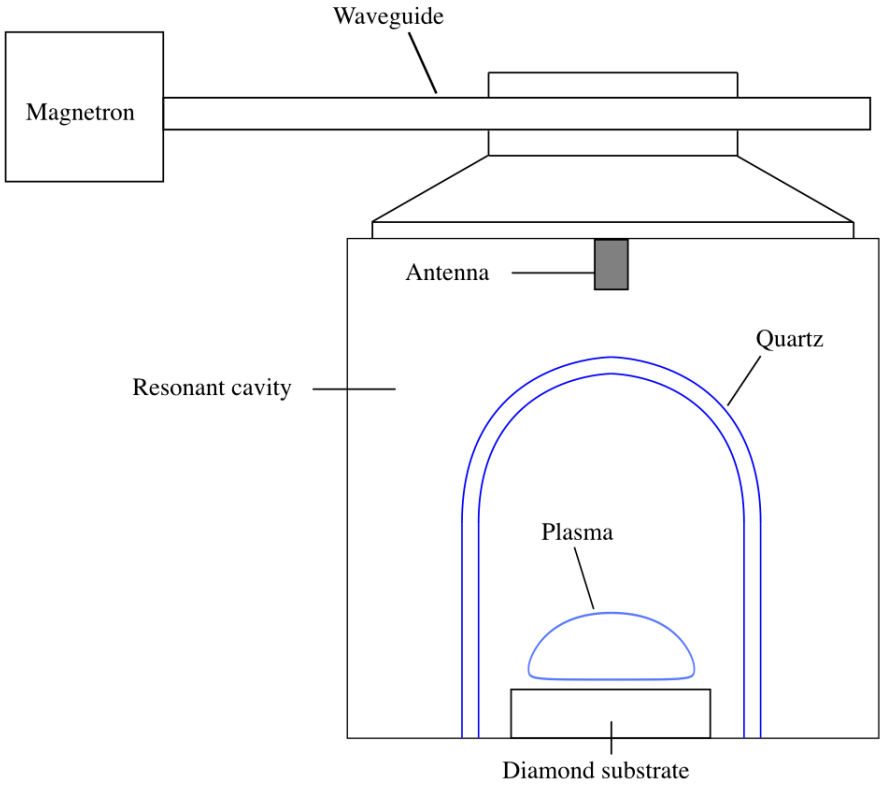


Figure 1.1: A cross section of a typical MPA-CVD reactor design.

and expensive process. In the case of complicated geometries, the shape of the electric field in the CVD reactor can be difficult to predict given its interaction with the plasma. Numerical solutions of CVD reactor models are therefore an essential requirement. Initial numerical experiments employing finite difference numerical schemes were performed in [53, 61, 136]; see [47, 123] for high performance computing experiments implementing the finite element method for multiphase gas flow occurring the CVD reactors.

In this thesis we aim to develop a fully self consistent numerical model of the processes which occur in the MPA-CVD diamond manufacture process. We seek to simultaneously find numerical approximations to the gas momentum transfer, heat transfer, electromagnetic field energy and plasma density. This will build on prior work of numerical models which only account for subsets of the physical and chemical processes occurring in a CVD reactor. Furthermore, the numerical methods employed should be efficient and robust to accommodate for the large range of operating conditions of CVD reactors. Whilst offering the power of modern numerical methods, we will also ensure its ease of use in a user-friendly environment. In the following sections, we will give an outline of each of these subjects.

1.3 Modelling MPA-CVD Reactors

In this section we give an outline of the descriptive models regarding the hydrogen dissociation chemistry in a CVD reactor based on the excellent review paper by Hassouni et al. [70]. For molecular hydrogen this includes accounting for:

- The collisions of electrons with molecular hydrogen resulting in excited rotational and vibrational state and hence the heating of the hydrogen gas [32, 68, 104, 105].
- Electronically excited states of molecular hydrogen, some of which result in the production of atomic hydrogen [31, 34].
- The resistive electron collision interaction with atomic hydrogen leading to the production of several excited state species including atomic and ionised hydrogen [33, 34].
- Collisions between electrons and atomic hydrogen resulting in excited atomic hydrogen states along with production of hydrogen ions [118, 125, 141].
- The reactions involving the heavy molecular and atomic hydrogen species also lead to production of their excited states and energy redistribution [104, 105], ionisation [116, 134], heated dissociation [68, 105] and reciprocal neutralisation [51, 101].

Hassouni et al. [70] summarise by stating that a fully descriptive model of hydrogen plasmas requires consideration of at least seven species along with their internal modes: H_2, H, H^+, H^-, H_2^+, H_3^+ and e^-. In order to reduce complexity of the model, the state-to-state kinetics of the internal modes are considered negligible. For example, the number density of electrically excited hydrogen molecules is several orders of magnitude smaller than its ground state in moderate pressure hydrogen plasma discharges [68]. With this in mind, a simplified model can be presented of moderate pressure hydrogen plasma discharges that accounts for collisional energy transfer between electrons and the heavy species in terms of ionisation and dissociation kinetics. The energy of the system is modelled via the thermodynamic temperature of the heavy species gas mixture and the temperature of the electrons measured from the electron energy distribution function. The simplified model also reduces the reactions and production of the seven species mentioned above to H_2, H, H^+, H_3^+ and e^-. Here, H^- is ignored due to molecular hydrogen discharges at moderate pressure being electropositive. Furthermore, the $H_2{}^+$ species is considered an instantaneous intermediate species which is converted to $H_3{}^+$. In this thesis, we consider a further simplified model where we only consider the dissociation of hydrogen and thereby the species H, H_2 and e^-.

Hassouni et al. [70] state the assumption, which will also be used in the model presented in this thesis, that the flow within the CVD reactor is subsonic, allowing for the mean system pressure to be treated as a system constant. When low flow rates are considered, the convective effects of mass transport are negligible and only diffusive transport need be considered. The particle flux of each species therefore obeys a continuity conservation law of the form

$$\frac{\partial \rho_i}{\partial t} + \nabla \cdot \mathbf{j}_i = r_i, \tag{1.3.1}$$

where ρ_i, $\mathbf{j}_i$ and r_i are the densities, diffusive fluxes and mass production rates of species i, respectively. The diffusive fluxes $\mathbf{j}_i$ are determined by Fick's law of diffusion due to inter-species particle concentration gradients. Furthermore, conservation of the energy density of each species provides a description of the temperature of the hydrogen gas and the temperature of the electrons whose energy source comes from the coupled microwave field. This pseudo-Soret effect is expressed as

$$\frac{\partial \left(\rho h\right)}{\partial t} + \nabla \cdot \mathbf{q} = Q, \tag{1.3.2}$$

where h, $\mathbf{q}$ and Q are the gas enthalpy, diffusive flux vector due to temperature gradients and external heating, respectively. Discussed later in this thesis, the model presented accounts for the case of high flow rates where the convective effects are not neglected.

The gas is heated via a coupled microwave frequency electric field, facilitating the dissociation of hydrogen. The ignited plasma in the gas mixture introduces a perturbation in the electrical permittivity of the gas mixture medium. Hassouni et al. [70] summarise two methods for modelling the electric field's absorbed power in the plasma as either a high frequency conducting medium or a dielectric.

Regarding the high frequency conducting medium model, solutions are sought for the electric field $\mathcal{E}$ and magnetic field $\mathcal{H}$ from the time dependent form of Maxwell's equations

$$\nabla \times \mathcal{E} = -\mu_0 \frac{\partial \mathcal{H}}{\partial t}, \tag{1.3.3}$$

$$\nabla \times \mathcal{H} = -\varepsilon_0 \frac{\partial \mathcal{E}}{\partial t} - q_\mathrm{e} n_\mathrm{e} \mathbf{u}_{\mathrm{e-HF}}. \tag{1.3.4}$$

Here, μ_0 and ε_0 are the permeability and permittivity of free space, respectively, q_e, n_e and $\mathbf{u}_{\mathrm{e-HF}}$ are the charge of the electron, electron number density and high frequency electron velocity, respectively; these combine to give the electron drift velocity $\mathbf{j}_\mathrm{e} =$

$-q_e n_e \mathbf{u}_{e-HF}$. In the context of electrically conducting plasmas, these solutions, $\mathcal{E}$ and $\mathcal{H}$, can be found in the frequency or time domain [60, 90, 122, 146], where examples specifically for a CVD reactor geometry are given in [135, 136]. A key ideal in the design of a CVD reactor is for its dimensions to support resonant modes of the contained electric field. The resonant frequencies supported by an empty CVD reactor cavity geometry can be calculated using frequency domain analysis [53–55, 126].

The time domain solutions to Maxwell's equations offer an advantage over the frequency domain analyses in the sense that they facilitate direct coupling with the plasma system. Furthermore, due to its suitability for numerical schemes such as finite differencing, finite volume and finite element methods, time domain analysis allows for easier numerical discretisation of complex geometries, which is typically the case for CVD reactors. Combined with rapid computational performance improvements between 1995 and 2005, time domain analysis became more widely popular, we refer to [53, 55, 59, 63, 69, 92, 136, 140, 145] for a comprehensive history of these developments.

The disadvantage of the time domain analyses of the high frequency model is the implementation of simultaneously solving several equations to determine the high frequency component of the electron drift velocity $\mathbf{j}_e$. The calculation of $\mathbf{j}_e$ depends on the collision cross section of the electron-heavy species momentum transfer and the electron energy distribution function [29, 30]. The dielectric model of the CVD reactor plasma formulates the high frequency electron drift velocity component of equation (1.3.4) as part of a plasma dielectric permittivity ε_p. In essence, equation (1.3.4) can be written as Ampère's law

$$\nabla \times \mathcal{H} = -\varepsilon_p \frac{\partial \mathcal{E}}{\partial t}. \tag{1.3.5}$$

The plasma parameters involved in the calculation of the plasma permittivity are dependent on the particle and energy density conservation of each constituent chemical species. Equation (1.3.5) offers the advantage of being well suited for the time-harmonic formulation of Maxwell's equations where $\mathcal{E}(\mathbf{x}, t) = \Re(\mathbf{E}(\mathbf{x})e^{j\omega t})$ and $\mathcal{H}(\mathbf{x}, t) = \Re(\mathbf{H}(\mathbf{x})e^{j\omega t})$ for electric field frequency ω, complex unit $j = \sqrt{-1}$ and complex phasors $\mathbf{E}$ and $\mathbf{H}$.

Although beyond the scope of this thesis, Hassouni et al. [70] also summarise the hydrogen-methane plasma models developed for diamond deposition. The underlying principles of the H_2/CH_4 model are very similar to the momentum, particle density and energy density conservation when coupled with a driven electric field which mod-

els the hydrogen plasma [67]. The introduction of methane to the model introduces much greater complexity due to the larger number of chemical species and chemical reactions. Furthermore, the ionisation properties and hence plasma parameters of the H_2/CH_4 mixture will be largely different from those incorporated in the hydrogen plasma model [50, 85, 119]. It should be noted that in diamond deposition reactors, which will only have peak ratios of CH_4/H_2 at $\backsim$10%, the difference in the plasma shape, position and optimal power deposited from the electric field compared to the hydrogen plasma is not significant [56, 57].

The series of articles by Füner et al. [53–55] and subsequently the work by Hassouni et al. [69] and Hagelaar et al. [63] give a chronological account of the development of numerical models of CVD reactors employing the finite difference method. Accounting only for diffusive transport in a hydrogen gas mixture, a numerical model is solved with the emphasis being on the measurement of the microwave power deposition in the plasma discharge, as well as the composition of the plasma at the diamond substrate surface. Modelling hydrogen dissociation allows for the simulation of the configuration of the shape, temperature and position of the plasma in the CVD reactor. Using the plasma shape calculated from this result, a further model is presented for the mixture of hydrogen and methane in one dimension along the axis of the ignited plasma. The results of the hydrogen methane model were then used to analyse transport and wave phenomena in the plasma, and therefore the optimisation of diamond growth processes at the substrate surface.

1.4 Numerical Simulation and Reactor Design

The use of numerical models for the optimisation of reactor design is reviewed by Silva et al. [127]. Simulations presented in [127] show that the concentration of atomic hydrogen resulting from its dissociation from molecular hydrogen is sufficient to encourage deposition of high purity diamond films whilst maintaining large growth rates in various reactor geometries. Most importantly for MPA-CVD reactors, the shape and volume of the plasma produced in the MPA cavity of the CVD reactor depends greatly on the careful design of the microwave system. The capacity to precisely account for the coupling between the electric field and the plasma and therefore the perturbation in the electric field is a key component in the reactor's geometric optimisation. For example, this perturbation can lead to large portions of the input power being reflected resulting in detrimental heating and damage of the reactor walls or quartz window.

Improving reactor design by exploiting numerical reactor modelling usually involves the optimisation of a so-called quality factor Q_f. The quality factor function is dependent on the CVD reactor operating parameters, such as geometric dimension, operating pressure and antenna/waveguide configuration. Computing simulations on a discrete mesh of cells representing a CVD reactor geometry, Füner et al. [55] demonstrate the optimisation of their ellipsoidal reactor design based on the metric of absorbed microwave energy in the plasma measured in terms of the electric field magnitude, $|\mathcal{E}|$, whereby Q_f is defined by

$$Q_f = \frac{\sum\limits_{\text{plasma cells}} |\mathcal{E}|^2}{\sum\limits_{\text{remaining cells}} |\mathcal{E}|^2}. \tag{1.4.1}$$

This quality factor metric is further implemented in recent numerical reactor design methods, cf. An et al. [6].

The use of numerical modelling for reactor design allows for novel geometric shapes of CVD reactors to be easily tested. We note that the choice of the quality factor function depends on a desired quantity of interest, and that we are not restricted to (1.4.1). For example, incorporating geometries designed to reflect the enclosed coupled resonant electric field maxima to a single focal point as in [55, 98]. Recent simulations used for CVD reactor designs, optimising geometries of reflective surfaces, are presented in [6, 97, 130, 131].

We emphasise that in these models only diffusive effects are considered, or the bulk gas flow of hydrogen is modelled as a velocity potential, cf. Koldanov et al. [92]. Our aim here is to model the conservation of momentum of the atomic and molecular hydrogen gas mixture and therein its diffusive and particularly its convective transport phenomena. Due to the convective terms arising in the model presented in this thesis, we will employ the discontinuous Galerkin finite element method.

1.5 Discontinuous Galerkin Finite Element Methods

1.5.1 Hyperbolic and Elliptic Operators

The finite element method seeks to approximate the weak formulation of a given partial differential equation (PDE) boundary value problem, based on employing piecewise polynomials. Indeed, the computational domain is subdivided into elements and

the underlying solution is approximated on each element by a polynomial of a given order ℓ. In 'traditional' continuous Galerkin (CG) finite element methods, the assumption is made that the numerical solution approximation is continuous across the boundaries of the elements. To enforce this continuity condition, the approximating polynomial of an element on the mesh will share degrees of freedom with its neighbours. Subject to enforcing appropriate boundary conditions, the CG finite element approximation closely follows the variational formulation of the underlying PDE by replacing trial and test functions by piecewise polynomials. In the case of linear problems, this discrete system leads to the matrix problem $A\mathbf{x} = \mathbf{b}$. The book by Babuška and Strouboulis [10] details the early history of the development of finite element methods.

Standard CG finite element approximations of convection dominated problems exhibit non-physical oscillatory solutions leading to issues of poor numerical stability. As an example of such numerical instability, on the interval $(0, 1)$ consider the equation

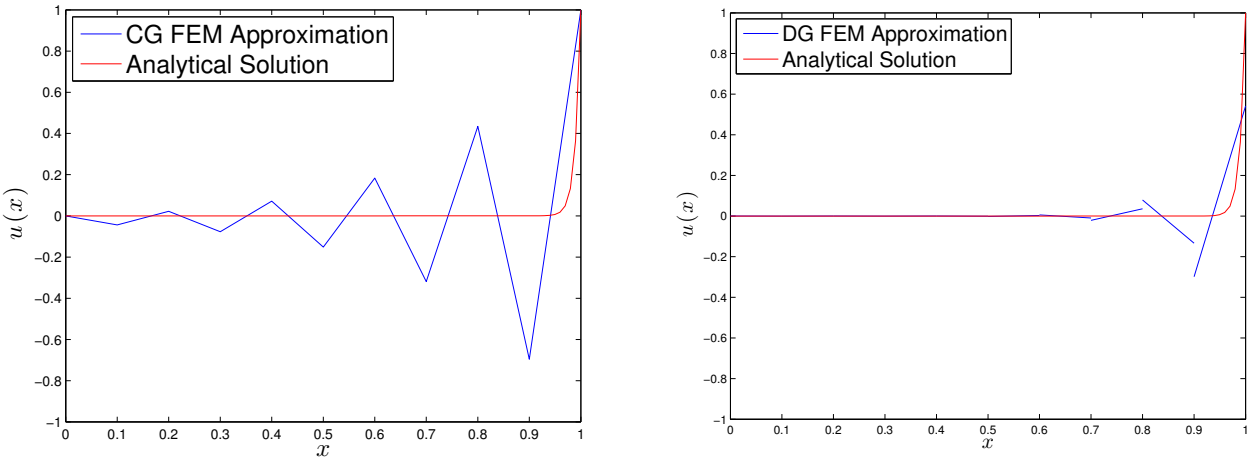
$$-\epsilon \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} + b\frac{\mathrm{d}u}{\mathrm{d}x} = 0, \quad u(0) = 0, \quad u(1) = 1, \tag{1.5.1}$$

which has the analytical solution

$$u(x) = \frac{1 - e^{\frac{b}{\epsilon}x}}{1 - e^{\frac{b}{\epsilon}}}. \tag{1.5.2}$$

Here, the quantity b/ϵ is the Péclet number; for a large Péclet number ($b \gg \epsilon$) equation (1.5.1) becomes convection dominated, leading to the formation of a boundary layer, where u rapidly changes close to $x = 1$. For $\epsilon = 0.01$ and $b = 1$, the analytical solution and CG finite element approximation with 10 elements are shown in Figure 1.2a. Stabilisation schemes for the standard CG finite element method have been proposed to alleviate this problem such as 'upwinding' [23] and the residual free bubble method [22, 27].

When modelling the transport phenomena found in CVD reactor gas flows, the resulting equations can be convection dominated in the case of high input gas flow rates. Thereby, numerical simulations require the approximation of systems of highly non-linear equations stemming from the exploitation of very fine resolution computational meshes. Brooks and Hughes [23] introduce the challenge of finding numerically stable solutions to the Navier-Stokes equations. With application to the Navier-Stokes equations, examples are given for including 'artificial viscosity' in the direction of convection in the form of an added diffusion term. This streamline up-wind/Petrov-Galerkin method introduces stability into the finite element solution of convection dominated



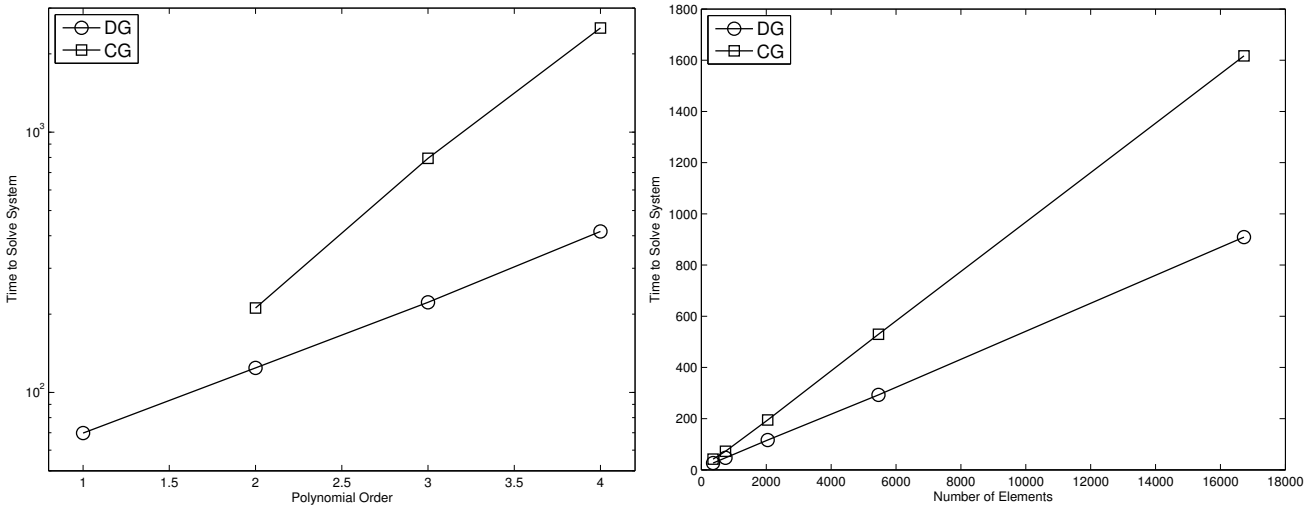
(a) CG **(b)** DG

Figure 1.2: Finite element solutions to equation (1.5.2). This presents an example of non-physical numerical oscillation error present in continuous Galerkin finite element approximations of convection dominated problems.

problems. Brooks and Hughes give examples ranging from a convection-diffusion equation to vortex shedding from a circular cylinder. A summary of stabilisation methods proposed for CG methods and their comparison with discontinuous methods is given by Cangiani et al. [28]. Implementing this notion of stabilised CG finite element methods applied to numerical approximations of chemically reacting flows, along with some strategies for re-meshing and mesh refinement, Braack and Richter [20] present results implementing the SUPG method developed by Brooks and Hughes [23].

Currently an area of great interest in computational modelling, discontinuous Galerkin (DG) finite element methods offer novel schemes which attempt to address the numerical stability problem. Further benefits of the DG finite element method arise from the richer space of functions in which the numerical approximation is sought (e.g. permitting discontinuities across element interfaces) and the consistent DG finite element formulation for PDE operators, as will be shown in Chapter 3.

Stemming from the concepts of weakly enforcing Dirichlet boundary conditions on the exterior of a computational domain [109], the DG method weakly enforces continuity of the solution variable across interior element faces. As a consequence, the 'upwinding' employed artificially by other methods is in fact implicitly a component of the DG formulation. Its application to elliptic PDEs is summarised by Arnold et al. [8] reviewing earlier proposals for non-conforming finite element methods. For exam-



(a) Varying polynomial order, ℓ, with a fixed geometry and mesh element density.

(b) Varying mesh element density of a fixed geometry and fixed polynomial order, $\ell = 2$.

Figure 1.3: Comparison of run times to complete a simulation of a CVD reactor geometry for a more extreme parameter set using a CG and DG finite element method on the same mesh. These results were generated from a preliminary model which does not include the coupling of the electromagnetic field and plasma model.

ple, some of the many DG finite element methods include: symmetric interior penalty methods [7, 11, 49, 143], non-symmetric interior penalty methods [77, 120], incomplete interior penalty methods [45, 48, 132] and the Baumann-Oden method [15]. Analysis of the DG finite element method applied to nonlinear second order PDEs has been undertaken in [26, 62, 81], for example. The application of DG methods to hyperbolic problems and its *a priori* analysis has been analysed in [17, 73, 75].

Numerical experiments show the advantages of the increased stability and robustness of the DG method compared with CG methods, however at a cost of greater computational expense. This arises due to the DG scheme allowing for discontinuities across boundaries of elements, which implies that degrees of freedom are not shared between elements as with CG schemes, but rather more degrees of freedom are required to enforce numerical flux restriction on the facets between neighbouring elements. A DG finite element approximation to the solution of equation (1.5.1) is given in Figure 1.2b for $\epsilon = 0.01$ and $b = 1$ demonstrating the advantages of the DG numerical scheme in this convection dominated system.

Examples of the DG method applied to the Stokes [39], Euler [64], compressible Navier-Stokes [66], incompressible Navier-Stokes [38, 40–42] and incompressible magnetohydrodanamics [82] equations give a comprehensive explanation of the formulation of DG methods for fluid flow problems along with their *a priori* numerical analysis. By fixing the element polynomial degree of the approximating polynomial, optimal convergence rates can be analytically derived and experimentally shown by varying the element size, so-called h-refinement. Furthermore, due to DG methods permitting basis functions discontinuous across element interfaces, varying the degree of the approximating polynomial across elements in the DG scheme is handled by the method in a simple manner. This leads to the so-called p- and hp-refinement methods being easily implemented in a DG scheme, which offer exponential convergence rates with an increasing number of degrees of freedom [74, 76]. Naturally this leads to adaptive refinement based on *a posteriori* dual weighted residual error estimation, for which DG methods are well suited. Examples of this approach for nonlinear hyperbolic conservation equations is given in [65].

The details of the CVD reactor model and the computation of its numerical approximation will be discussed later in this thesis; however, to highlight the benefit of using DG methods in this project, a comparison between using a CG and DG method for the CVD reactor model was devised. We use a reduced reactor model which only solves for hydrogen gas momentum, mass and enthalpy balance, but not the electric field and plasma model. Selecting a CVD reactor geometry and a set of parameters which would describe a typical CVD reactor state for growing diamond, the time elapsed to compute the solution is shown in Figure 1.3. The primary reason for the improved performance of the DG scheme is that the stability and robustness it offers leads to a reduction in the number of continuation steps required to compute the solution of the system state for the given parameter set. Another clear benefit of the DG scheme is from the richer space of functions in which the solution is sought. The DG Taylor Hood elements employed to solve the gas flow momentum balance equations permit piecewise constant polynomial approximation of the pressure and piecewise linear polynomials for each component of the velocity solution. This reduces the number of degrees of freedom in the system compared with the CG FEM Taylor Hood scheme, although at a cost of greater approximation error due to the lower polynomial order.

1.5.2 Time-Harmonic Maxwell Operator

The time-harmonic Maxwell equations present a series of challenges in their approximation using finite element methods. In a naïve approach, their reformulation to the Helmholtz equation through the divergence free field constraint of Gauss' Law, allows for the standard variational formulation of the Laplace operator using CG finite element spaces [83, 84]. Not only does this require that the discretisation of the underlying geometry satisfy Nyquist's theorem, but in the case of a non-convex domain, the solution to the time-harmonic Maxwell equations can be singular. Here, the use of standard finite element methods will lead to the discrete solution erroneously converging to a function which is not the solution to Maxwell's equations. Accounting for this, the tangentially continuous edge elements of Nédélec with $H(\text{curl})$ conforming basis were developed in [107, 108], see also [86, 106].

For a review of the development of DG methods for the time-harmonic form of Maxwell's equations, we refer to the series of articles [79, 80, 114, 115]. In the discretisation process, the flux formulation weakly enforces continuity of the inter-element tangential flux. Furthermore, the divergence free field condition of Gauss' law is enforced by a Lagrange multiplier. Buffa et al. present the application of DG methods to the Maxwell eigenvalue problem in [24, 25].

Eigenvalue problems include those required for computing electromagnetic resonant frequency estimates of resonant cavities. For example, consider the eigenvalues λ of the Maxwell operator acting on the vector eigenfunction $\mathbf{u}$ in a computational domain Ω with boundary $\partial\Omega$ together with prescribed homogeneous boundary conditions

$$\nabla \times \nabla \times \mathbf{u} = \lambda \mathbf{u} \qquad \text{in } \Omega, \tag{1.5.3}$$

$$\mathbf{n} \times \mathbf{u} = \mathbf{0} \qquad \text{on } \partial\Omega. \tag{1.5.4}$$

Given a finite dimensional solution space $\mathbf{V}_{h,\ell}$, the DG variational formulation of (1.5.3) is to find eigen pairs $(\mathbf{0} \neq \mathbf{u}_h, \lambda_h) \in \mathbf{V}_{h,\ell} \times \mathbb{C}$ such that

$$a_h(\mathbf{u}_h, \mathbf{v}_h) = \lambda_h(\mathbf{u}_h, \mathbf{v}_h) \quad \forall \mathbf{v}_h \in \mathbf{V}_{h,\ell}. \tag{1.5.5}$$

Here, $a_h(\cdot,\cdot)$ is the DG sesquilinear variational formulation of the Maxwell operator and $(\mathbf{u}, \mathbf{v}) = \int_\Omega \mathbf{u} \cdot \overline{\mathbf{v}}\, \mathrm{d}\mathbf{x}$ denotes the usual $L_2(\Omega)$ inner product. The DG finite element matrix formulation of the eigen problem is of the form $A_h\mathbf{U} = \lambda_h M\mathbf{U}$ where A_h is the discrete matrix formulation of the bilinear operator $a_h(\cdot,\cdot)$, $\mathbf{U}$ is the DG finite element solution vector and M is the mass matrix. Numerical solutions to this problem can be

calculated using eigenvalue and eigenvector computational libraries such as ARPACK [95].

1.6 Automatic Solution of Partial Differential Equations

Writing code to implement finite element methods is a challenging task, however several libraries exist which facilitate the solution process. At the low level, libraries such as FreeFem++ [72], deal.II [14] and OpenFOAM [142] provide data structures and functions encapsulating aspects of the finite element computation. These libraries still require in depth knowledge of the application of finite element methods and experience with the languages and their paradigms in which they are each written. Using such packages to formulate and solve the systems of equations associated with large multi-physics models in several subdomains of a parent geometry remains a very technical and time consuming task.

One approach which aims to bring a higher level of language syntax to the computation of finite element solutions is illustrated by the Unified Form Language (UFL) of the FEniCS project. Exploiting Python for its dynamic and weakly typed nature, along with automated memory management and synergy of object-orientation, procedural and functional constructs, the UFL syntax allows for user friendly and expressive specification of finite element problems. Examples of a similar syntax to the UFL are given in Table 1.1. By creating a layer of abstraction between the model and the numerical solution, the UFL allows a user to specify their problem with little requirement to be proficient with programming. Details of the implementation of the UFL and the FEniCS Form Compiler (FFC) which automatically generates C++ code to solve the finite element problem are discussed along with design choices based on the works of symbolic and automatic algebraic calculus collected by Bischof et al. [19]. A collection of numerical problems in various fields of mathematics, including fluid flow problems and the results of their calculation in the FEniCS suite is given by Logg et al. [103].

The practical use of packages such as FEniCS is the encapsulation of the complexity of writing finite element code. An overview of how such packages are created, with specific consideration to DOLFIN, part of the FEniCS project is given by Logg and Wells [102]. Futhermore, a discussion of the aesthetic links between the syntax of high level code and finite element mathematics along with its application in the FEniCS project is given by Alanæs et al. [2]. Examples of this high level language used for numerical

FE Operation	Code Syntax
$\int_\Omega uv \, \mathrm{d}\mathbf{x}$	`u*v*dx`
$\int_{\partial\Omega} gvds$	`g*v*ds`
∇u	`grad(u)`
$\nabla \cdot \mathbf{u}$	`div(u)`
$\mathbf{u} \cdot \mathbf{v}$	`dot(u, v)`
$[\![p]\!] \cdot \{\!\{\mathbf{q}\}\!\}$	`dot(jump(p), avg(q))`

Table 1.1: Examples of high level code syntax following aesthetically from the finite element method nomenclature.

analysis of the DG finite element formulation of the Poisson, advection-diffusion and Stoke's equations are presented in Ølgaard et al. [110].

Even in this user friendly setting, large systems of nonlinear PDEs with parameters which may consist of power series or functionals of solutions variables, writing the DG formulation can be a difficult task and prone to human error. Furthermore, the specification of a DG finite element formulation in the UFL can be somewhat verbose given that there are not any utility methods for handling elliptic or hyperbolic operators. A computational tool whose syntax resembles that of the UFL as a basis for automatic computation of large DG finite element problems, whilst ensuring ease of code generated is a further challenge addressed in this thesis with the development of the software suite AptoPy. A brief summary of AptoPy's structure is given in Section 1.7.3 and its syntax and operation in the subsequent Section 1.7.6. The development and operation of the AptoPy package is discussed in greater detail in Chapter 5.

1.7 AptoFEM and AptoPy

The underlying finite element software package used throughout this thesis is AptoFEM [1]. The AptoFEM project is lead by Paul Houston and has had several contributors. A major contribution of this thesis is the symbolic algebra front end to AptoFEM named AptoPy. In this section we introduce the basics of the implementation of a simple finite element problem, how AptoFEM and AptoPy are used to solve this problem, and then

the syntax and structure of the code required to compute the solution.

1.7.1 The Finite Element Method

The finite element method is constructed based on approximating the solution to the weak formulation of a PDE [87]. With this in mind consider the following variational formulation: find $u \in V$ such that

$$a(u, v) = l(v) \quad \forall v \in V. \tag{1.7.1}$$

Here, V is a Hilbert space, $a(\cdot, \cdot)$ is a bilinear form on $V \times V$ and $l(\cdot)$ is a linear functional on V. Here (1.7.1) may be considered as the weak formulation of a linear PDE.

Employing a Galerkin scheme, the finite element approximation $u_h \in V_h$ is sought such that

$$a(u_h, v_h) = l(v_h) \quad \forall v_h \in V_h, \tag{1.7.2}$$

where V_h is a finite dimensional subspace of V, composed of the span of continuous piecewise polynomial basis functions of fixed order defined on a given computational mesh. Employing a suitable basis for V_h, i.e., writing $V_h = \text{span}_{i=1,\dots,N}\{\phi_i\}$, where $N = \dim(V_h)$, the finite element method (1.7.2) may be re-written in the following equivalent matrix form: find $\mathbf{u} = (U_1, \dots, U_N)^\top$ such that

$$A\mathbf{u} = \mathbf{f}. \tag{1.7.3}$$

Here, $A_{ij} = a(\phi_j, \phi_i)$ and $\mathbf{f}_i = l(\phi_i)$. In principle, assuming that A is invertible, the solution vector $\mathbf{u}$ may be computed.

For a nonlinear PDE, a semilinear weak formulation can be derived; in this setting we seek $u \in V$ such that

$$\mathcal{N}(u; v) = 0 \quad \forall v \in V. \tag{1.7.4}$$

Here, $\mathcal{N}(u; v)$ is a semilinear form, which is nonlinear in u but linear in v. In this case, the finite element solution $u_h = \sum_{i=1}^{N} U_i \phi_i$ is determined as the solution of the system of nonlinear equations defined by

$$\mathcal{N}(u_h; v_h) = 0 \quad \forall v_h \in V_h. \tag{1.7.5}$$

To compute the solution we may employ Newton's method [20, 89]; thereby, we have the iteration

$$u_h^{n+1} = u_h^n + d_h^n, \tag{1.7.6}$$

where d_h^n is the update of u_h^n defined by: find $d_h^n \in V_h$ such that

$$\mathcal{N}'\left[d_h^n\right]\left(u_h^n; v_h\right) = -\mathcal{N}\left(u_h^n; v_h\right) \quad \forall v_h \in V_h. \tag{1.7.7}$$

Here, $\mathcal{N}'\left[w\right]\left(u; v\right)$ is the Gâteaux derivative of $\mathcal{N}\left(u; v\right)$ in the direction of w, i.e.,

$$\mathcal{N}'\left[w\right]\left(u; v\right) := \lim_{\tau \to 0} \frac{\mathcal{N}\left(u + \tau w; v\right) - \mathcal{N}\left(u; v\right)}{\tau}. \tag{1.7.8}$$

1.7.2 AptoFEM

AptoFEM serves as a practical tool for the automated solution of PDEs using the finite element method. Data structures and functions are made available to the user allowing them to specify the finite element form of the weak formulation. The AptoFEM kernel then manages and handles the computation of the solution of the linear equation (1.7.3) or the nonlinear equation (1.7.5), provided that the user specifies the form of the residual vector $\mathcal{N}\left(u_h; v_h\right)$ and the Jacobi matrix $\mathcal{N}'\left[u_h\right]\left(d_h; v_h\right)$. AptoFEM further provides interfaces to external linear algebra packages such as MUMPS [4], PETSc [12] and ARPACK [95].

Although AptoFEM is a powerful tool, in the case of increasingly large coupled systems of PDEs the user specification of the finite element form becomes evermore prone to human error. The evaluation of the Jacobi matrix can also be a somewhat laborious task, inviting the possibility of human error compounded by the fact that the syntax of the code does not follow aesthetically from the mathematics. A proposed remedy to this is to generate the finite element form code automatically with computational symbolic algebra as detailed by Cliffe and Tavener [37].

1.7.3 AptoPy

A major contribution of this thesis is the development of the AptoPy package. This provides a user friendly front end to AptoFEM which automatically formulates the required Jacobi matrix, enforces boundary conditions for a given domain, and manages the execution of the solution process. Whilst planning a front end for AptoFEM, a series of key requirements were drafted:

- Be concise, with as little verbosity as possible.
- Feel familiar to users with little programming experience.
- Follow the form of the mathematics as closely as possible.

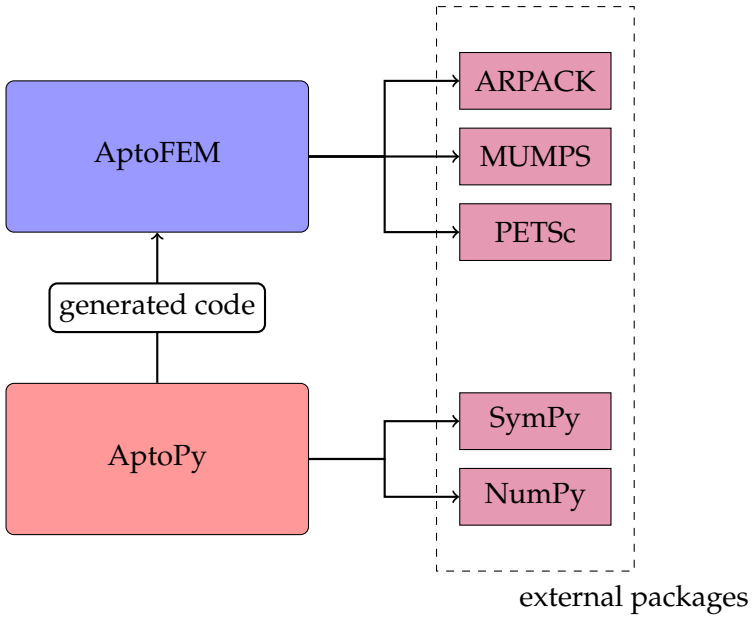


Figure 1.4: Relationship between AptoPy and AptoFEM and the implementation of their respective library dependencies.

- Automate the entire process between specification of a problem and AptoFEM computing a solution.

The preliminary objective was to choose a language to allow a user to write their finite element problem. Satisfying the requirement of familiarity for those with little programming experience, and the availability of the open source symbolic algebra package SymPy [133], the language chosen was Python. A simplified overview of the synergy of AptoPy, AptoFEM and their dependencies is given in Figure 1.4.

Pending adequate testing and review, it is hoped that the front end Python layer, named AptoPy, will allow AptoFEM to be used as more than a FEM code for research, but a tool for those who have little experience with numerics wanting to solve differential equations in complex geometries.

1.7.4 The 1D Poisson Equation and Weak Formulation

To provide some insight into the AptoPy package, in this section we provide a brief outline of the syntax employed to solve a PDE. The architecture and design of AptoPy, as well as a performance and validation review, will be further studied in detail in Chapter 5. As an example, here we consider the 1D Poisson equation

$$-u''(x) = f(x), \tag{1.7.9}$$

for which a solution to the unknown function u is sought in the interval $x \in [a, b]$, subject to the boundary conditions $u(a) = 0$ and $u(b) = 0$. The weak formulation is found by multiplying the above equation by a test function v and integrating by parts over the interval $x \in [a, b]$; thereby, we have

$$\int_a^b u'(x)v'(x)\,\mathrm{d}x - u'(x)v(x)\big|_{x=a}^{x=b} = \int_a^b f(x)v(x)\,\mathrm{d}x. \tag{1.7.10}$$

This operation requires that the functions u and v be members of the Sobolev space $H^1(a, b)$. Here, for the multi-index

$$\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}^d, \quad |\alpha| = \sum_{j=1}^{d} \alpha_j, \tag{1.7.11}$$

the Sobolev space $H^m(\Omega)$, $\Omega \subset \mathbb{R}^d$, $d \geq 1$, is defined with respect to the weak derivative operator of order D^α,

$$D^\alpha := \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}}, \tag{1.7.12}$$

such that

$$H^m(\Omega) := \{u \in L_2(\Omega) \ : \ D^\alpha u \in L_2(\Omega),\ |\alpha| \leq m\}\,. \tag{1.7.13}$$

The choice of test function v is arbitrary. Since there is no information available for $u'(a)$ and $u'(b)$, the test function should be chosen such that $v(a) = 0$ and $v(b) = 0$. Defining the space

$$H_0^1(a, b) := \left\{v \in H^1(a, b) : v(a) = 0,\ v(b) = 0\right\}, \tag{1.7.14}$$

we arrive at the weak formulation for the specific boundary value problem as follows: find $u \in V$ such that

$$\int_a^b u'(x)v'(x)\,\mathrm{d}x = \int_a^b f(x)v(x)\,\mathrm{d}x \tag{1.7.15}$$

for all $v \in V$, where $V = H_0^1(a, b)$.

1.7.5 Discretisation with Finite Elements

In order to discretise (1.7.15), the interval $x \in [a, b]$ is subdivided into individual 'elements' of length h. The solution space V is then replaced by a finite-dimensional subspace $V_h \subset V$ which consists of continuous piecewise polynomials of a fixed degree ℓ associated with this subdivision. With this notation, we define the following approximation: find $u_h \in V_h$ such that

$$\int_a^b u_h'(x)v_h'(x)\,\mathrm{d}x = \int_a^b f(x)v_h(x)\,\mathrm{d}x \tag{1.7.16}$$

for all $v_h \in V_h$. This is the the so-called finite element formulation of (1.7.9).

Writing $N(h) = \dim V_h$ to denote the dimension of the discrete solution space V_h, we let $V_h = \text{span}\left\{\phi_1, \ldots, \phi_{N(h)}\right\}$ for linearly independent basis functions ϕ_i, $i = 1, \ldots, N(h)$. Thereby, we may write u_h in the following form

$$u_h(x) = \sum_{j=1}^{N(h)} U_j \phi_j(x), \tag{1.7.17}$$

where the coefficients U_j must be determined computationally. Equation (1.7.16) can therefore be written in the following manner: find $U = (U_1, \ldots, U_{N(h)})^\top$ such that

$$\sum_{j=1}^{N(h)} U_j \int_a^b \phi_j' \phi_i' \, \mathrm{d}x = \int_a^b f \phi_i \, \mathrm{d}x, \tag{1.7.18}$$

for $i = 1, \ldots, N(h)$. This is a system of $N(h)$ linear equations which can be expressed as a matrix problem, with matrix $A_{ij} = \int_a^b \phi_j' \phi_i' \, \mathrm{d}x$ of size $N(h) \times N(h)$, and vector $b_i = \int_a^b f \phi_i \, \mathrm{d}x$ yielding the equation,

$$AU = b. \tag{1.7.19}$$

Solving equation (1.7.19) for U determines the finite element solution u_h.

1.7.6 Calculating the Solution

Using AptoPy, the solution vector U can be calculated after establishing the boundary value problem given in equation (1.7.16). In this case, we choose $a = 0$, $b = 1$ and $f = 1$. Initially we import the AptoPy library.

```
from AptoPy import *
```

The discretised interval $x \in [0, 1]$ with 15 elements can then be instantiated as a `Mesh` object, and its boundary, domain integration elements and spatial variable can be collected.

```
mesh = DiscreteInterval(15, 0.0, 1.0)
dS = mesh.boundary()
dx = mesh.domain()
x = mesh.space_vars()
```

The finite element function space V_h is instantiated based on this mesh, with default polynomial order $\ell = 1$. The Dirichlet boundary conditions of $u(a) = 0$ and $u(b) = 0$ are strongly enforced on this function space. Furthermore, the trial function u_h and the test function v_h associated with this function space can then be defined.

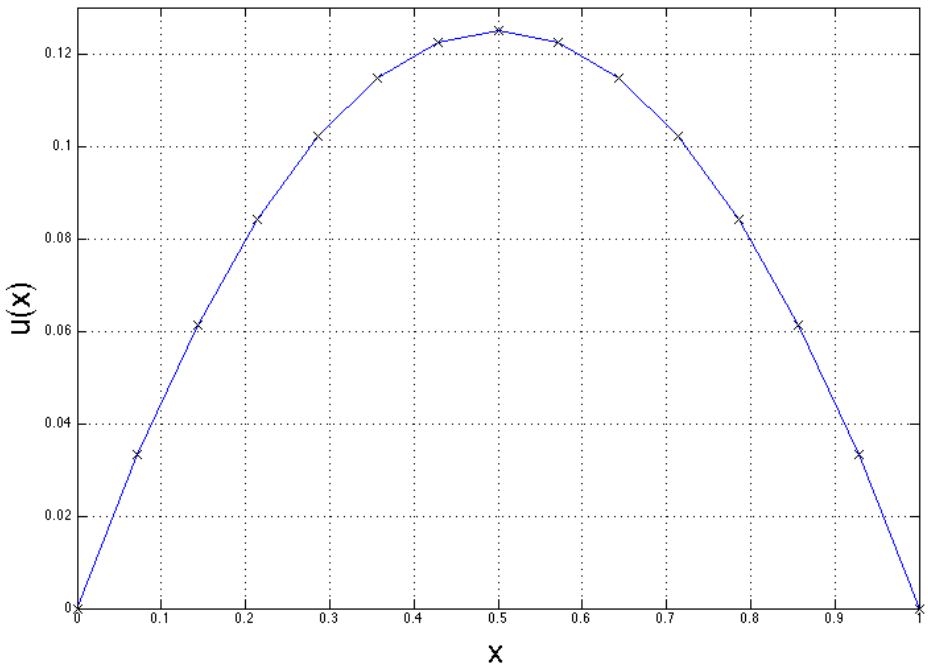


Figure 1.5: Finite element solution to equation (1.7.16) generated using the AptoPy code presented in Section 1.7.6.

```
V = FemFunctionSpace(mesh)
V.dirichlet(dS, 0.0)
u, v = V.trial(), V.test()
```

Now the residual of equation (1.7.16) can be constructed as follows.

```
residual = diff(u, x)*diff(v, x)*dx - 1.0*v*dx
```

Finally, the linear system can be solved for u_h by constructing a solution vector and solving the system of equations.

```
U = SolutionVector(V)
newton_solve(residual, U)
```

The result of running this code in AptoPy is shown in Figure 1.5.

1.8 Outline of Thesis

We first introduce the MPA-CVD reactor model in Chapter 2 as a continuum model of conservation laws. We then introduce the DG finite element method along with

preliminary definitions of notation in Chapter 3. The DG discretisation of the MPA-CVD reactor model, along with an abstract definition of a reactor geometry is discussed in Chapter 4 providing a summary of the PDEs to be solved subject to appropriate boundary conditions. Following their summary in the preceding chapter, Chapter 5 gives details of the design decisions, architecture and syntax of the AptoPy package, which will be used to compute the numerical approximation to the solution of each of the underlying PDEs in the system. A demonstration of the validity and performance of the AptoPy code is presented thereafter in Chapter 6. Based on the work presented in all of the preceding chapters, the numerical approximations to the MPA-CVD reactor model for a number of reactor designs is presented in Chapter 7. Finally we present our conclusions and possible avenues of future research in Chapter 8.

CHAPTER 2

Microwave Power Assisted Chemical Vapour Deposition Reactor Model

2.1 Introduction

The primary purpose of the CVD reactor model is to determine the shape, density and temperature of the ignited plasma above the diamond substrate surface. A self consistent description must couple the electric field propagation in the reactor cavity along with its perturbation resulting from the plasma. Further to this, the transport phenomena arising from particle conservation of the low pressure hydrogen gas should be accounted for. At high gas inlet flow rates the convective effects of the gas flow will have significant impact in the direction of the flow field streamlines. In this chapter we introduce the notion of conserving molecular and atomic hydrogen particle species' densities given their mass average flow field and heterogeneous chemistry due to chemical reaction. The time harmonic description of the CVD reactor's coupled electric field is then discussed for a medium whose electric permittivity is dependent on the characteristics of the reactor plasma. Finally, the electron particle conservation law is introduced subject to the gain and loss of electrons due to ionisation and electron-ion recombination, respectively.

2.2 Mass Averaged Chemically Reacting Flow

In order to adequately model the position and fraction of dissociated hydrogen in a gas mixture of molecular hydrogen, the mass transport phenomena and heat of the gas

mixture must be accounted for. In this section we consider flow for a multicomponent mixture comprising of n different chemically interacting species. The model presented accounts for conservation of each species' density, momentum and energy density. The theory of multicomponent flow is then applied to a binary gas mixture composed of molecular and atomic hydrogen. In this case, consideration will only be given to two components and the two chemical reactions enabling dissociation of molecular hydrogen. Here, the two components of atomic and molecular hydrogen will be labelled H and H_2, respectively. The derivation of this model closely follows the theory of multicomponent flow outlined by Bird, Stewart and Lightfoot [18].

2.2.1 Preliminaries of Multicomponent Flow

Consider the system of n interacting gaseous species. Each gas species has density ρ_i and velocity $\mathbf{u}_i$, $i = 1, \ldots, n$. For each species, the rate of increase of mass must be balanced by the net rate of addition of mass to that volume by flow convection, molecular diffusion and chemical reaction. Thereby, the conservation of mass of component i gives

$$\frac{\partial \rho_i}{\partial t} + \nabla \cdot (\rho_i \mathbf{u}_i) = r_i, \tag{2.2.1}$$

where r_i, $i = 1, \ldots, n$, is the rate of mass production of component i due to chemical reaction. Summing over all components $i = 1, \ldots, n$ yields

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.2.2}$$

where we employ the definitions of total density ρ and mass averaged flow velocity $\mathbf{u}$

$$\rho = \sum_{i=1}^{n} \rho_i, \tag{2.2.3}$$

$$\mathbf{u} = \frac{\sum_{i=1}^{n} \rho_i \mathbf{u}_i}{\rho}, \tag{2.2.4}$$

along with conservation of mass

$$\sum_{i=1}^{n} r_i = 0. \tag{2.2.5}$$

Introducing the diffusion flux of component i, $i = 1, \ldots, n$,

$$\mathbf{j}_i = \rho_i (\mathbf{u}_i - \mathbf{u}), \tag{2.2.6}$$

we rewrite equation (2.2.1) in the following form by employing the identity $\rho_i \mathbf{u} - \rho_i \mathbf{u} = \mathbf{0}$

$$\frac{\partial \rho_i}{\partial t} + \nabla \cdot (\rho_i \mathbf{u} + \mathbf{j}_i) = r_i, \tag{2.2.7}$$

where we note that the total diffusive flux

$$\sum_{i=1}^{n} \mathbf{j}_i = \mathbf{0}. \tag{2.2.8}$$

The conservation of mass has so far been expressed in terms of density, but we can further express this equation in terms of the molar concentration c_i of each component, $i = 1, \ldots, n$, and molar concentration of the mixture $c = \sum_{i=1}^{n} c_i$. Given gas component molar mass M_i, $i = 1, \ldots, n$, we note the relation between density and molar concentration

$$\rho_i = M_i c_i, \tag{2.2.9}$$

which along with the definition of the mixture mean molar mass

$$M = \frac{\sum_{i=1}^{n} c_i M_i}{c}, \tag{2.2.10}$$

we note that

$$\rho = Mc. \tag{2.2.11}$$

The conservation of component i in terms of molar concentration is

$$\frac{\partial c_i}{\partial t} + \nabla \cdot (c_i \mathbf{u}_i) = R_i, \tag{2.2.12}$$

where R_i, $i = 1, \ldots, n$, is the molar rate of mass production of component i due to chemical reaction, such that $r_i = R_i M_i$. Summing equation (2.2.12) over all species and noting that moles are, in general, not conserved yields

$$\frac{\partial c}{\partial t} + \nabla \cdot (c \mathbf{u}^*) = \sum_{i=1}^{n} R_i, \tag{2.2.13}$$

for molar mass average flow velocity

$$\mathbf{u}^* = \frac{\sum_{i=1}^{n} c_i \mathbf{u}_i}{c}. \tag{2.2.14}$$

Introducing the molar diffusion fluxes of each component

$$\mathbf{J}_i^* = c_i \left(\mathbf{u}_i - \mathbf{u}^* \right), \tag{2.2.15}$$

we write equation (2.2.12) as follows

$$\frac{\partial c_i}{\partial t} + \nabla \cdot (c_i \mathbf{u}^* + \mathbf{J}_i^*) = R_i, \tag{2.2.16}$$

where $\sum_{i=1}^{n} \mathbf{J}_i^* = \mathbf{0}$.

Further quantities of interest include the mass fraction $\omega_i = \rho_i / \rho$, $i = 1, \ldots, n$, and molar mass fraction $x_i = c_i / c$, $i = 1, \ldots, n$, provide insight into the empirically measurable state of a multicomponent system. Furthermore, we may also compute the

number density of each species, n_i, $i = 1, \ldots, n$, in terms of their molecular mass, m_i, $i = 1, \ldots, n$, where

$$\begin{aligned} n_i &= \frac{\rho_i}{m_i} = \frac{\omega_i \rho}{m_i} = \frac{M_i x_i}{M} \frac{Mc}{m_i}, \\ &= \frac{M_i x_i c}{m_i}. \end{aligned} \tag{2.2.17}$$

2.2.2 Conservation of Mass in a Binary Gas

Recall mass and molar mass conservation equations (2.2.7) and (2.2.16), respectively. For the hydrogen binary gas, the diffusion fluxes $\mathbf{j}_{\mathrm{H}}$ and $\mathbf{j}_{\mathrm{H}_2}$ or $\mathbf{J}^*_{\mathrm{H}}$ and $\mathbf{J}^*_{\mathrm{H}_2}$ should be expressed in terms of gradients of the concentration, temperature and pressure in order to yield a closed system of equations for the self consistent model. Diffusion due to pressure gradients generally occurs only when the gradient in pressure is very large, a condition not found in MPA-CVD reactors which leads us to neglect these terms. Therefore, the diffusion flux takes into account concentration gradients of the mixture, as well as thermal terms. In general, $\mathbf{j}_i$ is determined by the Maxwell-Stefan equations; however, in the case of a binary gas mixture, the binary diffusion flux is given by Fick's law along with thermal terms. Specifically for atomic hydrogen in a binary gas mixture of atomic and molecular hydrogen

$$\mathbf{j}_{\mathrm{H}} = -\rho \mathcal{D}_{\mathrm{HH}_2} \nabla \omega_{\mathrm{H}} - D^T_{\mathrm{H}} \nabla \ln T. \tag{2.2.18}$$

Here, $\mathcal{D}_{\mathrm{AB}}$ is the binary diffusion coefficient for component A in a mixture consisting of A and B, D^T_{A} is the thermal diffusivity coefficient for component A (note that $D^T_{\mathrm{A}} = -D^T_{\mathrm{B}}$ for a binary gas mix of A and B), and T is the temperature of the mixture. The molar diffusion flux for a binary gas mixture of atomic and molecular hydrogen is given by the equivalent formulation of equation (2.2.18), namely,

$$\mathbf{J}^*_{\mathrm{H}} = -c \mathcal{D}_{\mathrm{HH}_2} \nabla x_{\mathrm{H}} - \left(\frac{M}{M_{\mathrm{H}} M_{\mathrm{H}_2}} \right) D^T_{\mathrm{H}} \nabla \ln T. \tag{2.2.19}$$

Equation (2.2.19) is derived from equation (2.2.18) as follows: initially, recall the molar diffusion flux (2.2.15) with $i = \mathrm{H}$, and note that

$$\begin{aligned} \mathbf{J}^*_{\mathrm{H}} &= c_{\mathrm{H}} \left(\mathbf{u}_{\mathrm{H}} - \mathbf{u}^* \right) \\ &= c_{\mathrm{H}} \left(\mathbf{u}_{\mathrm{H}} - \mathbf{u} \right) - c_{\mathrm{H}} \left(\mathbf{u}^* - \mathbf{u} \right). \end{aligned} \tag{2.2.20}$$

The difference between the binary mixture's molar mass averaged velocity and its mass averaged velocity is given by

$$\begin{aligned}
\mathbf{u}^* - \mathbf{u} &= x_{\mathrm{H}}\mathbf{u}_{\mathrm{H}} + x_{\mathrm{H}_2}\mathbf{u}_{\mathrm{H}_2} - \mathbf{u} \\
&= x_{\mathrm{H}}\left(\mathbf{u}_{\mathrm{H}} - \mathbf{u}\right) + x_{\mathrm{H}_2}\left(\mathbf{u}_{\mathrm{H}_2} - \mathbf{u}\right) && (x_{\mathrm{H}} + x_{\mathrm{H}_2} = 1) \\
&= \frac{x_{\mathrm{H}}}{\rho_{\mathrm{H}}}\mathbf{j}_{\mathrm{H}} + \frac{x_{\mathrm{H}_2}}{\rho_{\mathrm{H}_2}}\mathbf{j}_{\mathrm{H}_2} && (\text{Equation (2.2.6)}) \\
&= \left(\frac{x_{\mathrm{H}}}{\rho_{\mathrm{H}}} - \frac{x_{\mathrm{H}_2}}{\rho_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}}. && (\text{Equation (2.2.8)}) \qquad (2.2.21)
\end{aligned}$$

Substituting (2.2.21) into (2.2.20) gives

$$\begin{aligned}
\mathbf{J}^*_{\mathrm{H}} &= \frac{c_{\mathrm{H}}}{\rho_{\mathrm{H}}}\mathbf{j}_{\mathrm{H}} - c_{\mathrm{H}}\left(\frac{x_{\mathrm{H}}}{\rho_{\mathrm{H}}} - \frac{x_{\mathrm{H}_2}}{\rho_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}} \\
&= \left(\frac{1}{M_{\mathrm{H}}} - \frac{x_{\mathrm{H}}}{M_{\mathrm{H}}} + \frac{x_{\mathrm{H}}}{M_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}} && \left(\frac{1}{M_i} = \frac{c_i}{\rho_i},\ \ x_i = \frac{c_i}{c}\right) \\
&= \left(\frac{x_{\mathrm{H}_2}}{M_{\mathrm{H}}} + \frac{x_{\mathrm{H}}}{M_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}} && (1 - x_{\mathrm{H}} = x_{\mathrm{H}_2}) \\
&= \left(\frac{M}{M_{\mathrm{H}}M_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}}. && (M = x_{\mathrm{H}}M_{\mathrm{H}} + x_{\mathrm{H}_2}M_{\mathrm{H}_2}) \qquad (2.2.22)
\end{aligned}$$

In essence the relationship between equations (2.2.18) and (2.2.19) is that $\mathbf{J}^*_{\mathrm{H}} = \left(\frac{M}{M_{\mathrm{H}}M_{\mathrm{H}_2}}\right)\mathbf{j}_{\mathrm{H}}$. This is clear in the second term of (2.2.19), and becomes clear in the first by noting that

$$\begin{aligned}
-\rho\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla\omega_{\mathrm{H}} &= -\rho\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla\left(\frac{\rho_{\mathrm{H}}}{\rho}\right) \\
&= -\rho\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla\left(\frac{M_{\mathrm{H}}c_{\mathrm{H}}}{Mc}\right) \\
&= -\rho\mathcal{D}_{\mathrm{H}\mathrm{H}_2}M_{\mathrm{H}}\nabla\left(\frac{x_{\mathrm{H}}}{M}\right) \\
&= -\rho\mathcal{D}_{\mathrm{H}\mathrm{H}_2}M_{\mathrm{H}}\left(\frac{M\nabla x_{\mathrm{H}} - x_{\mathrm{H}}\nabla M}{M^2}\right) \\
&= -\rho\frac{M_{\mathrm{H}}M_{\mathrm{H}_2}}{M^2}\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla x_{\mathrm{H}} \\
&= -c\frac{M_{\mathrm{H}}M_{\mathrm{H}_2}}{M}\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla x_{\mathrm{H}}. \qquad (2.2.23)
\end{aligned}$$

For the hydrogen binary gas, combining the equation of molar concentration conservation (2.2.16) for $i = \mathrm{H}$ and the definition of the molar flux in equation (2.2.19) yields the conservation equation

$$\frac{\partial c_{\mathrm{H}}}{\partial t} + \nabla\cdot\left(c_{\mathrm{H}}\mathbf{u}^*\right) - \nabla\cdot\left(c\mathcal{D}_{\mathrm{H}\mathrm{H}_2}\nabla x_{\mathrm{H}} + \left(\frac{M}{M_{\mathrm{H}}M_{\mathrm{H}_2}}\right)D^T_{\mathrm{H}}\nabla\ln T\right) = R_{\mathrm{H}}. \qquad (2.2.24)$$

We require (2.2.24) to be expressed in terms of the mass average flow $\mathbf{u}$ in order to easily couple with the conservation of momentum model which will be discussed in

the following section. To eliminate $\mathbf{u}^*$ we note that by employing the definiton of the mass average flow (2.2.4), we get

$$\begin{aligned}
\mathbf{u}^* &= \mathbf{u} - \omega_{\mathrm{H}}\mathbf{u}_{\mathrm{H}} - \omega_{\mathrm{H}_2}\mathbf{u}_{\mathrm{H}_2} + \mathbf{u}^* \\
&= \mathbf{u} - \omega_{\mathrm{H}}\left(\mathbf{u}_{\mathrm{H}} - \mathbf{u}^*\right) - \omega_{\mathrm{H}_2}\left(\mathbf{u}_{\mathrm{H}_2} - \mathbf{u}^*\right) && \left(\omega_{\mathrm{H}} + \omega_{\mathrm{H}_2} = 1\right) \\
&= \mathbf{u} - \frac{\omega_{\mathrm{H}}}{c_{\mathrm{H}}}\mathbf{J}_{\mathrm{H}}^* - \frac{\omega_{\mathrm{H}_2}}{c_{\mathrm{H}_2}}\mathbf{J}_{\mathrm{H}_2}^* && \text{(Equation (2.2.15))} \\
&= \mathbf{u} + \frac{M_{\mathrm{H}_2} - M_{\mathrm{H}}}{\rho}\mathbf{J}_{\mathrm{H}}^*. && \left(\omega_i = \frac{\rho_i}{\rho},\ c_i = \frac{\rho_i}{M_i}\right)
\end{aligned} \tag{2.2.25}$$

Hence,

$$c_{\mathrm{H}}\mathbf{u}^* = c_{\mathrm{H}}\mathbf{u} + x_{\mathrm{H}}\left(\frac{M_{\mathrm{H}_2} - M_{\mathrm{H}}}{M}\right)\mathbf{J}_{\mathrm{H}}^*. \tag{2.2.26}$$

Substituting this into (2.2.24) yields the conservation of mass in terms of each species' molar concentrations and their gradients

$$\begin{aligned}
\frac{\partial c_{\mathrm{H}}}{\partial t} + \nabla\cdot\left(c_{\mathrm{H}}\mathbf{u} + x_{\mathrm{H}}\left(\frac{M_{\mathrm{H}_2} - M_{\mathrm{H}}}{M}\right)\mathbf{J}_{\mathrm{H}}^* + \mathbf{J}_{\mathrm{H}}^*\right) &= R_{\mathrm{H}} \\
\frac{\partial c_{\mathrm{H}}}{\partial t} + \nabla\cdot\left(c_{\mathrm{H}}\mathbf{u} + \frac{x_{\mathrm{H}}\left(M_{\mathrm{H}_2} - M_{\mathrm{H}}\right) + M}{M}\mathbf{J}_{\mathrm{H}}^*\right) &= R_{\mathrm{H}} \\
\frac{\partial c_{\mathrm{H}}}{\partial t} + \nabla\cdot\left(c_{\mathrm{H}}\mathbf{u} + \frac{M_{\mathrm{H}_2}}{M}\mathbf{J}_{\mathrm{H}}^*\right) &= R_{\mathrm{H}},
\end{aligned} \tag{2.2.27}$$

from which we get the following conservation equation

$$\frac{\partial(cx_{\mathrm{H}})}{\partial t} + \nabla\cdot(cx_{\mathrm{H}}\mathbf{u}) - \nabla\cdot\left(\frac{M_{\mathrm{H}_2}}{M}c\mathcal{D}_{\mathrm{HH}_2}\nabla x_{\mathrm{H}} + \frac{D_{\mathrm{H}}^T}{M_{\mathrm{H}}}\nabla\ln T\right) = R_{\mathrm{H}}. \tag{2.2.28}$$

It should be noted here that the thermal diffusivity of atomic and molecular hydrogen is often small, especially in the case of the low pressure gas mixture in a CVD reactor, and can thereby be neglected [117].

In order to define the density and concentration of a mixture, here the equations of state for an ideal gas are introduced:

$$c = \frac{P}{RT}, \quad \rho = \frac{MP}{RT}, \tag{2.2.29}$$

where P is the constant mean pressure of the system and R is the gas constant. Noting that $\rho = Mc$, ρ can be defined in terms of molar mass fractions comprising the mixture, i.e.,

$$\rho = \frac{P}{RT}\sum_{i=1}^{n} M_i x_i. \tag{2.2.30}$$

The binary gas mixture's molar mass fractions of atomic and molecular hydrogen can be expressed in terms of $x_{\mathrm{H}_2} = 1 - x_{\mathrm{H}}$ yielding the expression for ρ

$$\rho = \frac{P}{RT}\left(M_{\mathrm{H}_2}\left(1 - x_{\mathrm{H}}\right) + M_{\mathrm{H}}x_{\mathrm{H}}\right). \tag{2.2.31}$$

In the case of the ideal gas, we may also extend the calculation of the gas species number density in (2.2.17) such that the number densities of atomic and molecular hydrogen are given by

$$n_{\mathrm{H}} = \frac{M_{\mathrm{H}} x_{\mathrm{H}}}{m_{\mathrm{H}}} \frac{P}{RT}, \tag{2.2.32}$$

$$n_{\mathrm{H}_2} = \frac{M_{\mathrm{H}_2} (1 - x_{\mathrm{H}})}{m_{\mathrm{H}_2}} \frac{P}{RT}. \tag{2.2.33}$$

In the specific case of a binary gas mixture involving the dissociation of hydrogen, two simple reactions are assumed to take place,

$$\mathrm{H}_2 + \mathrm{H}_2 \rightleftharpoons \mathrm{H} + \mathrm{H} + \mathrm{H}_2 \tag{2.2.34}$$

$$\mathrm{H}_2 + \mathrm{H} \rightleftharpoons \mathrm{H} + \mathrm{H} + \mathrm{H} \tag{2.2.35}$$

Assuming the chemical reaction rate constants to be the same for both reactions, the reaction rate for atomic hydrogen is given by

$$R_{\mathrm{H}} = R_f - R_r. \tag{2.2.36}$$

Here, R_f is the forward molar rate of production by dissociation and R_r is the reverse molar rate of production. Each of R_f and R_r are determined by the chemical kinetics of each reaction, depending on the concentration of each species H and H_2 along with the forward and reverse reaction rate coefficients, k_f and k_r, respectively. We refer to [18] regarding details of chemical kinetics in reaction terms. In the case of the rate of production of H by dissociation from H_2 we write

$$\begin{aligned} R_{\mathrm{H}} &= 2 \left(\underbrace{k_f c_{\mathrm{H}_2}^2 - k_r c_{\mathrm{H}}^2 c_{\mathrm{H}_2}}_{\text{reaction (2.2.34)}} + \underbrace{k_f c_{\mathrm{H}_2} c_{\mathrm{H}} - k_r c_{\mathrm{H}}^3}_{\text{reaction (2.2.35)}} \right) \\ &= 2 \left(k_f \left(c^2 (1 - x_{\mathrm{H}})^2 + c^2 x_{\mathrm{H}} (1 - x_{\mathrm{H}}) \right) - k_r \left(c^3 x_{\mathrm{H}}^2 (1 - x_{\mathrm{H}}) + c^3 x_{\mathrm{H}}^3 \right) \right) \\ &= 2 \left(k_f c^2 (1 - x_{\mathrm{H}}) - k_r c^3 x_{\mathrm{H}}^2 \right), \end{aligned} \tag{2.2.37}$$

where the factor of 2 accounts for 2 H atoms produced per reaction and we assume the forward and reverse rate coefficients to be equivalent for each reaction.

2.2.3 Conservation of Momentum

Assuming the viscosity η of the mixture of n individual chemical species to be isotropic, the conservation of momentum is described by the quasi-incompressible Navier Stokes

equations

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) - \nabla\cdot\tau = \rho\mathbf{g}, \tag{2.2.38}$$

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0. \tag{2.2.39}$$

Here, $\mathbf{g}$ is the acceleration due to gravity and the stress tensor τ is defined as

$$\tau = -p\underline{\mathbf{I}} + \eta\left(\nabla\mathbf{u} + \nabla\mathbf{u}^{\top} - \tfrac{2}{3}(\nabla\cdot\mathbf{u})\,\underline{\mathbf{I}}\right), \tag{2.2.40}$$

where p is the pressure and $\underline{\mathbf{I}}$ is the identity tensor. The viscosity of the mixture can be determined by the semi-empirical formula [144]

$$\eta = \sum_{i=1}^{n} \frac{x_i\eta_i}{\sum_{j=1}^{n} x_j\Phi_{ij}}, \tag{2.2.41}$$

where η_i is the viscosity of the ith component of the mixture, and Φ_{ij} is given by

$$\Phi_{ij} = \frac{1}{\sqrt{8}}\left(1+\frac{M_i}{M_j}\right)^{-\frac{1}{2}}\left(1+\left(\frac{\eta_i}{\eta_j}\right)^{\frac{1}{2}}\left(\frac{M_j}{M_i}\right)^{\frac{1}{4}}\right)^{2}. \tag{2.2.42}$$

In the case of the binary gas mixture of atomic and molecular hydrogen

$$\eta = \left(\frac{6x_{\mathrm{H}}}{6x_{\mathrm{H}} + \sqrt{3}\left(1+2^{\frac{1}{4}}N\right)^{2}(1-x_{\mathrm{H}})} + \frac{12\,(1-x_{\mathrm{H}})}{12N^{2}\,(1-x_{\mathrm{H}}) + \sqrt{3}\left(1+2^{\frac{1}{4}}N\right)^{2}x_{\mathrm{H}}}\right)\eta_{\mathrm{H}}, \tag{2.2.43}$$

where $N = \sqrt{\frac{\eta_{\mathrm{H}}}{\eta_{\mathrm{H_2}}}}$ [144].

2.2.4 Conservation of Energy

The conservation of the energy density ρE of a multicomponent mixture is governed by

$$\frac{\partial(\rho E)}{\partial t} + \nabla\cdot((\rho E - \tau)\,\mathbf{u}) + \nabla\cdot\mathbf{q} = Q + \rho\mathbf{u}\cdot\mathbf{g}, \tag{2.2.44}$$

where $\mathbf{q}$ is the energy flux of the mixture, Q is an energy source, $\rho\mathbf{u}\cdot\mathbf{g}$ is the power expended from external force acting on the gas mixture and E is the total energy of the system. E can also be expressed as $E = e + \frac{1}{2}u^2$, where e is the internal energy per unit mass, and $\frac{1}{2}u^2$ is the kinetic energy per unit mass, with $u^2 = \mathbf{u}\cdot\mathbf{u}$.

Now consider the mechanical energy equation, formulated by taking the product of the conservation of momentum equation (2.2.38) with $\mathbf{u}$ and rearranging with respect to the continuity equation [18]

$$\frac{\partial(\frac{1}{2}\rho u^2)}{\partial t} + \nabla\cdot\left(\tfrac{1}{2}\rho u^2\mathbf{u}\right) - \nabla\cdot(\tau\mathbf{u}) + \tau : \nabla\mathbf{u} = \rho\mathbf{u}\cdot\mathbf{g}. \tag{2.2.45}$$

By subtracting the mechanical energy in equation (2.2.45) from the conservation of energy in equation (2.2.44) it can be shown that

$$\frac{\partial(\rho e)}{\partial t} + \nabla \cdot (\rho e \mathbf{u}) - \tau : \nabla \mathbf{u} + \nabla \cdot \mathbf{q} = Q. \tag{2.2.46}$$

Similar to the process for finding the conservation of mass in terms of molar mass fractions, the conservation of energy equations should be reduced from their thermodynamic internal energy description to a form which is easily measured empirically. In this case, the conservation of energy should be expressed in terms of the system's temperature T and specific enthalpy $h = h(T)$. The specific enthalpy for the multicomponent mixture can be expressed in terms of each of the specific enthalpies of the mixture's species, weighted by their respective mass fraction, i.e.,

$$h = \sum_{i=1}^{n} \omega_i h_i. \tag{2.2.47}$$

The enthalpy of the system is composed of its internal energy in addition to the thermodynamic work done by the system on its adiabatic chamber, i.e.,

$$h = e + \frac{p}{\rho}, \tag{2.2.48}$$

where p is the pressure of the gas mixture. Substituting the expression for enthalpy given in equation (2.2.48) into the conservation of internal energy equation (2.2.46), it can be shown that

$$\begin{aligned} \frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{u}) + \nabla \cdot \mathbf{q} &= \frac{\partial p}{\partial t} + \nabla \cdot (p\mathbf{u}) + \tau : \nabla \mathbf{u} + Q \\ &= \frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p + \tau_{\text{dev}} : \nabla \mathbf{u} + Q, \end{aligned} \tag{2.2.49}$$

where we have introduced the deviatoric component of τ, given by

$$\tau_{\text{dev}} = \eta \left(\nabla \mathbf{u} + \nabla \mathbf{u}^{\top} - \tfrac{2}{3} (\nabla \cdot \mathbf{u}) \underline{\mathbf{I}} \right). \tag{2.2.50}$$

In the CVD reactor model we assume the viscous dissipation effects in the CVD vacuum to be negligible as they are important only in flows with enormous velocity gradients; thereby, we set

$$\tau_{\text{dev}} : \nabla \mathbf{u} = 0. \tag{2.2.51}$$

We further state the ideal gas assumption of (2.2.29) that the gas flow in the CVD reactor is flowing in a system with constant mean pressure P, i.e., we take

$$\frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p = 0. \tag{2.2.52}$$

Upon making these assumptions, equation (2.2.49) reduces to

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{u}) + \nabla \cdot \mathbf{q} = Q. \tag{2.2.53}$$

An expression for the energy flux can be found in terms of Fourier heat diffusion, where the Dufour effect of energy transport due to concentration gradients between chemical species is small and can be neglected [18]. Here, for multicomponent gas mixture temperature

$$\mathbf{q} = -\kappa \nabla T + \sum_{i=1}^{n} h_i \mathbf{j}_i. \tag{2.2.54}$$

The thermal conductivity κ can be expressed in terms of each chemical species thermal conductivity κ_i, $i = 1, \ldots, n$, (see [18]), i.e.,

$$\kappa = \frac{1}{2}\left(\sum_{i=1}^{n} x_i \kappa_i + \left(\sum_{i=1}^{n} \frac{x_i}{\kappa_i}\right)^{-1}\right). \tag{2.2.55}$$

The term involving the spatial gradients of the species' specific enthalpies in equation (2.2.54) is often small and can be neglected [18], leaving the form of the conservation of energy in terms of enthalpy and temperature given by

$$\frac{\partial (\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{u}) - \nabla \cdot (\kappa \nabla T) = Q. \tag{2.2.56}$$

A thermodynamic quantity of interest in the CVD reactor is the specific heat of the plasma, namely,

$$c_p = \left(\frac{\partial h}{\partial T}\right)_{p,\omega_i}. \tag{2.2.57}$$

In the specific case of the binary gas mixture, its specific heat is given by

$$c_p = \omega_{\mathrm{H}} c_{p,\mathrm{H}} + (1 - \omega_{\mathrm{H}}) c_{p,\mathrm{H}_2}, \tag{2.2.58}$$

where $c_{p,\mathrm{H}}$ and c_{p,H_2} are the specific heats of each species. It is worth noting that empirical data is usually obtained by measuring molar heat capacities, $C_{p,i}$, $i = 1, \ldots, n$, together with its relation to the specific heat capacity,

$$c_{p,i} = \frac{C_{p,i}}{M_i}. \tag{2.2.59}$$

2.3 Electromagnetic Waveguides and Resonators

The coupling of a microwave frequency electric field within the CVD reactor is key to the localised heating of the gas mixture above the diamond substrate. For a closed cavity whose walls act as perfect conductors, there are infinitely many harmonic electrical resonant modes at an equal number of corresponding frequencies which can be excited. Common commercially available magnetron devices operate at frequencies of 896 MHz and 2.45 GHz. A key aspect of CVD reactor design is to ensure that the

reactor's geometry supports a given resonance mode at the frequency of the driving magnetron. Furthermore, it is favourable that the profile of the resonant electric field's amplitude be at its greatest magnitude above the substrate surface, ensuring efficient dissociation of hydrogen above the diamond substrate surface.

2.3.1 Wave Propagation in Lossy Dielectrics

Electromagnetic fields are created by static charge distributions and directional flow of electric charge. For a scalar distribution of charge density denoted by ρ_e, and a current flow of electric charge $\mathbf{i}$, Maxwell's equations state the electric and magnetic field intensities $\mathcal{E}(\mathbf{x}, t)$ and $\mathcal{B}(\mathbf{x}, t)$, respectively, and the electric displacement and magnetic induction fields $\mathcal{D}(\mathbf{x}, t)$ and $\mathcal{H}(\mathbf{x}, t)$, respectively, are related by

$$\frac{\partial \mathcal{B}}{\partial t} + \nabla \times \mathcal{E} = \mathbf{0}, \tag{2.3.1}$$

$$\nabla \cdot \mathcal{D} = \rho_e, \tag{2.3.2}$$

$$\frac{\partial \mathcal{D}}{\partial t} - \nabla \times \mathcal{H} = -\mathbf{i}, \tag{2.3.3}$$

$$\nabla \cdot \mathcal{B} = \mathbf{0}. \tag{2.3.4}$$

The electromagnetic field, electric displacement and magnetic induction fields are related by

$$\mathcal{D} = \varepsilon \mathcal{E}, \tag{2.3.5}$$

$$\mathcal{B} = \mu \mathcal{H}, \tag{2.3.6}$$

where ε and μ are the material permittivity and permeability, respectively.

Analysing electromagnetic wave propagation at a single frequency, the time-dependent Maxwell's equations can be reduced to the time-harmonic Maxwell system. For a given radiative temporal angular frequency ω, the electromagnetic field is time-harmonic if

$$\mathcal{E}(\mathbf{x}, t) = \Re(\mathbf{E}(\mathbf{x})e^{j\omega t}), \qquad \mathcal{D}(\mathbf{x}, t) = \Re(\mathbf{D}(\mathbf{x})e^{j\omega t}), \tag{2.3.7}$$

$$\mathcal{B}(\mathbf{x}, t) = \Re(\mathbf{B}(\mathbf{x})e^{j\omega t}), \qquad \mathcal{H}(\mathbf{x}, t) = \Re(\mathbf{H}(\mathbf{x})e^{j\omega t}), \tag{2.3.8}$$

for time t, imaginary unit $j = \sqrt{-1}$, and complex-valued phasor terms $\mathbf{E}$, $\mathbf{D}$, $\mathbf{B}$ and $\mathbf{H}$. Similarly, the charge and current densities can be expressed in phasor form,

$$\rho_e(\mathbf{x}, t) = \Re(\rho_v(\mathbf{x})e^{j\omega t}), \tag{2.3.9}$$

$$\mathbf{i} = \Re(\hat{\mathbf{i}}(\mathbf{x})e^{j\omega t}). \tag{2.3.10}$$

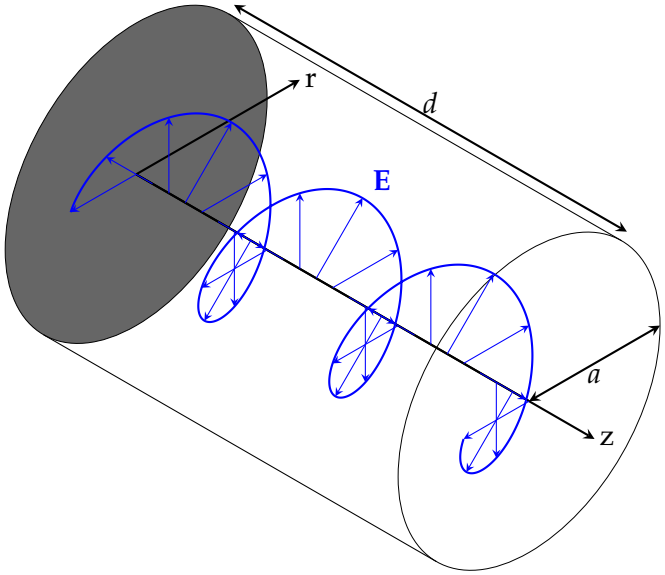


Figure 2.1: Cylindrical waveguide geometry of radius a and length d.

Substituting these expressions into Maxwell's equations and exploiting the electromagnetic field relations given in (2.3.5) and (2.3.6) along with the current density approximation for electrical conductivity σ, namely,

$$\mathbf{i} = \sigma \mathcal{E}, \tag{2.3.11}$$

yields the time-harmonic formulation of Maxwell's equations for a dielectric medium

$$\nabla \times \mathbf{E} = -j\omega\mu\mathbf{H}, \tag{2.3.12}$$

$$\nabla \cdot (\varepsilon \mathbf{E}) = \rho_v, \tag{2.3.13}$$

$$\nabla \times \mathbf{H} = (\sigma + j\omega\varepsilon)\mathbf{E}, \tag{2.3.14}$$

$$\nabla \cdot (\mu \mathbf{H}) = 0. \tag{2.3.15}$$

Applying the curl operator to equation (2.3.12) and substituting into (2.3.14) yields the time harmonic E-field formulation

$$\nabla \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) + j\omega \left(\sigma + j\omega\varepsilon\right) \mathbf{E} = \mathbf{0}. \tag{2.3.16}$$

Similarly, by applying the curl operator to equation (2.3.14) and substituting into (2.3.12) yields the time harmonic H-field formulation

$$\nabla \times \left(\left(\sigma + j\omega\varepsilon\right)^{-1} \nabla \times \mathbf{H}\right) + j\omega\mu\mathbf{H} = \mathbf{0}. \tag{2.3.17}$$

2.3.2 Uniform Isotropic Lossless Waveguides

The behaviour of uniform lossless waveguides filled with an isotropic dielectric material is well understood for cross-sectional geometries such as rectangles, circles and coaxial cylinders [99]. Here, the derivation of the solution to the time harmonic E-field formulation will be briefly demonstrated for the empty homogeneous cylindrical waveguide of radius a shown in Figure 2.1, with isotropic material parameters (i.e., $\rho_v = 0$, $\sigma = 0$, ε and μ are constant). We consider the case of the propagating electromagnetic wave operating in a transversal magnetic (TM) mode; this is where the magnetic field only has non-zero components transverse to the direction of wave propagation $\hat{\mathbf{z}}$, i.e., $\mathbf{H} = H_r\hat{\mathbf{r}} + H_\theta\hat{\boldsymbol{\theta}}$. This electric field configuration is commonly used in CVD reactor design.

Let $\Omega = \{r, \theta, z : r < a\}$ with boundary $\partial\Omega$, with unit outward pointing normal $\mathbf{n}$ represent the geometry of the waveguide depicted in Figure 2.1. We wish to solve the time harmonic formulation of Maxwell's equations subject to the perfect electric conductor boundary condition on $\partial\Omega$, namely,

$$\nabla \times \nabla \times \mathbf{E} - \mu\varepsilon\omega^2\mathbf{E} = \mathbf{0} \qquad \text{in } \Omega, \tag{2.3.18}$$

$$\nabla \cdot \mathbf{E} = 0 \qquad \text{in } \Omega, \tag{2.3.19}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega. \tag{2.3.20}$$

Employing the vector identity

$$\nabla \times (\nabla \times \mathbf{A}) = \nabla(\nabla \cdot \mathbf{A}) - \nabla^2\mathbf{A} \tag{2.3.21}$$

in (2.3.18), whilst taking note of the divergence free condition imposed by Gauss' law (2.3.19), yields the time harmonic Helmholtz formulation

$$\nabla^2\mathbf{E} + k^2\mathbf{E} = \mathbf{0} \qquad \text{in } \Omega, \tag{2.3.22}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega, \tag{2.3.23}$$

where $k = \sqrt{\mu\varepsilon}\omega$ is the propagation constant.

Equation (2.3.22) is linearly independent in each orthogonal direction $\hat{\mathbf{r}}$, $\hat{\theta}$ and $\hat{\mathbf{z}}$. As such, solving for only the $\hat{\mathbf{z}}$ component of the electric field allows the remaining components of the electromagnetic field to be derived from the time harmonic E- and H-field formulations of Maxwell's equations. Given the perfect electric conductor boundary condition $E_z = 0$ on the waveguide walls, the solution to

$$\nabla^2 E_z + k^2 E_z = 0 \tag{2.3.24}$$

can be shown to be the radially and azimuthally harmonic function [99]

$$E_z = E_{0z} J_n\left(\frac{X_{np}}{a} r\right) \cos(n\theta) e^{-j\beta_{np} z}. \tag{2.3.25}$$

Here, E_{0z} is the peak field amplitude, n is the integer order of the azimuthal harmonic mode, $J_n(\cdot)$ is the nth order Bessel function of the first kind, X_{np} is the pth root of $J_n(r)$ implicitly dictating the order of the radial harmonic mode and the electric field phase constant β_{np} is specified by

$$\beta_{np} = \sqrt{\mu\varepsilon\omega^2 - \left(\frac{X_{np}}{a}\right)^2}. \tag{2.3.26}$$

The harmonic mode numbers n and p are used to characterise a waveguide configuration with the nomenclature TM_{np}. The propagation term β_{np} determines the nature of the incident electromagnetic field with three cases:

1. *Propagation*: If $\mu\varepsilon\omega^2 > \left(\frac{X_{np}}{a}\right)^2$ the electromagnetic wave propagates due to real valued phase angle of $\Re\left(e^{-j\beta_{np}z}\right) = \cos(\beta_{np} z)$.
2. *Attenuation*: If $\mu\varepsilon\omega^2 < \left(\frac{X_{np}}{a}\right)^2$ the electromagnetic wave is dissipated due to the now complex valued phase angle of $e^{-j\beta_{np}z}$.
3. *Cutoff*: If $\mu\varepsilon\omega^2 = \left(\frac{X_{np}}{a}\right)^2$, this is the minimum angular frequency by which the electromagnetic field can propagate, denoted by

$$\omega_c = \frac{1}{\sqrt{\mu\varepsilon}} \frac{X_{np}}{a}, \quad \text{or} \quad k_c = \frac{X_{np}}{a}. \tag{2.3.27}$$

The full TM cylindrical waveguide electric vector field solution can therefore be shown to be

$$\mathbf{E} = \left(-j\frac{\beta_{np}}{k_c^2}\frac{\partial E_z}{\partial r},\ -j\frac{\beta_{np}}{k_c^2}\frac{1}{r}\frac{\partial E_z}{\partial \theta},\ E_z\right)^\top, \quad n \geq 0,\ p \geq 1. \tag{2.3.28}$$

2.3.3 Microwave Cavity Resonator

A cavity resonator is an electrically conducting enclosure in which electromagnetic energy is confined. Analytically, a given resonator does not offer a unique solution to its contained electromagnetic field, but rather an infinite number of resonant modes, each of which correspond to a given resonant frequency. When the driven frequency of the electromagnetic field corresponds to one of these resonant frequencies, the contained standing wave will reach a maximum amplitude at the cavity's peak energy. The standing wave mode arising from the lowest resonant frequency is known as the dominant mode.

In a given cavity resonator, the contained electric field **E** should satisfy the time harmonic formulation of Maxwell's equations. Consider the waveguide geometry in Figure 2.1 now with perfect electrically conducting walls at $r = a$, $z = 0$ and $z = d$, i.e., let $\Omega = \{r, \theta, z : r < a,\ 0 < z < d\}$ with boundary $\partial\Omega$. As derived for the empty uniform isotropic waveguide, the solution to the time harmonic Helmholtz formulation for a contained TM electromagnetic field must now also be harmonic in the $\hat{\mathbf{z}}$ direction. The solution to E_z can be shown to be

$$E_z = E_{0z} J_n\left(\frac{X_{np}}{a} r\right) \cos(n\theta) \cos\left(\frac{q\pi}{d} z\right), \quad n \geq 0,\ p \geq 1,\ q \geq 0. \tag{2.3.29}$$

Here, q is in the integer modal value of the number of half waves in the axial direction. The enclosed transverse magnetic field in the cavity resonator is characterised by the harmonic mode numbers n, p and q using the nomenclature TM_{npq}. Again, the remaining unknown quantities of the system E_r, E_θ and **H** can be found from the set of simultaneous relations in the time harmonic formulation of Maxwell's equations. Here, the resonant frequency of each harmonic mode is analytically determined by

$$\omega_r = \frac{1}{\sqrt{\mu\varepsilon}} \sqrt{\left(\frac{X_{np}}{a}\right)^2 + \left(\frac{q\pi}{d}\right)^2}. \tag{2.3.30}$$

The electric field solutions of the first four TM_{0pq} resonant modes are shown in Figure 2.2.

2.4 Plasma Ignition

2.4.1 Introduction

The source of electrons in the CVD reactor plasma should be modelled in the same way as the species of hydrogen, in the sense that their number density and internal energy should be conserved. We present here a simple model of electron momentum and energy conservation, along with electron generation by ionisation and loss by recombination with heavy ions according to the Maxwell-Boltzmann distribution. The derivation which follows summarises the plasma physics model presented in Goldston and Rutherford [58] and Lieberman and Lichtenberg [100].

2.4.2 Boltzmann Equation

Let $f(\mathbf{x}, \mathbf{v}, t)$ be the distribution function which describes particle position and velocity in the six-dimensional phase space $(\mathbf{x}, \mathbf{v})$ at time t. The number of particles in a six-

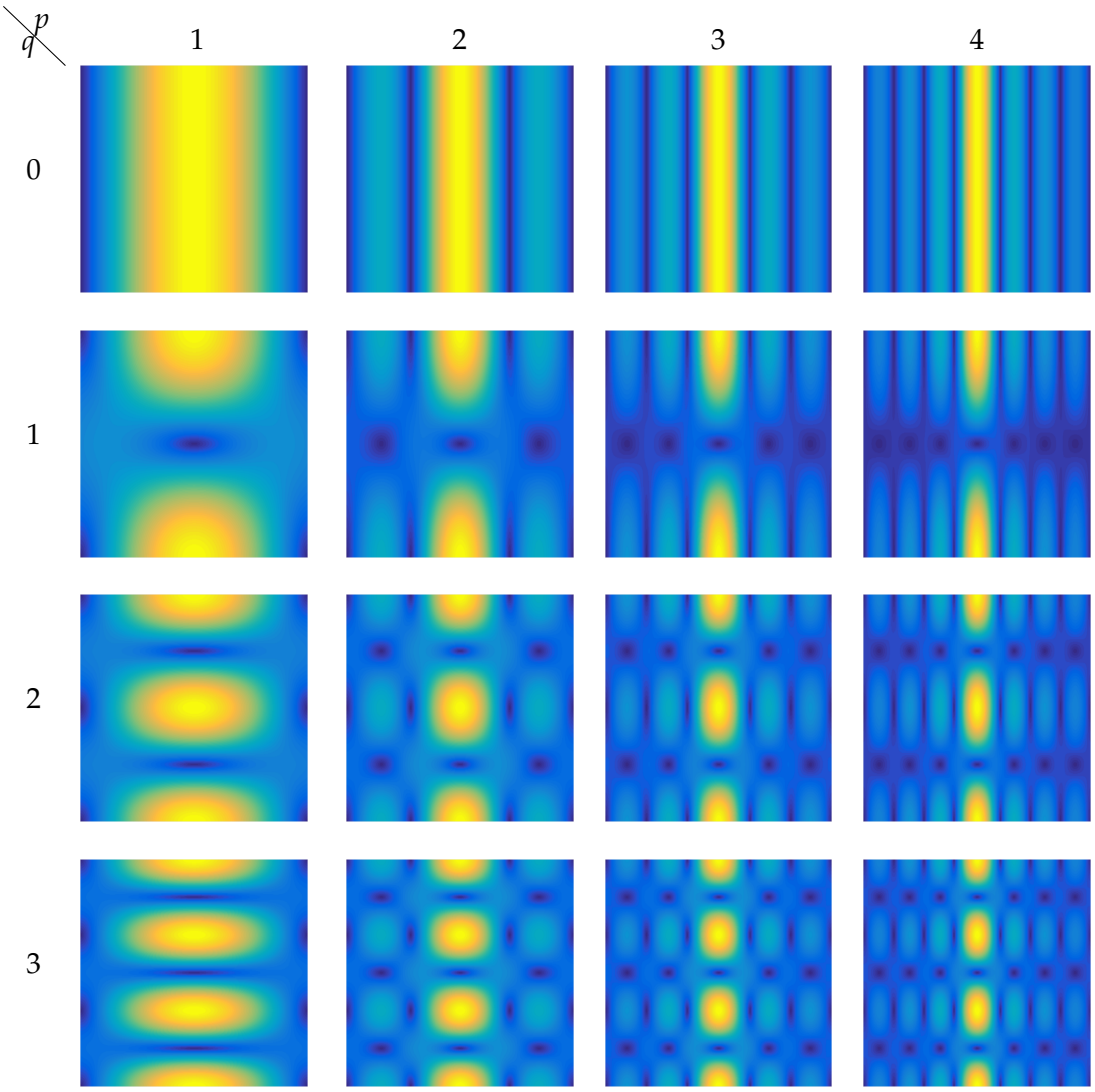


Figure 2.2: The electric field magnitude of the first four TM_{0pq} transverse magnetic resonance modes in a cylindrical cavity.

dimensional phase space volume $\mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{v}$ at time t is then given by

$$f(\mathbf{x},\mathbf{v},t)\;\mathrm{d}\mathbf{x}\;\mathrm{d}\mathbf{v}. \tag{2.4.1}$$

Under the influence of macroscopic forces in the $(\mathbf{x},\mathbf{v})$ phase space, the flux of particles in the volume element $\mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{v}$ should obey the continuity equation. Consider the rate of flux across the infinitesimal six-dimensional volume element from time t to $t+\mathrm{d}t$,

$$\begin{aligned}\left[f(\mathbf{x},\mathbf{v},t+\mathrm{d}t)-f(\mathbf{x},\mathbf{v},t)\right]\;\mathrm{d}\mathbf{x}\;\mathrm{d}\mathbf{v} = \\ \left[f(\mathbf{x},\mathbf{v},t)\mathbf{v}-f(\mathbf{x}+\mathrm{d}\mathbf{x},\mathbf{v},t)\mathbf{v}\right]\;\mathrm{d}\mathbf{v}\;\mathrm{d}t \\ +\left[f(\mathbf{x},\mathbf{v},t)\mathbf{v}_t(\mathbf{x},\mathbf{v},t)-f(\mathbf{x},\mathbf{v}+\mathrm{d}\mathbf{v},t)\mathbf{v}_t(\mathbf{x},\mathbf{v}+\mathrm{d}\mathbf{v},t)\right]\;\mathrm{d}\mathbf{x}\;\mathrm{d}t,\end{aligned} \tag{2.4.2}$$

where $\mathbf{v}_t$ denotes the particles' acceleration. Division of (2.4.2) by $\mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{v}\,\mathrm{d}t$ yields

$$\frac{\partial f}{\partial t} = -\nabla\left(f\mathbf{v}\right)-\nabla_{\mathbf{v}}\left(f\mathbf{v}_t\right), \tag{2.4.3}$$

where we introduce the operator $\nabla_{\mathbf{v}} = \left(\partial/\partial\mathbf{v}_x, \partial/\partial\mathbf{v}_y, \partial/\partial\mathbf{v}_z\right)^{\top}$. Noting that $\mathbf{v}$ is independent of position $\mathbf{x}$, and that the particles' acceleration $\mathbf{v}_t = \mathbf{F}/m$, where $\mathbf{F}$ is the force and m is the particle mass, has no dependence on $\mathbf{v}$, yields the collisionless Boltzmann equation

$$\frac{\partial f}{\partial t}+\mathbf{v}\cdot\nabla f+\mathbf{v}_t\cdot\nabla_{\mathbf{v}} f = 0. \tag{2.4.4}$$

The Boltzmann equation, accounting for particle collisions, incorporates a further term such that (2.4.4) is modified as follows,

$$\frac{\partial f}{\partial t}+\mathbf{v}\cdot\nabla f+\mathbf{v}_t\cdot\nabla_{\mathbf{v}} f = \left[\frac{\partial f}{\partial t}\right]_{\mathrm{col}}, \tag{2.4.5}$$

where the term $\left[\partial f/\partial t\right]_{\mathrm{col}}$ accounts for the inter-particle collisions occurring almost instantaneously compared with the time scale of the evolution of f.

2.4.3 Electron Particle Conservation

Consider the electron particle density $n_{\mathrm{e}}(\mathbf{x},t)$ given by

$$n_{\mathrm{e}}(\mathbf{x},t) = \int f\;\mathrm{d}\mathbf{v} \tag{2.4.6}$$

and the associated particle flux $\Gamma(\mathbf{x},t)$ for mean particle velocity $\mathbf{u}_{\mathrm{e}}$, i.e.,

$$\Gamma(\mathbf{x},t) = n_{\mathrm{e}}\mathbf{u}_{\mathrm{e}} = \int \mathbf{v} f\;\mathrm{d}\mathbf{v}. \tag{2.4.7}$$

Using these quantities, and by taking the zeroth velocity moment of equation (2.4.5), the macroscopic particle continuity equation of the system is given by

$$\frac{\partial n_{\mathrm{e}}}{\partial t}+\nabla\cdot\left(n_{\mathrm{e}}\mathbf{u}_{\mathrm{e}}\right) = G-L. \tag{2.4.8}$$

The collision term $\left[\frac{\partial f}{\partial t}\right]_{\text{col}}$ yields the particle gain G and loss L terms though processes such as ionisation and recombination. The particle continuity equation does not provide a complete description of the evolution of n_e, since the mean particle velocity $\mathbf{u}_\text{e}$ is unknown.

2.4.4 Particle Momentum Conservation

The plasma electron momentum conservation equation for mean particle velocity $\mathbf{u}_\text{e}$ can be derived from multiplying the Boltzmann equation (2.4.5) by $\mathbf{v}$ and integrating over the particle velocity, i.e., taking the first velocity moment [100] (see also [93, p. 31]),

$$m_\text{e} n_\text{e} \left[\frac{\partial \mathbf{u}_\text{e}}{\partial t} + (\mathbf{u}_\text{e} \cdot \nabla \mathbf{u}_\text{e})\right] + \nabla p = q_\text{e} n_\text{e} \left(\mathcal{E} + \mathbf{u}_\text{e} \times \mathcal{B}\right) + [\mathbf{f}]_\text{col}\,, \tag{2.4.9}$$

where m_e is the electron mass, q_e is the electron charge, p is the pressure, $\mathcal{E}$ is the electric field and $\mathcal{B}$ is the magnetic field.

The collision term in (2.4.9) must encapsulate the momentum transfer due to inter-species collisions. When considering only electrons and positive ions, the Krook collision operator provides a good approximation [100, p. 32]

$$[\mathbf{f}]_\text{col} = -\sum_i m_\text{e} n_\text{e} \nu_{m_\text{e} i} \left(\mathbf{u}_\text{e} - \mathbf{u}_i\right) - m_\text{e} \left(\mathbf{u}_\text{e} - \mathbf{u}_G\right) G + m_\text{e} \left(\mathbf{u}_\text{e} - \mathbf{u}_L\right) L, \tag{2.4.10}$$

where i denotes each species, $\mathbf{u}_i$ is the mean velocity of species i and $\nu_{m_\text{e} i}$ is the momentum transfer frequency for collisions with species i. Here, $\mathbf{u}_G$ and $\mathbf{u}_L$ represent the mean velocities of created and lost particles, respectively.

A simplification of (2.4.9) is made by neglecting magnetic forces and considering only the mass averaged velocity neutral species of the background gas $\mathbf{u}$ and average electron neutral collision frequency with the background gas ν_{m_e}. In general the mean velocity of generated particles by ionisation is much less than the particles' mean velocity $|\mathbf{u}_G| \ll |\mathbf{u}_\text{e}|$ and particles lost through recombination have little effect on their mean velocities $\mathbf{u}_L \approx \mathbf{u}_\text{e}$ [100, p. 32]. This leads to the following simplified model:

$$m_\text{e} n_\text{e} \left[\frac{\partial \mathbf{u}_\text{e}}{\partial t} + \mathbf{u}_\text{e} \cdot \nabla \mathbf{u}_\text{e}\right] = q_\text{e} n_\text{e} \mathcal{E} - \nabla p - m_\text{e} n_\text{e} \nu_{m_\text{e}} \left(\mathbf{u}_\text{e} - \mathbf{u}\right). \tag{2.4.11}$$

Equations (2.4.8) and (2.4.11) can then be closed by choosing a thermodynamic equation of state to eliminate the pressure term p. In the isothermal case, we have $\nabla p = k_B T \nabla n$, where k_B is Boltzmann's constant and T is the temperature. In the adiabatic setting $\nabla \ln p = \gamma \nabla \ln n$, where γ is the ratio of specific heats at constant pressure to constant volume.

2.4.5 Particle Diffusive Mobility

Due to the low density of electrons, we neglect the convective derivative terms and consider the steady state version of (2.4.11), i.e.,

$$q_{\mathrm{e}} n_{\mathrm{e}} \mathcal{E} - \nabla p - m_{\mathrm{e}} n_{\mathrm{e}} \nu_{m_{\mathrm{e}}} \left(\mathbf{u}_{\mathrm{e}} - \mathbf{u} \right) = 0. \tag{2.4.12}$$

Considering an isothermal plasma with the thermodynamic equation of state $\nabla p = k_B T \nabla n_{\mathrm{e}}$ and rearranging for $n_{\mathrm{e}} \mathbf{u}_{\mathrm{e}}$ equation (2.4.12) becomes

$$n_{\mathrm{e}} \mathbf{u}_{\mathrm{e}} = \frac{q_{\mathrm{e}} n_{\mathrm{e}} \mathcal{E}}{m_{\mathrm{e}} \nu_{m_{\mathrm{e}}}} - \frac{k_B T}{m_{\mathrm{e}} \nu_{m_{\mathrm{e}}}} \nabla n_{\mathrm{e}} + n_{\mathrm{e}} \mathbf{u}. \tag{2.4.13}$$

This can be written as the flux term

$$\Gamma = -\mu_{\mathrm{e}} n_{\mathrm{e}} \mathcal{E} - D \nabla n_{\mathrm{e}} + n_{\mathrm{e}} \mathbf{u}, \tag{2.4.14}$$

where the macroscopic mobility μ_{e} and diffusion D coefficients are given, respectively, by

$$\mu_{\mathrm{e}} = \frac{q_{\mathrm{e}}}{m_{\mathrm{e}} \nu_{m_{\mathrm{e}}}}, \quad D = \frac{k_B T}{m_{\mathrm{e}} \nu_{m_{\mathrm{e}}}}. \tag{2.4.15}$$

The flux $\Gamma \equiv n_{\mathrm{e}} \mathbf{u}_{\mathrm{e}}$ can be employed in the particle continuity equation (2.4.8) derived from Fick's law, i.e.,

$$\frac{\partial n_{\mathrm{e}}}{\partial t} + \nabla \cdot \Gamma = G - L. \tag{2.4.16}$$

In the case when the applied electric field is absent, the single species particle diffusion equation follows

$$\frac{\partial n_{\mathrm{e}}}{\partial t} + \nabla \cdot (n_{\mathrm{e}} \mathbf{u}) - \nabla \cdot (D \nabla n_{\mathrm{e}}) = G - L. \tag{2.4.17}$$

2.4.6 Ambipolar Diffusion

The presence of the electric field term in (2.4.14) necessitates that this equation must hold for both electrons and ions. Furthermore, for ionising collisions it should be assumed that the flux of electrons and ions through any volume must be equivalent in order for charge to be conserved, i.e., $\Gamma_e = \Gamma_i = \Gamma$ and $n_{\mathrm{e}} \approx n_i$ for electrons and ions, respectively. Thereby,

$$\mu_{\mathrm{i}} n_{\mathrm{e}} \mathcal{E} - D_{\mathrm{i}} \nabla n_{\mathrm{e}} = -\mu_{\mathrm{e}} n_{\mathrm{e}} \mathcal{E} - D_{\mathrm{e}} \nabla n_{\mathrm{e}}, \tag{2.4.18}$$

which rearranged for $\mathcal{E}$ in terms of ∇n_{e} is

$$\mathcal{E} = \frac{D_{\mathrm{i}} - D_{\mathrm{e}}}{\mu_{\mathrm{i}} + \mu_{\mathrm{e}}} \frac{\nabla n_{\mathrm{e}}}{n_{\mathrm{e}}}. \tag{2.4.19}$$

Substituting this expression for $\mathcal{E}$ into the flux term (2.4.14) and subsequently the continuity equation (2.4.16) gives the ambipolar electron diffusion equation

$$\frac{\partial n_e}{\partial t} + \nabla \cdot (n_e \mathbf{u}) - \nabla \cdot (D_a \nabla n_e) = G - L \tag{2.4.20}$$

for ambipolar diffusion coefficient

$$D_a = \frac{\mu_i D_e + \mu_e D_i}{\mu_i + \mu_e}. \tag{2.4.21}$$

2.4.7 Ionisation and Recombination

Modelling the multitude of ionisation, excitation and recombination reactions of electrons with heavy species is a difficult task. In this thesis, we adopt the heuristic model employed in [131]. The collisions between the electrons and ions in the plasma, especially at low velocity, have a finite probability of resulting in their recombination into a neutral atom. This process is dependent on the collision cross section of electron recombination with the heavy ions σ_{rec}, the kinetic energy of the electron u_e and the number density of the electrons and ions. This results in a loss of electrons in the plasma

$$L = n_e n_i \langle \sigma_{rec} u_e \rangle , \tag{2.4.22}$$

where, $n_i \langle \sigma_{rec} u_e \rangle$ is the average collision frequency of the electrons with the ions. We can write (2.4.22) in terms of a recombination rate coefficient which is a function of the electron temperature T_e and the recombination energy E_{rec}, i.e.,

$$\langle \sigma_{rec} u_e \rangle = k_{rec}(T_e) \sim \sqrt{\frac{E_{rec}}{T_e}}. \tag{2.4.23}$$

Using the assumption that $n_e \approx n_i$ and that k_{rec} is constant, the loss term in equation (2.4.20) is written

$$L = R_0 {n_e}^2 \tag{2.4.24}$$

for recombination constant R_0.

The plasma is maintained in the steady state if the gain of electrons due to ionisation balances the losses due to diffusion and recombination. This process is dependent on the momentum exchange of electrons with the heavy species. In a similar fashion to the electrons lost by recombination, the electrons gained by ionisation is dependent on the average electron-heavy species ionisation cross section σ_i, average electron kinetic energy, the number density of electrons and the neutral heavy species n_n, i.e.,

$$G = n_e n_n \langle \sigma_i u_e \rangle . \tag{2.4.25}$$

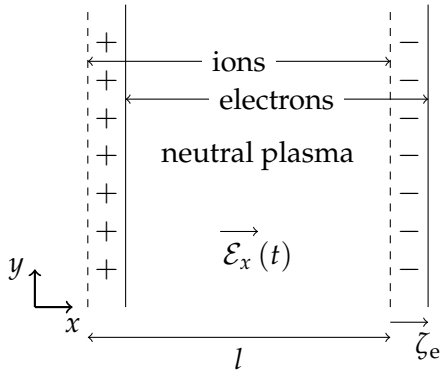


Figure 2.3: Plasma whose electrons are displaced by ζ_e with respect to the ions.

This electron particle source term can also be written in terms of rate coefficient $k_i(T_e)$ and ionisation energy E_i, namely,

$$\langle \sigma_i u_e \rangle = k_i(T_e) \sim A_i e^{-\frac{E_i}{T_e}}, \tag{2.4.26}$$

where, A_i is the ionisation rate constant. We make the simplified assumption here that the electron temperature is strongly influenced by the time averaged magnitude of the applied electric field and the number density of the background gas of neutrals remains roughly constant such that

$$G = A_0 |\mathbf{E}|^2 n_e \tag{2.4.27}$$

for ionisation constant A_0. Regarding appropriate choices of constants R_0 and A_0 we refer to [131].

2.5 Non Magnetised Plasma Properties

2.5.1 Introduction

There is a series of fundamental plasma properties and parameters which are key to the CVD reactor model. We summarise those which are required here. Details of their derivations and analyses can be found in [100].

2.5.2 Natural Plasma Frequency

As shown in Figure 2.3, consider a plasma of finite width l containing equal numbers of stationary $T_e = 0$ electrons and infinite mass ions. Displacing the electrons of this plasma relative to the ions in the positive $\hat{\mathbf{x}}$ direction by a distance $\zeta_e(t) \ll l$ at time t leads to a charge density on the left surface of $\rho_e = en\zeta_e$ and on the right surface

$\rho_e = -en\zeta_e$. Applying Gauss' law (2.3.2), the equal and opposite charges generate an electric field

$$\mathcal{E}_x = \frac{q_e n_e \zeta_e}{\varepsilon_0}, \tag{2.5.1}$$

and the force acting on the electrons is

$$m_e \frac{d^2\zeta_e}{dt^2} = -q_e \mathcal{E}_x. \tag{2.5.2}$$

Substituting (2.5.1) into (2.5.2) reveals a natural oscillatory nature of the electron plasma

$$\frac{d^2\zeta_e}{dt^2} = -\omega_{pe}^2 \zeta_e \tag{2.5.3}$$

for the fundamental characteristic frequency of an electron plasma

$$\omega_{pe}^2 = \frac{{q_e}^2 n_e}{\varepsilon_0 m_e}. \tag{2.5.4}$$

2.5.3 Plasma Permittivity and Conductivity

In the case of a plasma occupying the volume of a background gas in the presence of an applied time harmonic electric field, where the mass of the ions is considered to be infinite, the force acting on the electrons is

$$m_e \frac{d\mathbf{u}_{ex}}{dt} = -q_e \mathcal{E}_x - m_e \nu_{m_e} \mathbf{u}_{ex}. \tag{2.5.5}$$

The solution $\mathbf{u}_{ex}$ in (2.5.5) is also time harmonic with $\mathbf{u}_{ex} = \Re\left\{\hat{\mathbf{u}}_{ex} e^{j\omega t}\right\}$ and amplitude

$$\hat{\mathbf{u}}_{ex} = -\frac{q_e}{m_e} \frac{1}{j\omega + \nu_{m_e}} E_x. \tag{2.5.6}$$

When there is no applied magnetic field, the total current density $\mathbf{i}_T$ can be derived from (2.3.3), i.e.,

$$\mathbf{i}_{Tx} = \varepsilon_0 \frac{\partial \mathcal{E}_x}{\partial t} + \mathbf{i}_x, \tag{2.5.7}$$

where in the cold plasma approximation the current $\mathbf{i}_x$, is due to motion of electrons only. The time harmonic current density $\hat{\mathbf{i}}_x$ is therefore

$$\hat{\mathbf{i}}_x = -q_e n_e \hat{\mathbf{u}}_{ex} \tag{2.5.8}$$

which leads to the time harmonic total current density $\hat{\rho_e}_{Tx}$ given by

$$\hat{\rho_e}_{Tx} = j\omega\varepsilon E_x - q_e n_e \hat{\mathbf{u}}_{ex}. \tag{2.5.9}$$

Substituting $\hat{\mathbf{u}}_{ex}$ in (2.5.9) for the velocity amplitude of equation (2.5.6) gives

$$\hat{\mathbf{i}}_{Tx} = j\omega\varepsilon_0 \left[1 - \frac{\omega_{pe}^2}{\omega\left(\omega - j\nu_{m_e}\right)}\right] E_x. \tag{2.5.10}$$

As a result, this relation of the electric field to the total current density allows the definition of the plasma permittivity ε_p by

$$\varepsilon_p = \varepsilon_0 \left[1 - \frac{\omega_{\text{pe}}^2}{\omega \left(\omega - j\nu_{m_\text{e}} \right)} \right]. \tag{2.5.11}$$

Furthermore, equation (2.5.10) can be written in the form $\hat{\mathbf{i}}_{\text{T}x} = \left(\sigma_p + j\omega\varepsilon_0 \right) E_x$ for plasma conductivity

$$\sigma_p = \frac{\varepsilon_0 \omega_{\text{pe}}^2}{j\omega + \nu_{m_\text{e}}}. \tag{2.5.12}$$

Substituting (2.5.12) into the time harmonic form of Maxwell's equation (2.3.14) gives the new relation

$$\nabla \times \mathbf{H} = \left(\sigma_p + j\omega\varepsilon_0 \right) \mathbf{E}. \tag{2.5.13}$$

Also note that in the low frequency case $\omega \ll \nu_{m_\text{e}}$ and $\omega \ll \omega_{\text{pe}}$, the direct current cold plasma conductivity approximation can be derived

$$\sigma_{\text{dc}} = \frac{\varepsilon_0 \omega_{\text{pe}}^2}{\nu_{m_\text{e}}} = \frac{{q_\text{e}}^2 n_\text{e}}{m_\text{e} \nu_{m_\text{e}}}. \tag{2.5.14}$$

2.5.4 Ohmic Heating

As a result of electron-neutral collisions arising from the electric field in a plasma, the time averaged collisional ohmic power absorbed by those electrons is given by

$$P_{\text{ohm}} = \frac{1}{2} \left| \mathbf{E} \right|^2 \sigma_{\text{dc}} \frac{\nu_{m_\text{e}}^2}{\omega^2 + \nu_{m_\text{e}}^2}, \tag{2.5.15}$$

where $\left| \mathbf{E} \right|^2 = \mathbf{E} \cdot \overline{\mathbf{E}}$. This source of heat is then applied in the conservation of energy of the multicomponent gas mixture heat source term Q of equation (2.2.44). Outside of the CVD reactor's vacuum region in which a plasma cannot be ignited, the standard Joule heating model is applied

$$Q = \sigma \left| \mathbf{E} \right|^2. \tag{2.5.16}$$

2.6 Summary

In this chapter we have derived a fully self consistent model of the MPA-CVD reactor. By this we mean that the solution to the equations derived from the physical conservations laws of momentum, energy, mass and number density, along with the contained electromagnetic field, is to be determined simultaneously. The numerical solution of this system requires the approximation of the unknown quantities of the mass average

gas velocity $\mathbf{u}$, the relative pressure p, the molar mass fraction of atomic hydrogen x_{H}, the system temperature T, the time harmonic electric field $\mathbf{E}$ and the number density of the electrons in the plasma n_{e}. Compound with the difficulty of simultaneously solving the system of equations, the coefficients relating to dissociation of hydrogen are temperature dependent. In this thesis we employ chemical data made available in the National Institute of Standards and Technology chemistry database [35, 43] which approximates empirical results as power series expansions of the temperature variable. A brief summary of the system of equations to be solved and their nonlinear dependence on each of the solution variables is shown in Appendix A.

The model CVD reactor problem and a detailed summary of the conservation equations of this model, along with its numerical approximation will be discussed in detail later in Chapter 4. A primary novelty in the numerical approximation to this model is the ability to account for both effects of macroscopic diffusion and convection, the latter of which is often neglected in the MPA-CVD reactor modelling community.

CHAPTER 3

Discontinuous Galerkin Approximation of Hyperbolic and Elliptic Partial Differential Equations

3.1 Introduction

The MPA-CVD reactor model consists of several constituent equations which in turn are composed of nonlinear diffusive, transport and reaction terms. Prior to presenting the discontinuous Galerkin (DG) finite element discretisation of these terms, a series of definitions fundamental to the DG formulation is given in Section 3.2. In this chapter we introduce a method for classes of elliptic and hyperbolic operators in Sections 3.4 and 3.5, respectively. Furthermore, we examine the treatment of the mass average continuity equation of the quasi-incompressible Navier-Stokes equations in Section 3.6. In a similar fashion, the DG formulation of the Maxwell operator and divergence free electric field condition of Maxwell's equations are also derived in Section 3.7.

3.2 Preliminaries

3.2.1 Function Spaces

Let $\Omega \subset \mathbb{R}^d$, $d \geq 1$, be an open domain with boundary $\partial\Omega$. We define the multi-index tuple of natural numbers

$$\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}^d, \quad |\alpha| = \sum_{j=1}^{d} \alpha_j, \tag{3.2.1}$$

such that the weak derivative operator D^α is defined by

$$D^\alpha := \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}}. \tag{3.2.2}$$

We write $C^s(\Omega)$ to denote the set of all continuous real valued functions in Ω for $s \in \mathbb{N}$ where

$$C^s(\Omega) = \left\{ v \in C(\Omega) : D^\alpha v \in C^0(\Omega),\ |\alpha| \leq s \right\}. \tag{3.2.3}$$

We further denote the space of square integrable functions on Ω as $L_2(\Omega)$ which is equipped with the norm

$$\|v\|_{L_2(\Omega)} := \left(\int_\Omega |v|^2 \ \mathrm{d}\mathbf{x} \right)^{\frac{1}{2}}. \tag{3.2.4}$$

This allows for the definition of the standard Sobolev space

$$H^s(\Omega) := \left\{ u \in L_2(\Omega) \ :\ D^\alpha u \in L_2(\Omega),\ |\alpha| \leq s \right\}, \tag{3.2.5}$$

which is equipped with the seminorm

$$|v|^2_{H^s(\Omega)} = \sum_{|\alpha|=s} \|D^\alpha v\|^2_{L_2(\Omega)} \tag{3.2.6}$$

and norm

$$\|v\|^2_{H^s(\Omega)} = \sum_{i=0}^{s} |v|^2_{H^i(\Omega)}. \tag{3.2.7}$$

For a function space $\mathcal{X}(\Omega)$, we extend the above notation of scalar function spaces to vector and tensor function spaces. To this end, we write $[\mathcal{X}(\Omega)]^d$ and $[\mathcal{X}(\Omega)]^{m\times d}$, $m, d \in \mathbb{N}$, to denote vector and tensor function spaces, respectively. We also define the outer product and tensor contraction operators, respectively, for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$ and $\sigma, \tau \in \mathbb{R}^{m\times d}$, by

$$(\mathbf{u} \otimes \mathbf{v})_{ij} = \mathbf{u}_i \mathbf{v}_j, \quad \sigma : \tau = \mathrm{Trace}\left(\sigma \tau^\top \right). \tag{3.2.8}$$

3.2.2 Discontinuous Function Spaces and Operators

Consider a subdivision of Ω into a shape regular mesh $\mathcal{T}_\Omega^h$, triangulated with non-overlapping elements κ with boundary $\partial\kappa$, each with outward pointing unit normal $\mathbf{n}_\kappa$, such that $\mathcal{T}_\Omega^h = \{\kappa\}$ and $\overline{\Omega} = \bigcup_{\kappa\in\mathcal{T}_\Omega^h} \overline{\kappa}$. We define the interior faces of the mesh by $\Gamma_\mathcal{I} = \bigcup_{\kappa\in\mathcal{T}_\Omega^h} \partial\kappa \setminus \partial\Omega$. We denote the space of L_2 functions on Ω whose restriction to each element κ belongs to $H^s(\kappa)$ as the 'broken' Sobolev space,

$$H^s(\mathcal{T}_\Omega^h) = \{v \in L_2(\Omega) : v|_\kappa \in H^s(\kappa),\ \kappa \in \mathcal{T}_\Omega^h\}. \tag{3.2.9}$$

In addition, the broken gradient of a function $q \in H^1(\mathcal{T}_\Omega^h)$ is defined by

$$(\nabla_h q)|_\kappa := \nabla\left(q|_\kappa\right),\ \ \kappa \in \mathcal{T}_\Omega^h. \tag{3.2.10}$$

Similarly, the broken divergence of a function $\mathbf{v} \in \left[H^1(\mathcal{T}_\Omega^h)\right]^d$ is given by

$$(\nabla_h \cdot \mathbf{v})|_\kappa := \nabla \cdot (\mathbf{v}|_\kappa),\ \ \kappa \in \mathcal{T}_\Omega^h, \tag{3.2.11}$$

and the broken curl

$$(\nabla_h \times \mathbf{v})|_\kappa := \nabla \times (\mathbf{v}|_\kappa),\ \ \kappa \in \mathcal{T}_\Omega^h. \tag{3.2.12}$$

For a given polynomial of order $\ell \geq 0$, we denote the space of polynomials on each element by

$$\mathcal{P}^\ell(\kappa) := \{v : v \text{ is a polynomial of degree} \leq \ell \text{ on } \kappa\} \tag{3.2.13}$$

and the finite element space of discontinuous vector valued polynomial functions of degree $\ell \geq 0$ and dimension m

$$\mathbf{V}_\ell^m(\mathcal{T}_\Omega^h) := \left\{\mathbf{v}_h \in [L_2(\Omega)]^m : \mathbf{v}_h|_\kappa \in \left[\mathcal{P}^\ell(\kappa)\right]^m, \kappa \in \mathcal{T}_\Omega^h\right\}. \tag{3.2.14}$$

For convenience we further denote the finite element space consisting of discontinuous scalar polynomial functions by $V_\ell(\mathcal{T}_\Omega^h) := \mathbf{V}_\ell^1(\mathcal{T}_\Omega^h)$. Similarly, the finite element space consisting of discontinuous tensor valued polynomials of degree $\ell \geq 0$ and dimension $m \times d$ is given by

$$\Sigma_\ell^{m\times d}(\mathcal{T}_\Omega^h) := \left\{\tau \in \left[L^2(\Omega)\right]^{m\times d} : \tau|_\kappa \in \left[\mathcal{P}^\ell(\kappa)\right]^{m\times d},\ \kappa \in \mathcal{T}_\Omega^h\right\}. \tag{3.2.15}$$

3.2.3 Trace Operators

Given two neighbouring elements in the mesh which share a common face $F \in \Gamma_\mathcal{I}$, denoted $\kappa^+, \kappa^- \in \mathcal{T}_\Omega^h$, such that $F = \partial\kappa^+ \cap \partial\kappa^-$, we write the traces of scalar $q \in H^1(\mathcal{T}_\Omega^h)$, vector $\mathbf{w} \in \left[H^1(\mathcal{T}_\Omega^h)\right]^d$ and tensor $\tau \in \left[H^1(\mathcal{T}_\Omega^h)\right]^{m\times d}$ functions on F as $q^\pm$, $\mathbf{w}^\pm$

and $\tau^{\pm}$, respectively (relative to the interior of element $\kappa^{\pm}$). Similarly, we also write the unit outward normal vector of F pointing from κ^+ into κ^- as $\mathbf{n}_\kappa^+$ and from κ^- into κ^+ as $\mathbf{n}_\kappa^-$. Using this notation we define the average, jump, tensor-jump and tangential-jump operators. To this end, the average operator $\{\!\{\cdot\}\!\}$ is defined as

$$\{\!\{q\}\!\} = \frac{1}{2}\left(q^+ + q^-\right) \quad \text{on } \Gamma_{\mathcal{I}}, \qquad \{\!\{q\}\!\} = q^+ \quad \text{on } \partial\Omega, \tag{3.2.16}$$

$$\{\!\{\mathbf{w}\}\!\} = \frac{1}{2}\left(\mathbf{w}^+ + \mathbf{w}^-\right) \quad \text{on } \Gamma_{\mathcal{I}}, \qquad \{\!\{\mathbf{w}\}\!\} = \mathbf{w}^+ \quad \text{on } \partial\Omega, \tag{3.2.17}$$

$$\{\!\{\tau\}\!\} = \frac{1}{2}\left(\tau^+ + \tau^-\right) \quad \text{on } \Gamma_{\mathcal{I}}, \qquad \{\!\{\tau\}\!\} = \tau^+ \quad \text{on } \partial\Omega, \tag{3.2.18}$$

the jump operator $[\![\cdot]\!]$ is defined as

$$[\![q]\!] = q^+\mathbf{n}_\kappa^+ + q^-\mathbf{n}_\kappa^- \quad \text{on } \Gamma_{\mathcal{I}}, \qquad [\![q]\!] = q^+\mathbf{n}_\kappa^+ \quad \text{on } \partial\Omega, \tag{3.2.19}$$

$$[\![\mathbf{w}]\!] = \mathbf{w}^+ \cdot \mathbf{n}_\kappa^+ + \mathbf{w}^- \cdot \mathbf{n}_\kappa^- \quad \text{on } \Gamma_{\mathcal{I}}, \qquad [\![\mathbf{w}]\!] = \mathbf{w}^+ \cdot \mathbf{n}_\kappa^+ \quad \text{on } \partial\Omega, \tag{3.2.20}$$

$$[\![\tau]\!] = \tau^+\mathbf{n}_\kappa^+ + \tau^-\mathbf{n}_\kappa^- \quad \text{on } \Gamma_{\mathcal{I}}, \qquad [\![\tau]\!] = \tau^+\mathbf{n}_\kappa^+ \quad \text{on } \partial\Omega, \tag{3.2.21}$$

the tensor-jump operator $\underline{[\![\cdot]\!]}$ is given by

$$\underline{[\![\mathbf{w}]\!]} = \mathbf{w}^+ \otimes \mathbf{n}_\kappa^+ + \mathbf{w}^- \otimes \mathbf{n}_\kappa^- \quad \text{on } \Gamma_{\mathcal{I}}, \quad \underline{[\![\mathbf{w}]\!]} = \mathbf{w}^+ \otimes \mathbf{n}_\kappa^+ \quad \text{on } \partial\Omega, \tag{3.2.22}$$

and the tangential-jump operator $[\![\cdot]\!]_{\mathrm{T}}$ is defined by

$$[\![\mathbf{w}]\!]_{\mathrm{T}} = \mathbf{n}_\kappa^+ \times \mathbf{w}^+ + \mathbf{n}_\kappa^- \times \mathbf{w}^- \quad \text{on } \Gamma_{\mathcal{I}}, \qquad [\![\mathbf{w}]\!]_{\mathrm{T}} = \mathbf{n}_\kappa^+ \times \mathbf{w}^+ \quad \text{on } \partial\Omega. \tag{3.2.23}$$

It is worth noting that:

$$\{\!\{\{\!\{q\}\!\}\}\!\} = \{\!\{q\}\!\}, \qquad \{\!\{\{\!\{\mathbf{w}\}\!\}\}\!\} = \{\!\{\mathbf{w}\}\!\}, \qquad \{\!\{\{\!\{\tau\}\!\}\}\!\} = \{\!\{\tau\}\!\}, \tag{3.2.24}$$

$$\{\!\{[\![q]\!]\}\!\} = [\![q]\!], \qquad \{\!\{[\![\mathbf{w}]\!]\}\!\} = [\![\mathbf{w}]\!], \qquad \{\!\{[\![\tau]\!]\}\!\} = [\![\tau]\!], \tag{3.2.25}$$

$$[\![[\![q]\!]]\!] = 0, \qquad [\![[\![\mathbf{w}]\!]]\!] = \mathbf{0}, \qquad [\![[\![\tau]\!]]\!] = 0, \tag{3.2.26}$$

$$[\![\{\!\{q\}\!\}]\!] = \mathbf{0}, \qquad [\![\{\!\{\mathbf{w}\}\!\}]\!] = 0, \qquad [\![\{\!\{\tau\}\!\}]\!] = \mathbf{0}. \tag{3.2.27}$$

Each of these operators allow us to rewrite integrals over element boundaries $\partial\kappa$ as integrals over the interior $\Gamma_{\mathcal{I}}$ and exterior $\partial\Omega$ element faces, namely,

$$\sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa} q\mathbf{w}\cdot\mathbf{n}_\kappa \,\mathrm{d}s = \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} [\![q]\!]\cdot\{\!\{\mathbf{w}\}\!\} \,\mathrm{d}s + \int_{\Gamma_{\mathcal{I}}} \{\!\{q\}\!\}[\![\mathbf{w}]\!] \,\mathrm{d}s, \tag{3.2.28}$$

$$\sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa} \tau : (\mathbf{z}\otimes\mathbf{n}_\kappa) \,\mathrm{d}s = \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} \underline{[\![\mathbf{z}]\!]} : \{\!\{\tau\}\!\} \,\mathrm{d}s + \int_{\Gamma_{\mathcal{I}}} \{\!\{\mathbf{z}\}\!\}\cdot[\![\tau]\!] \,\mathrm{d}s, \tag{3.2.29}$$

$$\sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa} \mathbf{v}\cdot(\mathbf{n}_\kappa\times\mathbf{w}) \,\mathrm{d}s = \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} [\![\mathbf{w}]\!]_{\mathrm{T}}\cdot\{\!\{\mathbf{v}\}\!\} \,\mathrm{d}s - \int_{\Gamma_{\mathcal{I}}} \{\!\{\mathbf{w}\}\!\}\cdot[\![\mathbf{v}]\!]_{\mathrm{T}} \,\mathrm{d}s, \tag{3.2.30}$$

where $q \in H^1\left(\mathcal{T}_\Omega^h\right)$, $\mathbf{w} \in \left[H^1\left(\mathcal{T}_\Omega^h\right)\right]^d$, $\mathbf{v} \in \left[H^1\left(\mathcal{T}_\Omega^h\right)\right]^d$, $\mathbf{z} \in \left[H^1\left(\mathcal{T}_\Omega^h\right)\right]^m$ and $\tau \in \left[H^1\left(\mathcal{T}_\Omega^h\right)\right]^{m\times d}$. Proofs of (3.2.28), (3.2.29) and (3.2.30) are given in Appendix B.

3.3 Conservation Laws

Given $\Omega \subset \mathbb{R}^d$, $d \geq 1$, with boundary $\partial\Omega$, we write $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$, where $\partial\Omega_D$ is closed and non-empty. A typical model conservation law is to find $\mathbf{u}$ such that

$$\nabla \cdot (\mathcal{F}^c(\mathbf{u}) - \mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})) = \mathbf{0}, \qquad \text{in } \Omega, \tag{3.3.1}$$

$$\mathbf{u} = \mathbf{g}_D, \qquad \text{on } \partial\Omega_D, \tag{3.3.2}$$

$$\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) \cdot \mathbf{n} = \mathbf{g}_N, \qquad \text{on } \partial\Omega_N, \tag{3.3.3}$$

where $\mathbf{n}$ denotes the unit outward normal vector on $\partial\Omega$. Here, $\mathbf{u}$ is the conserved solution vector,

$$\mathcal{F}^c(\mathbf{u}) = \begin{pmatrix} f^c_{1,1}(\mathbf{u}) & \cdots & f^c_{1,d}(\mathbf{u}) \\ \vdots & \ddots & \vdots \\ f^c_{m,1}(\mathbf{u}) & \cdots & f^c_{m,d}(\mathbf{u}) \end{pmatrix} \equiv \left(\mathbf{f}^c_1(\mathbf{u}), \ldots, \mathbf{f}^c_d(\mathbf{u})\right) \tag{3.3.4}$$

is the nonlinear hyperbolic convective flux and

$$\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) = \begin{pmatrix} f^v_{1,1}(\mathbf{u}; \nabla\mathbf{u}) & \cdots & f^v_{1,d}(\mathbf{u}; \nabla\mathbf{u}) \\ \vdots & \ddots & \vdots \\ f^v_{m,1}(\mathbf{u}; \nabla\mathbf{u}) & \cdots & f^v_{m,d}(\mathbf{u}; \nabla\mathbf{u}) \end{pmatrix} \equiv \left(\mathbf{f}^v_1(\mathbf{u}; \nabla\mathbf{u}), \ldots, \mathbf{f}^v_d(\mathbf{u}; \nabla\mathbf{u})\right) \tag{3.3.5}$$

is the viscous flux, which is assumed here to be nonlinear in $\mathbf{u}$ and linear in $\nabla\mathbf{u}$. Furthermore, $\mathbf{g}_D$ and $\mathbf{g}_N$ are the Dirichlet and Neumann boundary functions, respectively. A typical example includes the compressible Navier-Stokes equations [66].

3.4 DG Finite Element Formulation of Hyperbolic Terms

The DG formulation of the convective terms arising in the PDE (3.3.1) presented here is based on the work undertaken by Hartmann and Houston [65]. We present a specific case where $\partial\Omega_N = \emptyset$ assuming that Dirichlet data is prescribed on $\partial\Omega_D$ as required by (3.3.1), and refer to [65] regarding the more general case. Consider the convective term $\nabla \cdot \mathcal{F}^c(\mathbf{u})$ of (3.3.1) which we rewrite as

$$\nabla \cdot \mathcal{F}^c(\mathbf{u}) = \mathbf{0}, \qquad \text{in } \Omega, \tag{3.4.1}$$

$$B^-(\mathbf{u}, \mathbf{n})(\mathbf{u} - \mathbf{g}_D) = \mathbf{0}, \qquad \text{on } \partial\Omega_D. \tag{3.4.2}$$

Here,

$$B(\mathbf{u}, \mathbf{n}) := \sum_{i=1}^{d} \frac{\partial \mathbf{f}^c_i}{\partial \mathbf{u}} \mathbf{n}_i \tag{3.4.3}$$

is the flux Jacobian, where $\mathbf{n}_i$ is the ith component of $\mathbf{n}$ and the positive/negative part of $B(\mathbf{u},\mathbf{n})$ is given by

$$B^{\pm}(\mathbf{u},\mathbf{n}) = P\Lambda^{\pm}P^{-1}. \tag{3.4.4}$$

Here, P is the matrix of eigenvectors of $B(\mathbf{u},\mathbf{n})$ and the diagonal matrices of the positive and negative eigenvalues $\lambda(B)$ are written

$$\Lambda^{+} = \operatorname{diag}\left(\max\left(\lambda(B),0\right)\right), \tag{3.4.5}$$

$$\Lambda^{-} = \operatorname{diag}\left(\min\left(\lambda(B),0\right)\right). \tag{3.4.6}$$

Multiplying (3.4.1) by a test function $\mathbf{v} \in [H^1(\mathcal{T}_\Omega^h)]^m$ and integrating by parts on an element $\kappa \in \mathcal{T}_\Omega^h$ yields

$$-\int_\kappa \mathcal{F}^c(\mathbf{u}) : \nabla\mathbf{v}\,\mathrm{d}\mathbf{x} + \int_{\partial\kappa} \mathcal{F}^c(\mathbf{u})\cdot\mathbf{n}_\kappa\cdot\mathbf{v}\,\mathrm{d}\mathbf{x} = \mathbf{0}. \tag{3.4.7}$$

The DG finite element formulation is derived by replacing $\mathbf{u}$ by the finite element approximation $\mathbf{u}_h$ and the test function $\mathbf{v}$ by $\mathbf{v}_h$, where $\mathbf{u}_h, \mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$. Summing (3.4.7) over all elements $\kappa \in \mathcal{T}_\Omega^h$ and replacing the inter-element convective flux with a consistent and conservative numerical flux function $\mathcal{H}(\mathbf{u}_h^+,\mathbf{u}_h^-,\mathbf{n}_\kappa)$, the DG semilinear formulation is given by: find $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ such that

$$\begin{aligned}\mathcal{N}_\Omega^c(\mathbf{u}_h;\mathbf{v}_h) := &-\int_\Omega \mathcal{F}^c(\mathbf{u}_h) : \nabla_h\mathbf{v}_h\,\mathrm{d}\mathbf{x}\\ &+\sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa\setminus\partial\Omega}\mathcal{H}\left(\mathbf{u}_h^+,\mathbf{u}_h^-,\mathbf{n}_\kappa\right)\cdot\mathbf{v}_h^+\,\mathrm{d}s\\ &+\sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa\cap\partial\Omega}\mathcal{H}\left(\mathbf{u}_h^+,\mathbf{u}_\Gamma\left(\mathbf{u}_h^+\right),\mathbf{n}_\kappa\right)\cdot\mathbf{v}_h^+\,\mathrm{d}s = 0\end{aligned} \tag{3.4.8}$$

for all $\mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$.

Several numerical fluxes $\mathcal{H}(\cdot,\cdot,\cdot)$, such as the Roe flux [121] and the Vijayasundaram flux [139], are discussed in the application of numerical schemes for hyperbolic conservation laws, see LeVeque [96] and Kröner [94]. The boundary function $\mathbf{u}_\Gamma(\mathbf{u}_h^+)$ determines the weakly enforced boundary conditions. Imposing the Dirichlet boundary condition on $\partial\Omega_D$ requires that $\mathbf{u}_\Gamma(\mathbf{u}^+) = \mathbf{g}_D$ on $\partial\Omega_D$, cf. Hartmann and Houston [65].

In this thesis we employ the consistent and conservative local-Lax Friedrichs flux

$$\mathcal{H}_{\mathrm{LF}}\left(\mathbf{u}_h^+,\mathbf{u}_h^-,\mathbf{n}_\kappa\right)\big|_{\partial\kappa} := \tfrac{1}{2}\left(\mathcal{F}^c\left(\mathbf{u}_h^+\right)\cdot\mathbf{n}_\kappa + \mathcal{F}^c\left(\mathbf{u}_h^-\right)\cdot\mathbf{n}_\kappa + \alpha\left(\mathbf{u}_h^+ - \mathbf{u}_h^-\right)\right). \tag{3.4.9}$$

The dissipation parameter α is defined by

$$\alpha|_{\partial\kappa} = \max_{\mathbf{w}=\mathbf{u}_h^+,\mathbf{u}_h^-}\left\{\left|\lambda\left(B\left(\mathbf{w},\mathbf{n}_\kappa\right)\right)\right|\right\}. \tag{3.4.10}$$

3.5 DG Finite Element Formulation of Elliptic Terms

The DG discretisation of the viscous terms arising in the PDE (3.3.1) presented here is again based on the work undertaken by Hartmann and Houston [66]. Consider now the viscous term $-\nabla \cdot \mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u})$ of (3.3.1) which we rewrite as

$$-\nabla \cdot \mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u}) = \mathbf{0}, \qquad \text{in } \Omega, \tag{3.5.1}$$

$$\mathbf{u} = \mathbf{g}_D, \qquad \text{on } \partial\Omega_D, \tag{3.5.2}$$

$$\mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u}) \cdot \mathbf{n} = \mathbf{g}_N, \qquad \text{on } \partial\Omega_N. \tag{3.5.3}$$

We define the homogeneity tensor by

$$G_{kl}(\mathbf{u}) = \frac{\partial \mathbf{f}_k^v}{\partial (\nabla \mathbf{u})_l}, \quad k, l = 1, \ldots, d; \tag{3.5.4}$$

thereby, we may write

$$(G(\mathbf{u})\nabla \mathbf{u})_{ik} = \sum_{j=1}^{m} \sum_{l=1}^{d} (G_{kl}(\mathbf{u}))_{ij} (\nabla \mathbf{u})_{jl}. \tag{3.5.5}$$

Here, we also define the transpose homogeneity tensor product acting on a tensor variable $\tau \in \mathbb{R}^{m \times d}$

$$\left(G^\top(\mathbf{u})\,\tau\right)_{jl} = \sum_{i=1}^{m} \sum_{k=1}^{d} (G_{kl}(\mathbf{u}))_{ij}\, \tau_{ik}. \tag{3.5.6}$$

In order to define the DG formulation of (3.5.1) we rewrite (3.5.1) as a first order system, i.e., we have

$$\sigma = G(\mathbf{u})\nabla \mathbf{u} \equiv \mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u}) \quad \text{and} \quad -\nabla \cdot \sigma = 0. \tag{3.5.7}$$

Multiplying both parts of (3.5.7) by test functions $\tau \in \left[H^1(\mathcal{T}_\Omega^h)\right]^{m \times d}$ and $\mathbf{v} \in \left[H^1(\mathcal{T}_\Omega^h)\right]^m$, respectively, and integrating by parts on each element $\kappa \in \mathcal{T}_\Omega^h$ gives

$$\int_\kappa \sigma : \tau \, \mathrm{d}\mathbf{x} = -\int_\kappa \mathbf{u} \cdot \nabla \cdot \left(G^\top(\mathbf{u})\,\tau\right) \mathrm{d}\mathbf{x} + \int_{\partial\kappa} \mathbf{u} \cdot \left(G^\top(\mathbf{u})\,\tau\right) \cdot \mathbf{n}_\kappa \, \mathrm{d}s, \tag{3.5.8}$$

$$\int_\kappa \sigma : \nabla \mathbf{v} \, d\mathbf{x} = \int_{\partial\kappa} \sigma \cdot \mathbf{n}_\kappa \cdot \mathbf{v} \, \mathrm{d}s. \tag{3.5.9}$$

Here, as above, $\mathbf{n}_\kappa$ denotes the unit outward normal vector on the boundary of element $\kappa \in \mathcal{T}_\Omega^h$. Here, we have employed (3.5.5) and (3.5.6) in (3.5.8); indeed, we have that

$$\begin{aligned}\int_\kappa \sigma : \tau \, \mathrm{d}\mathbf{x} &= \int_\kappa \sum_{i=1}^{m}\sum_{k=1}^{d} \sigma_{ik}\tau_{ik} \, \mathrm{d}\mathbf{x} \\ &= \int_\kappa \sum_{i=1}^{m}\sum_{k=1}^{d} \left(\sum_{j=1}^{m}\sum_{l=1}^{d} \left(G(\mathbf{u})_{kl}\right)_{ij} \left(\nabla \mathbf{u}\right)_{jl} \right) \tau_{ik} \, \mathrm{d}\mathbf{x} \\ &= \int_\kappa \sum_{j=1}^{m}\sum_{l=1}^{d} \left(\nabla \mathbf{u}\right)_{jl} \left(\sum_{i=1}^{m}\sum_{k=1}^{d} \left(G(\mathbf{u})_{kl}\right)_{ij} \tau_{ik} \right) \, \mathrm{d}\mathbf{x} \\ &= \int_\kappa \sum_{j=1}^{m}\sum_{l=1}^{d} \left(\nabla \mathbf{u}\right)_{jl} \left(G(\mathbf{u})^\top \tau \right)_{jl} \, \mathrm{d}\mathbf{x} \\ &= \int_\kappa \nabla \mathbf{u} : \left(G(\mathbf{u})^\top \tau \right) \, \mathrm{d}\mathbf{x}. \end{aligned} \tag{3.5.10}$$

We sum over all elements $\kappa \in \mathcal{T}_\Omega^h$ and replace $\mathbf{u}$, $\mathbf{v}$, σ and τ by their discrete finite element counterparts, $\mathbf{u}_h, \mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ and $\sigma_h, \tau_h \in \Sigma_\ell^{m\times d}(\mathcal{T}_\Omega^h)$. This yields the flux formulation: find $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ and $\sigma_h \in \Sigma_\ell^{m\times d}(\mathcal{T}_\Omega^h)$ such that, respectively

$$\int_\Omega \sigma_h : \tau_h \, \mathrm{d}\mathbf{x} = -\int_\Omega \mathbf{u}_h \cdot \nabla_h \cdot \left(G^\top\left(\mathbf{u}_h\right) \tau_h \right) \, \mathrm{d}\mathbf{x} + \sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa} \hat{\mathbf{u}}_h \cdot \left(G^\top\left(\mathbf{u}_h\right) \tau_h \right) \cdot \mathbf{n}_\kappa \, \mathrm{d}s, \tag{3.5.11}$$

$$\int_\Omega \sigma_h : \nabla_h \mathbf{v}_h \, d\mathbf{x} = \sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa\setminus\partial\Omega_N} \hat{\sigma}_h \cdot \mathbf{n}_\kappa \cdot \mathbf{v}_h \, \mathrm{d}s + \sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa\cap\partial\Omega_N} \mathbf{g}_N \cdot \mathbf{v}_h \, \mathrm{d}s \tag{3.5.12}$$

for all $\mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ and $\tau \in \Sigma_\ell^{m\times d}(\mathcal{T}_\Omega^h)$.

The numerical fluxes $\hat{\mathbf{u}}_h$ and $\hat{\sigma}_h$ represent approximations to $\mathbf{u}$ and $\nabla\mathbf{u}$, respectively. The formulation of the DG method will depend on the particular choice of these numerical fluxes which will be addressed later. To determine the primal formulation of equations (3.5.11) and (3.5.12) in terms of the variable $\mathbf{u}_h$, we perform integration by parts in (3.5.11) on all elements $\kappa \in \mathcal{T}_\Omega^h$ and choose the test function $\tau_h = \nabla_h \mathbf{v}_h$; thereby, we get

$$\begin{aligned}\int_\Omega \sigma_h : \nabla_h \mathbf{v}_h \, \mathrm{d}\mathbf{x} = \int_\Omega \mathcal{F}^v\left(\mathbf{u}_h; \nabla_h \mathbf{u}_h\right) : \nabla_h \mathbf{v}_h \, \mathrm{d}\mathbf{x} \\ + \sum_{\kappa\in\mathcal{T}_\Omega^h} \int_{\partial\kappa} \left(\hat{\mathbf{u}}_h - \mathbf{u}_h\right) \cdot \left(G^\top(\mathbf{u}_h) \nabla_h \mathbf{v}_h \right) \cdot \mathbf{n}_\kappa \, \mathrm{d}s. \end{aligned} \tag{3.5.13}$$

Upon substituting equation (3.5.13) into equation (3.5.12), the primal semilinear form can be obtained for the viscous components, namely: find $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$, such that

$$\begin{aligned}\mathcal{N}_\Omega^v(\mathbf{u}_h;\mathbf{v}_h) := &\int_\Omega \mathcal{F}^v(\mathbf{u}_h;\nabla_h\mathbf{u}_h) : \nabla_h\mathbf{v}_h \,\mathrm{d}\mathbf{x} \\ &+ \sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa}(\hat{\mathbf{u}}_h - \mathbf{u}_h)\cdot\left(G^\top(\mathbf{u}_h)\nabla_h\mathbf{v}_h\right)\cdot\mathbf{n}_\kappa \,\mathrm{d}s \\ &- \sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa\setminus\partial\Omega_N}\hat{\sigma}_h\cdot\mathbf{n}_\kappa\cdot\mathbf{v}_h \,\mathrm{d}s - \sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa\cap\partial\Omega_N}\mathbf{g}_N\cdot\mathbf{v}_h \,\mathrm{d}s = 0 \quad (3.5.14)\end{aligned}$$

for all $\mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$.

It should be noted that the face terms which arise on the interior of the mesh $\mathcal{T}_\Omega^h$ occur twice in the sum over the elements $\kappa \in \mathcal{T}_\Omega^h$ in equation (3.5.14). In order to rewrite the primal flux formulation in terms of a face-based rather than an element-based form, we apply the identities stated in equations (3.2.28) and (3.2.29). The DG residual primal flux formulation is to find $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ such that

$$\begin{aligned}\mathcal{N}_\Omega^v(\mathbf{u}_h;\mathbf{v}_h) := &\int_\Omega \mathcal{F}^v(\mathbf{u}_h;\nabla_h\mathbf{u}_h) : \nabla_h\mathbf{v}_h \,\mathrm{d}\mathbf{x} - \int_{\Gamma_\mathcal{I}\cup\partial\Omega}\{\!\{\hat{\sigma}_h\}\!\} : \underline{[\![\mathbf{v}_h]\!]} \,\mathrm{d}s \\ &- \int_{\Gamma_\mathcal{I}}[\![\hat{\sigma}_h]\!]\cdot\{\!\{\mathbf{v}_h\}\!\} \,\mathrm{d}s + \int_{\Gamma_\mathcal{I}\cup\partial\Omega}\underline{[\![\hat{\mathbf{u}}_h - \mathbf{u}_h]\!]} : \{\!\{G^\top(\mathbf{u}_h)\nabla_h\mathbf{v}_h\}\!\} \,\mathrm{d}s \\ &+ \int_{\Gamma_\mathcal{I}}\{\!\{\hat{\mathbf{u}}_h - \mathbf{u}_h\}\!\}\cdot[\![G^\top(\mathbf{u}_h)\nabla_h\mathbf{v}_h]\!] \,\mathrm{d}s - \int_{\partial\Omega_N}\mathbf{g}_N\cdot\mathbf{v}_h \,\mathrm{d}s = 0 \quad (3.5.15)\end{aligned}$$

for all $\mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$, where the choices of numerical flux vector function $\hat{\mathbf{u}}_h$ and tensor function $\hat{\sigma}_h$ are discussed at length in [8].

Here we employ the interior penalty method; to this end, the numerical flux functions are defined by

$$\hat{\mathbf{u}}_h = \{\!\{\mathbf{u}_h\}\!\}, \quad \hat{\sigma}_h = \{\!\{\mathcal{F}^v(\mathbf{u}_h;\nabla_h\mathbf{u}_h)\}\!\} - \delta(\mathbf{u}_h) \quad \text{on } \Gamma_\mathcal{I}, \tag{3.5.16}$$

where the penalisation term $\delta(\mathbf{u}_h)$ for the interior penalty method is chosen to be

$$\delta(\mathbf{u}_h) = C_{\mathrm{IP}}\frac{\ell^2}{h_F}\{\!\{G(\mathbf{u}_h)\}\!\}\underline{[\![\mathbf{u}_h]\!]}, \tag{3.5.17}$$

where C_{IP} is a sufficiently large positive constant and h_F is defined by

$$h_F|_F = \min(\mathrm{meas}(\kappa^+), \mathrm{meas}(\kappa^-))/\mathrm{meas}(F), \quad F = \partial\kappa^+ \cap \partial\kappa^-. \tag{3.5.18}$$

On the exterior boundary we select the numerical flux functions as follows

$$\hat{\mathbf{u}}_h = \mathbf{u}_\Gamma(\mathbf{u}_h), \quad \hat{\sigma}_h = \{\!\{\mathcal{F}^v(\mathbf{u}_\Gamma(\mathbf{u}_h);\nabla_h\mathbf{u}_h)\}\!\} - \delta_\Gamma(\mathbf{u}_h) \quad \text{on } \partial\Omega, \tag{3.5.19}$$

where the penalisation term $\delta_\Gamma(\mathbf{u}_h)$ on the exterior boundary $\partial\Omega_D$ is given by

$$\delta_\Gamma(\mathbf{u}_h) = C_{\text{IP}} \frac{\ell^2}{h_F} \{\!\{ G(\mathbf{u}_\Gamma(\mathbf{u}_h)) \}\!\} \underline{[\![\mathbf{u}_h - \mathbf{u}_\Gamma(\mathbf{u}_h)]\!]}. \tag{3.5.20}$$

Here, the boundary function $\mathbf{u}_\Gamma(\mathbf{u}_h)$ weakly enforces the Dirichlet boundary condition. In the case of the original conservation model equation (3.3.1), $\mathbf{u}_\Gamma(\mathbf{u}_h)|_{\partial\Omega_D} = \mathbf{g}_D$; on the Neumann boundary we set $\mathbf{u}_\Gamma(\mathbf{u}_h)|_{\partial\Omega_N} = \mathbf{u}_h^+$.

3.5.1 Complete Formulation for Convective and Viscous Terms

Collecting the DG discretisations of the convective and viscous terms of equation (3.3.1), the full DG discretisation may be defined by: find $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$ such that

$$\begin{aligned}
\mathcal{N}_\Omega^{c,v}(\mathbf{u}_h; \mathbf{v}_h) := &- \int_\Omega \mathcal{F}^c(\mathbf{u}_h) : \nabla_h \mathbf{v}_h \, \mathrm{d}\mathbf{x} + \int_\Omega \mathcal{F}^v(\mathbf{u}_h; \nabla_h \mathbf{u}_h) : \nabla_h \mathbf{v}_h \, \mathrm{d}\mathbf{x} \\
&+ \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \setminus \partial\Omega} \mathcal{H}(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa) \cdot \mathbf{v}_h^+ \, \mathrm{d}s - \int_{\Gamma_\mathcal{I}} \underline{[\![\mathbf{u}_h]\!]} : \{\!\{ G^\top(\mathbf{u}_h) \nabla_h \mathbf{v}_h \}\!\} \, \mathrm{d}s \\
&- \int_{\Gamma_\mathcal{I}} \{\!\{ \mathcal{F}^v(\mathbf{u}_h; \nabla_h \mathbf{u}_h) \}\!\} : \underline{[\![\mathbf{v}_h]\!]} \, \mathrm{d}s + \int_{\Gamma_\mathcal{I}} \delta(\mathbf{u}_h) : \underline{[\![\mathbf{v}_h]\!]} \, \mathrm{d}s \\
&+ \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \cap \partial\Omega} \mathcal{H}\left(\mathbf{u}_h^+, \mathbf{u}_\Gamma\left(\mathbf{u}_h^+\right), \mathbf{n}_\kappa\right) \cdot \mathbf{v}_h^+ \, \mathrm{d}s - \int_{\partial\Omega_D} \{\!\{ \mathcal{F}^v\left(\mathbf{u}_\Gamma(\mathbf{u}_h^+); \nabla_h \mathbf{u}_h\right) \}\!\} : \underline{[\![\mathbf{v}_h]\!]} \, \mathrm{d}s \\
&- \int_{\partial\Omega_D} \underline{[\![\mathbf{u}_h - \mathbf{u}_\Gamma(\mathbf{u}_h^+)]\!]} : \{\!\{ G^\top(\mathbf{u}_\Gamma(\mathbf{u}_h)) \nabla_h \mathbf{v}_h \}\!\} \, \mathrm{d}s + \int_{\partial\Omega_D} \delta_\Gamma(\mathbf{u}_h^+) : \underline{[\![\mathbf{v}_h]\!]} \, \mathrm{d}s \\
&- \int_{\partial\Omega_N} \mathbf{g}_N \cdot \mathbf{v}_h^+ \, \mathrm{d}s = 0
\end{aligned} \tag{3.5.21}$$

for all $\mathbf{v}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$.

3.6 The Quasi-Incompressible Navier-Stokes Continuity Equation

The work of Cockburn and co-workers analyses the application of the DG finite element method to the Stokes [39], Oseen [41] and Navier-Stokes [40] equations subject to the constraint of the continuity equation. Based on these works, the DG formulation for the quasi-incompressible Navier-Stokes equations discussed here closely follows that presented by Cliffe et al. [38].

Consider the quasi-incompressible Navier-Stokes equations given by

$$\nabla \cdot (\mathcal{F}^c_{\text{mom}}(\mathbf{u}) - \mathcal{F}^v_{\text{mom}}(\mathbf{u}, \nabla\mathbf{u})) = \rho\mathbf{g} \qquad \text{in } \Omega, \tag{3.6.1}$$

$$\nabla \cdot (\rho\mathbf{u}) = 0 \qquad \text{in } \Omega, \tag{3.6.2}$$

$$\mathbf{u} = \mathbf{g}_D \qquad \text{on } \partial\Omega_D, \tag{3.6.3}$$

$$\mathcal{F}^v_{\text{mom}}(\mathbf{u}; \nabla\mathbf{u}) \cdot \mathbf{n} = \mathbf{g}_N \qquad \text{on } \partial\Omega_N, \tag{3.6.4}$$

where

$$\mathcal{F}^c_{\text{mom}}(\mathbf{u}) = \rho\mathbf{u} \otimes \mathbf{u}, \tag{3.6.5}$$

$$\mathcal{F}^v_{\text{mom}}(\mathbf{u}; \nabla\mathbf{u}) = \eta\left(\nabla\mathbf{u} + \nabla\mathbf{u}^\top - \frac{2}{3}(\nabla \cdot \mathbf{u})\,\underline{\mathbf{I}}\right) - p\underline{\mathbf{I}}. \tag{3.6.6}$$

In this section, we consider only the DG discretisation of the continuity equation (3.6.2). To this end, multiplying (3.6.2) by a test function $q \in H^1(\mathcal{T}^h_\Omega)$ and integrating by parts on an element $\kappa \in \mathcal{T}^h_\Omega$ we get

$$-\int_\kappa (\rho\mathbf{u}) \cdot \nabla q \, \mathrm{d}\mathbf{x} + \int_{\partial\kappa} \mathbf{u} \cdot \mathbf{n}_\kappa (\rho q) \, \mathrm{d}s = 0. \tag{3.6.7}$$

Introducing the numerical flux $\hat{\mathbf{u}}$ and integrating by parts a second time gives

$$\int_\kappa q \nabla \cdot (\rho\mathbf{u}) \, \mathrm{d}\mathbf{x} + \int_{\partial\kappa} (\hat{\mathbf{u}} - \mathbf{u}) \cdot \mathbf{n}_\kappa (\rho q) \, \mathrm{d}s = 0. \tag{3.6.8}$$

Summing over all elements in the mesh $\kappa \in \mathcal{T}^h_\Omega$ and replacing $\mathbf{u}$ and q with their discrete finite element approximations $\mathbf{u}_h$ and q_h, respectively, the flux formulation is given by: find $\mathbf{u}_h \in \mathbf{V}^d_\ell(\mathcal{T}^h_\Omega)$ such that,

$$\int_\Omega q_h \nabla_h \cdot (\rho\mathbf{u}_h) \, \mathrm{d}\mathbf{x} + \sum_{\kappa \in \mathcal{T}^h_\Omega} \int_{\partial\kappa} (\hat{\mathbf{u}} - \mathbf{u}_h) \cdot \mathbf{n}_\kappa (\rho q_h) \, \mathrm{d}s = 0 \tag{3.6.9}$$

for all $q_h \in V_{\ell-1}(\mathcal{T}^h_\Omega)$. Regarding the choice of DG finite element space $V_{\ell-1}(\mathcal{T}^h_\Omega)$ we refer to the literature of finite element methods concerning saddle point problems [21, 39, 44]. The primal formulation of equation (3.6.9) can be written in terms of the jump $[\![\cdot]\!]$ and average $\{\!\{\cdot\}\!\}$ operators by applying the identity in (3.2.28). To this end, we get

$$\begin{aligned}\mathcal{N}^{\text{cont}}_\Omega(\mathbf{u}_h; q_h) := &\int_\Omega q_h \nabla_h \cdot (\rho\mathbf{u}_h) \, \mathrm{d}\mathbf{x} \\ &+ \int_{\Gamma_\mathcal{I} \cup \partial\Omega} [\![\rho q_h]\!] \cdot \{\!\{\hat{\mathbf{u}} - \mathbf{u}_h\}\!\} \, \mathrm{d}s + \int_{\Gamma_\mathcal{I}} \{\!\{\rho q_h\}\!\} [\![\hat{\mathbf{u}} - \mathbf{u}_h]\!] \, \mathrm{d}s.\end{aligned} \tag{3.6.10}$$

Here, we define the numerical flux $\hat{\mathbf{u}}$ as follows

$$\hat{\mathbf{u}} = \begin{cases} \{\!\{\mathbf{u}_h\}\!\} & \text{on } \Gamma_\mathcal{I}, \\ \mathbf{g}_D & \text{on } \partial\Omega_D, \\ \mathbf{u}_h^+ & \text{on } \partial\Omega_N, \end{cases} \tag{3.6.11}$$

cf. Section 3.5. Hence, we may write

$$\mathcal{N}_\Omega^{\text{cont}}(\mathbf{u}_h; q_h) := \int_\Omega q_h \nabla_h \cdot (\rho \mathbf{u}_h)\, \mathrm{d}\mathbf{x} - \int_{\Gamma_\mathcal{I}} [\![\mathbf{u}_h]\!] \{\!\{\rho q_h\}\!\}\, \mathrm{d}s - \int_{\partial\Omega} [\![\mathbf{u}_h - \mathbf{u}_\Gamma(\mathbf{u}_h)]\!] \{\!\{\rho q_h\}\!\}\, \mathrm{d}s. \quad (3.6.12)$$

3.7 Discontinuous Galerkin Approximation of the Maxwell Operator

The DG discretisation of the Maxwell operator employed in this thesis is based on the method developed by Houston, Perugia and Schötzau [80]; in particular, it enforces the divergence free condition of the electric field through the introduction of a Lagrange multiplier $\mathfrak{p}$. The properties of this Lagrange multiplier term are discussed in [46, 138]. Let $\Omega \subset \mathbb{R}^d$, $d \geq 1$, be a bounded domain with boundary $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ with unit outward normal vector $\mathbf{n}$, where $\partial\Omega_D$ is closed and non-empty. As before, we let $\mathcal{T}_\Omega^h = \{\kappa\}$ be the subdivision of Ω into shape regular elements of granularity h_κ. The Maxwell operator acting on an unknown vector field $\mathbf{E}$ along with Lagrange multiplier term $\mathfrak{p}$ for material permeability μ and permittivity ε is given by

$$\nabla \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) - \varepsilon \nabla \mathfrak{p} = \mathbf{0} \ \text{ in } \Omega, \quad (3.7.1)$$

$$\nabla \cdot (\varepsilon \mathbf{E}) = 0 \ \text{ in } \Omega, \quad (3.7.2)$$

subject to the boundary conditions

$$\mathbf{n} \times \mathbf{E} = \mathbf{g}_D \ \text{ on } \partial\Omega_D, \quad (3.7.3)$$

$$\mathbf{n} \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) = \mathbf{g}_N \ \text{ on } \partial\Omega_N, \quad (3.7.4)$$

$$\mathfrak{p} = 0 \ \text{ on } \partial\Omega_D. \quad (3.7.5)$$

3.7.1 The *curl-curl* Operator

To define the DG finite element formulation of the Maxwell operator, initially consider the curl-curl operator

$$\nabla \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) = \mathbf{0}. \quad (3.7.6)$$

Writing this as a first order system, we get

$$\sigma = \mu^{-1} \nabla \times \mathbf{E} \ \text{ and } \ \nabla \times \sigma = \mathbf{0}. \quad (3.7.7)$$

We now proceed as in the case of discretising the viscous terms, cf. Section 3.5. Thereby, multiplying both equations by the complex conjugate of complex valued test functions

$\mathbf{F} \in \left[H^1(\mathcal{T}_\Omega^h)\right]^d$ and $\tau \in \left[H^1(\mathcal{T}_\Omega^h)\right]^d$, respectively, and integrating by parts elementwise yields,

$$\int_\kappa \sigma \cdot \overline{\tau} \, \mathrm{d}\mathbf{x} = \int_\kappa \mathbf{E} \cdot \nabla \times \left(\mu^{-1}\overline{\tau}\right) \, \mathrm{d}\mathbf{x} - \int_{\partial\kappa} (\mathbf{E} \times \mathbf{n}_\kappa) \cdot \left(\mu^{-1}\overline{\tau}\right) \, \mathrm{d}s, \tag{3.7.8}$$

$$\int_\kappa \sigma \cdot \nabla \times \overline{\mathbf{F}} = \int_{\partial\kappa} (\sigma \times \mathbf{n}_\kappa) \cdot \overline{\mathbf{F}} \, \mathrm{d}s, \tag{3.7.9}$$

where $\overline{\tau}$ and $\overline{\mathbf{F}}$ denotes the complex conjugate of τ and $\mathbf{F}$, respectively. Summing over all elements $\kappa \in \mathcal{T}_\Omega^h$ and replacing $\mathbf{E}$, $\mathbf{F}$, σ and τ by their discrete finite element counterparts $\mathbf{E}_h, \mathbf{F}_h, \sigma_h, \tau_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$, respectively, gives the flux formulation: find $\mathbf{E}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$ and $\sigma_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$ such that

$$\int_\Omega \sigma_h \cdot \overline{\tau}_h \, \mathrm{d}\mathbf{x} = \int_\Omega \mathbf{E}_h \cdot \nabla_h \times \left(\mu^{-1}\overline{\tau}_h\right) \, \mathrm{d}\mathbf{x} - \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} (\hat{\mathbf{E}}_h \times \mathbf{n}_\kappa) \cdot \left(\mu^{-1}\overline{\tau}_h\right) \, \mathrm{d}s, \tag{3.7.10}$$

$$\int_\Omega \sigma_h \cdot \nabla_h \times \overline{\mathbf{F}}_h = \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \setminus \partial\Omega_N} (\hat{\sigma}_h \times \mathbf{n}_\kappa) \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s + \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \cap \partial\Omega_N} \mathbf{g}_N \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s \tag{3.7.11}$$

for all $\mathbf{F}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$ and $\tau_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$.

Just as with the treatment of the numerical fluxes of viscous terms in Section 3.5, the numerical fluxes $\hat{\mathbf{E}}_h$ and $\hat{\sigma}_h$ represent approximations to $\mathbf{E}$ and $\mu^{-1}\nabla \times \mathbf{E}$, respectively. Choosing the test function $\tau_h = \nabla_h \times \mathbf{F}_h$ and integrating equation (3.7.10) by parts a second time yields

$$\int_\Omega \sigma_h \cdot \nabla_h \times \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} = \int_\Omega \mu^{-1}\nabla_h \times \mathbf{E}_h \cdot \nabla_h \times \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} - \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} \left((\hat{\mathbf{E}}_h - \mathbf{E}_h) \times \mathbf{n}_\kappa\right) \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s. \tag{3.7.12}$$

Substituting (3.7.12) into equation (3.7.11) and noting the following identities for vectors $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^d$

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = \mathbf{b} \cdot (\mathbf{c} \times \mathbf{a}) = \mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}), \tag{3.7.13}$$

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = -\mathbf{a} \cdot (\mathbf{c} \times \mathbf{b}), \tag{3.7.14}$$

the primal sesquilinear formulation can be derived

$$\begin{aligned} a_\Omega^{\mathrm{Max}}(\mathbf{E}_h, \mathbf{F}_h) := \int_\Omega \mu^{-1}\nabla_h \times \mathbf{E}_h \cdot \nabla_h \times \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} + \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} \left(\mathbf{n}_\kappa \times \left(\hat{\mathbf{E}}_h - \mathbf{E}_h\right)\right) \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s \\ - \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} \hat{\sigma} \cdot \left(\mathbf{n}_\kappa \times \overline{\mathbf{F}}_h\right) \, \mathrm{d}s - \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \cap \partial\Omega_N} \mathbf{g}_N \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s = 0. \end{aligned} \tag{3.7.15}$$

By applying the identity in equation (3.2.30) to (3.7.15), the sesquilinear operator can be written in terms of the average and jump operators:

$$\begin{aligned} a_{\Omega}^{\mathrm{Max}}(\mathbf{E}_h, \mathbf{F}_h) := & \int_{\Omega} \mu^{-1} \nabla_h \times \mathbf{E}_h \cdot \nabla_h \times \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} \\ & + \int_{\partial\Omega \cup \Gamma_{\mathcal{I}}} \{\!\{ \mu^{-1} \nabla_h \times \overline{\mathbf{F}}_h \}\!\} \cdot [\![\hat{\mathbf{E}}_h - \mathbf{E}_h]\!]_{\mathrm{T}} \, \mathrm{d}s - \int_{\Gamma_{\mathcal{I}}} [\![\mu^{-1} \nabla_h \times \overline{\mathbf{F}}_h]\!]_{\mathrm{T}} \cdot \{\!\{ \hat{\mathbf{E}}_h - \mathbf{E}_h \}\!\} \, \mathrm{d}s \\ & - \int_{\partial\Omega \cup \Gamma_{\mathcal{I}}} \{\!\{ \hat{\sigma}_h \}\!\} \cdot [\![\overline{\mathbf{F}}_h]\!]_{\mathrm{T}} \, \mathrm{d}s + \int_{\Gamma_{\mathcal{I}}} [\![\hat{\sigma}_h]\!]_{\mathrm{T}} \cdot \{\!\{ \overline{\mathbf{F}}_h \}\!\} \, \mathrm{d}s - \int_{\partial\Omega_N} \mathbf{g}_N \cdot \overline{\mathbf{F}}_h \, \mathrm{d}s. \end{aligned} \tag{3.7.16}$$

The numerical fluxes of the DG discretisation $\hat{\mathbf{E}}_h$ and $\hat{\sigma}_h$ employed here are chosen to be those of the symmetric interior penalty method (see Perugia et al. [115])

$$\hat{\mathbf{E}}_h = \{\!\{ \mathbf{E}_h \}\!\}, \quad \hat{\sigma}_h = \{\!\{ \mu^{-1} \nabla_h \times \mathbf{E}_h \}\!\} - \delta(\mathbf{E}_h) \quad \text{on } \Gamma_{\mathcal{I}}. \tag{3.7.17}$$

Here, the penalisation term $\delta(\mathbf{E}_h)$ for the interior penalty method is

$$\delta(\mathbf{E}_h) = C_{\mathrm{IP}}^{\varepsilon} \frac{\ell^2}{h_F} \left(\min \left\{ \mu^+, \mu^- \right\} \right)^{-1} [\![\mathbf{E}_h]\!]_{\mathrm{T}}, \tag{3.7.18}$$

where $C_{\mathrm{IP}}^{\varepsilon}$ is a positive constant, ℓ is the local element polynomial order, and h_F element face size. Regarding choices of $C_{\mathrm{IP}}^{\varepsilon}$ we refer to [25, 78]. On the exterior boundary, the numerical fluxes incorporate the boundary flux function

$$\hat{\mathbf{E}}_h = \mathbf{E}_{\Gamma}(\mathbf{E}_h), \quad \hat{\sigma}_h = \mu^{-1} \nabla_h \times \mathbf{E}_h - \delta_{\Gamma}(\mathbf{E}_h) \quad \text{on } \partial\Omega_D \tag{3.7.19}$$

with exterior boundary penalisation term $\delta_{\Gamma}(\mathbf{E}_h)$

$$\delta_{\Gamma}(\mathbf{E}_h) = C_{\mathrm{IP}}^{\varepsilon} \frac{\ell^2}{h_F} \left(\mu^+ \right)^{-1} [\![\mathbf{E}_h - \mathbf{E}_{\Gamma}(\mathbf{E}_h)]\!]_{\mathrm{T}}. \tag{3.7.20}$$

In the case of equation (3.7.3) the boundary function $\mathbf{E}_{\Gamma}(\mathbf{E}_h^+)\big|_{\partial\Omega_D} = \mathbf{g}_D$. On the Neumann component of the exterior boundary the numerical flux function $\mathbf{E}_{\Gamma}(\mathbf{E}_h^+)\big|_{\partial\Omega_N} = \mathbf{E}_h^+$.

Substituting the flux terms of equations (3.7.17) and (3.7.19) into the DG primal flux formulation of equation (3.7.16), the full DG discretisation of the *curl-curl* component of the Maxwell operator can be derived

$$\begin{aligned} a_{\Omega}^{\mathrm{Max}}(\mathbf{E}_h, \mathbf{F}_h) := & \int_{\Omega} \mu^{-1} \nabla_h \times \mathbf{E}_h \cdot \nabla_h \times \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} \\ & - \int_{\partial\Omega_D \cup \Gamma_{\mathcal{I}}} \{\!\{ \mu^{-1} \nabla_h \times \overline{\mathbf{F}}_h \}\!\} \cdot [\![\mathbf{E}_h]\!]_{\mathrm{T}} \, \mathrm{d}s - \int_{\partial\Omega_D \cup \Gamma_{\mathcal{I}}} \{\!\{ \mu^{-1} \nabla_h \times \mathbf{E}_h \}\!\} \cdot [\![\overline{\mathbf{F}}_h]\!]_{\mathrm{T}} \, \mathrm{d}s \\ & + \int_{\Gamma_{\mathcal{I}}} \delta(\mathbf{E}_h) \cdot [\![\overline{\mathbf{F}}_h]\!]_{\mathrm{T}} \, \mathrm{d}s + \int_{\partial\Omega_D} \delta_{\Gamma}(\mathbf{E}_h) \cdot [\![\overline{\mathbf{F}}_h]\!]_{\mathrm{T}} \, \mathrm{d}s \\ & + \int_{\partial\Omega_D} [\![\mathbf{E}_{\Gamma}(\mathbf{E}_h)]\!]_{\mathrm{T}} \cdot \{\!\{ \mu^{-1} \nabla_h \times \overline{\mathbf{F}}_h \}\!\} \, \mathrm{d}s - \int_{\partial\Omega_N} \mathbf{g}_N \cdot \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x}. \end{aligned} \tag{3.7.21}$$

3.7.2 The Divergence Free Field Constraint

The DG discretisation of the divergence free electric field constraint and the corresponding Lagrange multiplier term can be determined in an analogous manner to that employed for the continuity equation in Section 3.6. First consider the term $-\varepsilon\nabla\mathfrak{p}$ of equation (3.7.1) which is multiplied by test function $\mathbf{F}$ and integrated by parts on and element $\kappa \in \mathcal{T}_\Omega^h$:

$$-\int_\kappa (\varepsilon\nabla\mathfrak{p})\cdot\overline{\mathbf{F}}\,\mathrm{d}\mathbf{x} = \int_\kappa \mathfrak{p}\nabla\cdot\left(\varepsilon\overline{\mathbf{F}}\right)\,\mathrm{d}\mathbf{x} - \int_{\partial\kappa}\mathfrak{p}\varepsilon\overline{\mathbf{F}}\cdot\mathbf{n}_\kappa\,\mathrm{d}s. \tag{3.7.22}$$

Integrating by parts a second time, summing over all elements, introducing the numerical flux function $\hat{\mathfrak{p}}_h$ and replacing $\mathfrak{p}$ and $\mathbf{F}$ by their discrete finite element counterparts $\mathfrak{p}_h \in V_{\ell+1}(\mathcal{T}_\Omega^h)$ and $\mathbf{F}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$ gives

$$-\sum_{\kappa\in\mathcal{T}_\Omega^h}\int_\kappa (\varepsilon\nabla\mathfrak{p}_h)\cdot\overline{\mathbf{F}}_h\,\mathrm{d}\mathbf{x} = -\int_\Omega (\varepsilon\nabla\mathfrak{p}_h)\cdot\overline{\mathbf{F}}_h\,\mathrm{d}\mathbf{x} - \sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa}(\hat{\mathfrak{p}}_h-\mathfrak{p}_h)\,\varepsilon\overline{\mathbf{F}}_h\cdot\mathbf{n}_\kappa\,\mathrm{d}s. \tag{3.7.23}$$

Regarding the choice of the richer space $V_{\ell+1}(\mathcal{T}_\Omega^h)$ for the Lagrange multiplier, we refer to [46]. Applying the identity relating integrals on element boundaries to integrals over element faces in (3.2.28) allows the derivation of the primal flux semilinear residual:

$$\begin{aligned}\mathcal{N}_\Omega^{\varepsilon\nabla\mathfrak{p}}(\mathfrak{p}_h;\mathbf{F}_h) := &-\int_\Omega (\varepsilon\nabla\mathfrak{p}_h)\cdot\overline{\mathbf{F}}_h\,\mathrm{d}\mathbf{x}\\ &-\int_{\Gamma_\mathcal{I}\cup\partial\Omega}\{\!\{\varepsilon\overline{\mathbf{F}}_h\}\!\}\cdot[\![\hat{\mathfrak{p}}_h-\mathfrak{p}_h]\!]\,\mathrm{d}s - \int_{\Gamma_\mathcal{I}}[\![\varepsilon\overline{\mathbf{F}}_h]\!]\{\!\{\hat{\mathfrak{p}}_h-\mathfrak{p}_h\}\!\}\,\mathrm{d}s;\end{aligned} \tag{3.7.24}$$

here we choose the numerical flux $\hat{\mathfrak{p}} = \{\!\{\mathfrak{p}_h\}\!\}$ on $\partial\Omega\cup\Gamma_\mathcal{I}$. The DG semilinear residual can then be written as

$$\mathcal{N}_\Omega^{\varepsilon\nabla\mathfrak{p}}(\mathfrak{p}_h;\mathbf{F}_h) = -\int_\Omega (\varepsilon\nabla\mathfrak{p}_h)\cdot\overline{\mathbf{F}}_h\,\mathrm{d}\mathbf{x} + \int_{\Gamma_\mathcal{I}\cup\partial\Omega}\{\!\{\varepsilon\overline{\mathbf{F}}_h\}\!\}\cdot[\![\mathfrak{p}_h]\!]\,\mathrm{d}s. \tag{3.7.25}$$

Now consider the divergence free electric field constraint $\nabla\cdot(\varepsilon\mathbf{E}) = 0$. Multiplying by scalar test function $\mathfrak{q} \in H^1(\mathcal{T}_\Omega^h)$ and integrating by parts on an element $\kappa \in \mathcal{T}_\Omega^h$ gives

$$\int_\kappa \overline{\mathfrak{q}}\nabla\cdot(\varepsilon\mathbf{E})\,\mathrm{d}\mathbf{x} = -\int_\kappa \varepsilon\mathbf{E}\cdot\nabla\overline{\mathfrak{q}}\,\mathrm{d}\mathbf{x} + \int_{\partial\kappa}\varepsilon\mathbf{E}\cdot\mathbf{n}_\kappa\overline{\mathfrak{q}}\,\mathrm{d}s = 0. \tag{3.7.26}$$

Summing over all elements $\kappa \in \mathcal{T}_\Omega^h$, replacing $\mathbf{E}$ and $\mathfrak{q}$ by their DG finite element counterparts $\mathbf{E}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$ and $\mathfrak{q}_h \in V_{\ell+1}(\mathcal{T}_\Omega^h)$, respectively, and introducing the numerical flux function $\hat{\mathbf{E}}$ gives

$$-\int_\Omega \varepsilon\mathbf{E}_h\cdot\nabla\overline{\mathfrak{q}_h}\,\mathrm{d}\mathbf{x} + \sum_{\kappa\in\mathcal{T}_\Omega^h}\int_{\partial\kappa}\varepsilon\hat{\mathbf{E}}_h\cdot\mathbf{n}_\kappa\overline{\mathfrak{q}_h}\,\mathrm{d}s = 0. \tag{3.7.27}$$

Applying the identity in equation (3.2.28) to equation (3.7.27), and rewriting the boundary terms as integrals over the faces yields the primal flux formulation:

$$\mathcal{N}_{\Omega}^{\nabla\cdot(\varepsilon\mathbf{E})}(\mathbf{E}_h, \mathfrak{q}_h) := -\int_{\Omega} \varepsilon\mathbf{E}_h \cdot \nabla\overline{\mathfrak{q}_h}\,\mathrm{d}\mathbf{x} + \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} \{\!\{\varepsilon\hat{\mathbf{E}}_h\}\!\} \cdot [\![\overline{\mathfrak{q}_h}]\!]\,\mathrm{d}s + \int_{\Gamma_{\mathcal{I}}} [\![\varepsilon\hat{\mathbf{E}}_h]\!]\{\!\{\overline{\mathfrak{q}_h}\}\!\}\,\mathrm{d}s. \quad (3.7.28)$$

Here, the numerical flux function $\hat{\mathbf{E}}_h = \{\!\{\mathbf{E}_h\}\!\} - \delta_\varepsilon [\![\mathfrak{p}_h]\!]$ for penalisation parameter

$$\delta_\varepsilon = \begin{cases} C_{\mathrm{IP}}^{\varepsilon}\frac{\ell^2}{h_F}\max\{\varepsilon^+,\varepsilon^-\} & \text{on } \Gamma_{\mathcal{I}}, \\ C_{\mathrm{IP}}^{\varepsilon}\frac{\ell^2}{h_F}\max\{\varepsilon^+\} & \text{on } \partial\Omega. \end{cases} \quad (3.7.29)$$

3.8 Summary

In this chapter we have written the DG finite element formulation of nonlinear hyperbolic and elliptic PDEs as well as the Maxwell operator. The DG formulation of the hyperbolic and elliptic terms can now be applied to those found in the equations describing the conservation of mass, molar mass fraction, momentum, energy and electron density in the MPA-CVD reactor model in the previous chapter. Furthermore, the derivation of the DG finite element formulation of the Maxwell operator will be implemented to discretise the time harmonic formulation of Maxwell's equations modelling the microwave field in the MPA-CVD reactor cavity. In the following chapter we will summarise the equations of the MPA-CVD reactor model, their DG finite element formulation, and physically appropriate boundary conditions.

CHAPTER 4

CVD Reactor Model Problem

4.1 Reactor Geometry

Let $\Omega \subset \mathbb{R}^3$ be a bounded domain with exterior boundary $\partial\Omega$ denoting the geometry of a given CVD reactor. An example of such a computational domain is shown in Figure 4.1. Adopting a cylindrical coordinate system and assuming azimuthal symmetry, an axial slice at $\theta = 0$ and $r \geq 0$ is taken yielding the bounded domain $\Omega^* \subset \Omega$ with boundary $\partial\Omega^*$; for simplicity of notation, we simply denote this two-dimensional slice by Ω. This domain is then subdivided into three subdomains characterising components of the CVD reactor such that $\overline{\Omega} = \overline{\Omega}_a \cup \overline{\Omega}_q \cup \overline{\Omega}_v$. Here, Ω_a is the subdomain filled with air at atmospheric pressure, Ω_q is the fused silica window and Ω_v is the vacuum chamber of the CVD reactor containing atomic and molecular hydrogen in which diamond growth occurs on a substrate surface.

The boundary of Ω is divided such that $\partial\Omega = \partial\Omega_{surf} \cup \partial\Omega_{wall} \cup \partial\Omega_{in} \cup \partial\Omega_{out} \cup \partial\Omega_{ant} \cup \partial\Omega_{axis}$. Here, $\partial\Omega_{surf}$ is the component of the boundary specifying the substrate surface on which the diamond is grown, $\partial\Omega_{wall}$ is the wall of the reactor, $\partial\Omega_{in}$ is the gas inlet, $\partial\Omega_{out}$ the gas outlet, $\partial\Omega_{ant}$ the microwave antenna which excites the electric field in the cavity and $\partial\Omega_{axis}$ the exterior boundary component which lies on the axis of symmetry $r = 0$. Each of Ω_v, Ω_q and Ω_a, has exterior boundaries $\partial\Omega_v, \partial\Omega_q$ and $\partial\Omega_a$ respectively. We also denote the interior subdomain interface boundaries $\Gamma_{aq} = \overline{\Omega}_a \cap \overline{\Omega}_q$ and $\Gamma_{vq} = \overline{\Omega}_v \cap \overline{\Omega}_q$. On boundary Γ_{aq} we define the unit normal vector pointing from Ω_a to Ω_q by $\mathbf{n}_{aq}$ and from Ω_q to Ω_a by $\mathbf{n}_{qa}$. Similarly, on boundary Γ_{vq} we define the unit normal pointing from Ω_v to Ω_q by $\mathbf{n}_{vq}$ and from Ω_q to Ω_v by $\mathbf{n}_{qv}$. The two dimensional slice of the CVD computational domain is shown in Figure 4.2.

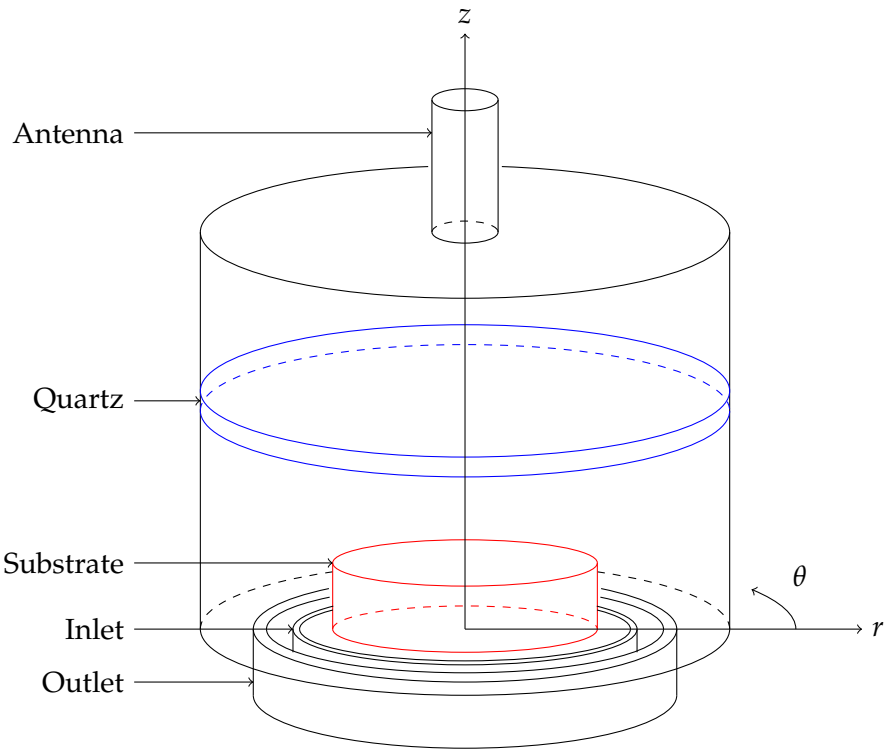


Figure 4.1: Example of a computational domain Ω, a basic representation of a chemical vapour deposition reactor in cylindrical coordinates.

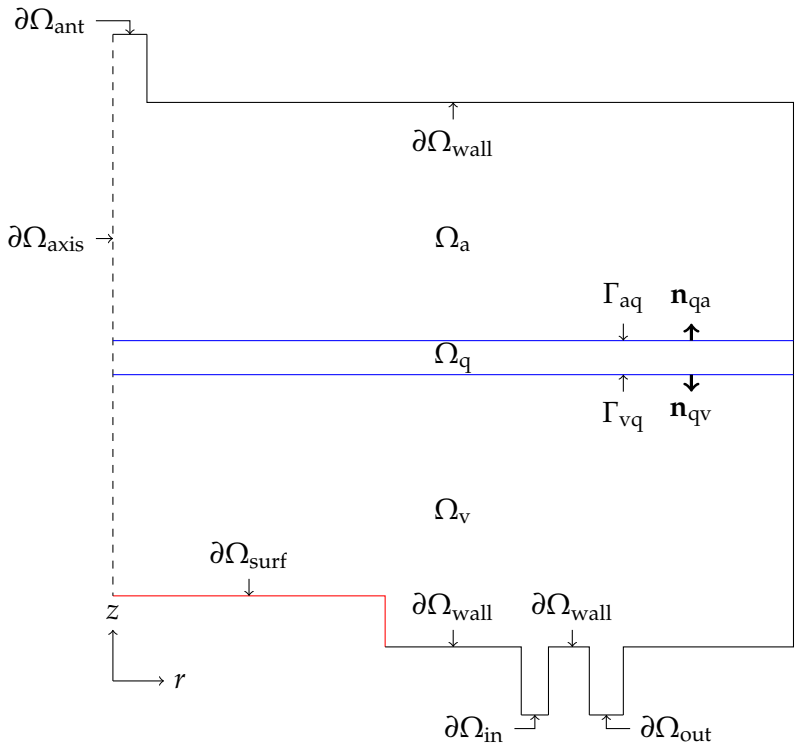


Figure 4.2: Axial slice of the chemical vapour deposition reactor shown in Figure 4.1 at azimuth $\theta = 0$. Collapsing the CVD reactor volume to this two dimensional computational domain is permitted by exploiting azimuthal symmetry.

4.2 Summary of Equations, Boundary Conditions and their DG Formulations

In this thesis we seek numerical approximations to the steady state DG FEM formulation of the CVD reactor model equations, i.e., the time derivatives of the solution variables are zero. To this end, in this section each PDE arising in the CVD model equations is stated; moreover, we define suitable boundary conditions, together with their DG finite element formulations. The unknown quantities to be solved for are: the mass averaged gas flow of the molecular and atomic hydrogen mix $\mathbf{u}$, its relative pressure p, the molar mass fraction of atomic hydrogen x_{H}, the reactor temperature T, the electron density n_{e}, the complex phasor of the time harmonic electric field $\mathbf{E}$ and the complex Lagrange multiplier $\mathfrak{p}$ enforcing a divergence free solution of the electric field variable.

4.2.1 Momentum

Modelling an inlet gas flow profile by $\mathbf{u}_{\mathrm{inlet}}$, no slip $\mathbf{u} = \mathbf{0}$ on the walls of the reactor and allowing the gas to exit the vacuum through the outlet pipe, the multicomponent gas mixture momentum conservation equation takes the form

$$\nabla \cdot (\mathcal{F}^c_{\mathrm{mom}}(\mathbf{u}) - \mathcal{F}^v_{\mathrm{mom}}(\mathbf{u}; \nabla \mathbf{u})) = \rho \mathbf{g} \qquad \text{in } \Omega_{\mathrm{v}}, \tag{4.2.1}$$

$$\nabla \cdot (\rho \mathbf{u}) = 0 \qquad \text{in } \Omega_{\mathrm{v}}, \tag{4.2.2}$$

$$\mathbf{u} = \mathbf{0} \qquad \text{on } \partial\Omega_{\mathrm{ant}} \cup \partial\Omega_{\mathrm{surf}} \cup \partial\Omega_{\mathrm{wall}} \cup \Gamma_{\mathrm{vq}}, \tag{4.2.3}$$

$$\mathbf{u} = \mathbf{u}_{\mathrm{inlet}} \qquad \text{on } \partial\Omega_{\mathrm{in}}, \tag{4.2.4}$$

$$\eta \nabla \mathbf{u} \cdot \mathbf{n} - p\mathbf{n} = \mathbf{0} \qquad \text{on } \partial\Omega_{\mathrm{axis}} \cup \partial\Omega_{\mathrm{out}} \tag{4.2.5}$$

where

$$\mathcal{F}^c_{\mathrm{mom}}(\mathbf{u}) = \rho \mathbf{u} \otimes \mathbf{u}, \tag{4.2.6}$$

$$\mathcal{F}^v_{\mathrm{mom}}(\mathbf{u}; \nabla \mathbf{u}) = \eta \left(\nabla \mathbf{u} + \nabla \mathbf{u}^\top - \frac{2}{3} (\nabla \cdot \mathbf{u}) \underline{\mathbf{I}} \right) - p \underline{\mathbf{I}}. \tag{4.2.7}$$

The density is given in terms of molar masses M_{H} and M_{H_2}, the gas constant R and the mean vacuum pressure P, namely, $\rho = {}^{PM}/_{RT}$, where M is the mean molar mass of the gas mixture, i.e., $M = (M_{\mathrm{H}} x_{\mathrm{H}} + M_{\mathrm{H}_2} (1 - x_{\mathrm{H}}))$. The viscosity is given by

$$\eta = \left(\frac{6 x_{\mathrm{H}}}{6 x_{\mathrm{H}} + \sqrt{3} \left(1 + 2^{\frac{1}{4}} N\right)^2 (1 - x_{\mathrm{H}})} + \frac{12 (1 - x_{\mathrm{H}})}{12 N^2 (1 - x_{\mathrm{H}}) + \sqrt{3} \left(1 + 2^{\frac{1}{4}} N\right)^2 x_{\mathrm{H}}} \right) \eta_{\mathrm{H}}, \tag{4.2.8}$$

where $N = \sqrt{\eta_{\mathrm{H}}/\eta_{\mathrm{H}_2}}$ and η_{H} and η_{H_2} are the viscosities of atomic and molecular hydrogen, respectively. The Neumann boundary conditions require that on $\partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}$

$$\mathbf{u}_\Gamma(\mathbf{u}^+) = \mathbf{u}^+, \tag{4.2.9}$$

$$\mathcal{F}^v_{\text{mom}}\left(\mathbf{u}_\Gamma(\mathbf{u}^+); \nabla\mathbf{u}\right) \cdot \mathbf{n} = \eta\left(\nabla\mathbf{u}^\top - \frac{2}{3}\left(\nabla\cdot\mathbf{u}\right)\underline{\mathbf{I}}\right)\cdot\mathbf{n}. \tag{4.2.10}$$

The DG formulation of the gas flow model is to find $(\mathbf{u}_h, p_h) \in \mathbf{V}^d_\ell(\mathcal{T}^h_{\Omega_\text{v}}) \times V_{\ell-1}(\mathcal{T}^h_{\Omega_\text{v}})$ such that

$$\mathcal{N}^{\text{gas flow}}_{\Omega_\text{v}}\left(\mathbf{u}_h, p_h; \mathbf{v}_h, q_h\right) := \mathcal{N}^{c,\text{mom}}_{\Omega_\text{v}}\left(\mathbf{u}_h; \mathbf{v}_h\right) + \mathcal{N}^{v,\text{mom}}_{\Omega_\text{v}}\left(\mathbf{u}_h; \mathbf{v}_h\right) + \mathcal{N}^{\text{cont}}_{\Omega_\text{v}}\left(\mathbf{u}_h; q_h\right) = 0 \tag{4.2.11}$$

for all $(\mathbf{v}_h, q_h) \in \mathbf{V}^d_\ell(\mathcal{T}^h_{\Omega_\text{v}}) \times \in V_{\ell-1}(\mathcal{T}^h_{\Omega_\text{v}})$.

4.2.2 Mass

Assuming no presence of atomic hydrogen on the walls of the reactor and that the gas at the inlet is pure molecular hydrogen, the conservation law enforcing continuity of mass fraction is

$$\nabla\cdot\left(\mathcal{F}^c_{\text{mass}}(x_\mathrm{H}) - \mathcal{F}^v_{\text{mass}}(x_\mathrm{H}; \nabla x_\mathrm{H})\right) = R_\mathrm{H} \qquad \text{in } \Omega_\text{v}, \tag{4.2.12}$$

$$x_\mathrm{H} = 0 \qquad \text{on } \partial\Omega_\text{v} \setminus \left(\partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}\right), \tag{4.2.13}$$

$$\nabla x_\mathrm{H} \cdot \mathbf{n} = 0 \qquad \text{on } \partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}, \tag{4.2.14}$$

where

$$\mathcal{F}^c_{\text{mass}}(x_\mathrm{H}) = c x_\mathrm{H}\mathbf{u}, \tag{4.2.15}$$

$$\mathcal{F}^v_{\text{mass}}(x_\mathrm{H}; \nabla x_\mathrm{H}) = \frac{M_{\mathrm{H}_2}}{M} c\mathcal{D}_{\mathrm{HH}_2}\nabla x_\mathrm{H}. \tag{4.2.16}$$

Here, D_{HH_2} is the diffusivity of atomic hydrogen in the binary gas mixture, $c = {}^P/_{RT}$ is the molar concentration of the gas mixture and $R_\mathrm{H} = 2\left(k_f c^2\left(1 - x_\mathrm{H}\right) - k_r c^3 x^2_\mathrm{H}\right)$ is the rate of mass production of atomic hydrogen for forward and reverse rate constants k_f and k_r, respectively. The natural Neumann condition $(\nabla x_\mathrm{H})\cdot\mathbf{n} = 0$ is prescribed on the axis of symmetry and gas outlet requiring that $x_{H\Gamma}(x^+_\mathrm{H}) = x^+_\mathrm{H}$ on $\partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}$. The DG formulation is to find $x_{\mathrm{H}h} \in V_\ell(\mathcal{T}^h_{\Omega_\text{v}})$ such that

$$\begin{aligned}\mathcal{N}^{\text{mass fraction}}_{\Omega_\text{v}}\left(x_{\mathrm{H}h}; \xi_{x_\mathrm{H}h}\right) := \mathcal{N}^{c,\text{mass}}_{\Omega_\text{v}}\left(x_{\mathrm{H}h}; \xi_{x_\mathrm{H}h}\right) + \mathcal{N}^{v,\text{mass}}_{\Omega_\text{v}}\left(x_{\mathrm{H}h}; \xi_{x_\mathrm{H}h}\right) \\ - \int_{\Omega_\text{v}} R_\mathrm{H}\xi_{x_\mathrm{H}h}\,\mathrm{d}\mathbf{x} = 0\end{aligned} \tag{4.2.17}$$

for all $\xi_{x_\mathrm{H}h} \in V_\ell(\mathcal{T}^h_{\Omega_\text{v}})$.

4.2.3 Energy

The source of heat in the hydrogen gas is generated by the time averaged ohmic power absorbed, i.e.,

$$P_{\text{ohm}} = \frac{1}{2} |\mathbf{E}|^2 \sigma_{\text{dc}} \frac{\nu_{m_e}^2}{\omega^2 + \nu_{m_e}^2}. \tag{4.2.18}$$

Here, ω is the electric field angular frequency and ν_{m_e} is the electron-neutral collision frequency. The direct current cold plasma conductivity approximation σ_{dc} is given in terms of the electron rest mass m_e and electron charge q_e, namely,

$$\sigma_{\text{dc}} = \frac{{q_e}^2 n_e}{m_e \nu_{m_e}}. \tag{4.2.19}$$

The sources of heat in the quartz and the air filled cavity are approximated by the Joule heating model for material conductivity σ, i.e.,

$$P = \sigma |\mathbf{E}|^2. \tag{4.2.20}$$

The temperature within the reactor is modelled such that the walls and the inlet gas are held at room temperature T_{room} and the diamond substrate surface is heated to T_{surface}. The temperature and heat flux are required to be continuous across the interfaces between the vacuum, the quartz window and the air filled cavity. The energy balance of the CVD reactor is then modelled by conserving energy in the vacuum T_{v}, in the quartz T_{q} and the air filled cavity T_{a} according to

$$\nabla \cdot \left(\mathcal{F}^c_{\text{energy,v}}(T_{\text{v}}) - \mathcal{F}^v_{\text{energy,v}}(T_{\text{v}}; \nabla T_{\text{v}}) \right) = P_{\text{ohm}} \qquad \text{in } \Omega_{\text{v}}, \tag{4.2.21}$$

$$-\nabla \cdot \mathcal{F}^v_{\text{energy,q}}(T_{\text{q}}; \nabla T_{\text{q}}) = \sigma_{\text{q}} |\mathbf{E}|^2 \qquad \text{in } \Omega_{\text{q}}, \tag{4.2.22}$$

$$-\nabla \cdot \mathcal{F}^v_{\text{energy,a}}(T_{\text{a}}; \nabla T_{\text{a}}) = \sigma_{\text{a}} |\mathbf{E}|^2 \qquad \text{in } \Omega_{\text{a}}, \tag{4.2.23}$$

$$T_{\text{v}} = T_{\text{a}} = T_{\text{q}} = T_{\text{room}} \qquad \text{on } \partial\Omega_{\text{wall}} \cup \partial\Omega_{\text{ant}} \cup \partial\Omega_{\text{in}}, \tag{4.2.24}$$

$$T_{\text{v}} = T_{\text{surface}} \qquad \text{on } \partial\Omega_{\text{surf}}, \tag{4.2.25}$$

$$\nabla T_{\text{v}} \cdot \mathbf{n} = 0 \qquad \text{on } \partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}, \tag{4.2.26}$$

$$\nabla T_{\text{a}} \cdot \mathbf{n} = \nabla T_{\text{q}} \cdot \mathbf{n} = 0 \qquad \text{on } \partial\Omega_{\text{axis}}, \tag{4.2.27}$$

where

$$\mathcal{F}^c_{\text{energy,v}}(T_{\text{v}}) = \rho h \mathbf{u}, \qquad \mathcal{F}^v_{\text{energy,v}}(T_{\text{v}}; \nabla T_{\text{v}}) = \kappa_{\text{v}} \nabla T_{\text{v}}, \tag{4.2.28}$$

$$\mathcal{F}^v_{\text{energy,q}}(T_{\text{q}}; \nabla T_{\text{q}}) = \kappa_{\text{q}} \nabla T_{\text{q}}, \qquad \mathcal{F}^v_{\text{energy,a}}(T_{\text{a}}; \nabla T_{\text{a}}) = \kappa_{\text{a}} \nabla T_{\text{a}}, \tag{4.2.29}$$

and κ_{v}, κ_{q} and κ_{a} are the thermal conductivities in the vacuum, quartz and air, respectively. Here, κ_{v} is given in terms of thermal conductivities of atomic κ_{H} and molecular

κ_{H_2} hydrogen, i.e.,

$$\kappa_v = \frac{1}{2}\left(\sum_{i\in\{H,H_2\}} x_i\kappa_i + \left(\sum_{i\in\{H,H_2\}} \frac{x_i}{\kappa_i}\right)^{-1}\right). \tag{4.2.30}$$

We employ the thermal conductivity of fused silica κ_q presented in [128]. These equations are also subject to the interface boundary conditions on Γ_{aq} and Γ_{vq} enforcing continuity of the heat flux:

$$T_v = T_q, \qquad \kappa_v \nabla T_v \cdot \mathbf{n}_{vq} = \kappa_q \nabla T_q \cdot \mathbf{n}_{vq} \qquad \text{on } \Gamma_{vq}, \tag{4.2.31}$$

$$T_q = T_a, \qquad \kappa_q \nabla T_q \cdot \mathbf{n}_{qa} = \kappa_a \nabla T_a \cdot \mathbf{n}_{qa} \qquad \text{on } \Gamma_{aq}. \tag{4.2.32}$$

Writing the temperature of the CVD reactor as T such that $T|_{\Omega_v} = T_v$, $T|_{\Omega_q} = T_q$ and $T|_{\Omega_a} = T_a$, convective and viscous flux operators can be written for the whole reactor domain $\mathcal{F}^c_{\text{energy}}(T)$ and $\mathcal{F}^v_{\text{energy}}(T;\nabla T)$, respectively, in terms of the thermal conductivity κ and heat source Q. The equivalent conservation of energy equation is

$$\nabla \cdot \left(\mathcal{F}^c_{\text{energy}}(T) - \mathcal{F}^v_{\text{energy}}(T;\nabla T)\right) = Q \quad \text{in } \Omega, \tag{4.2.33}$$

where

$$\mathcal{F}^c_{\text{energy}}(T) = \begin{cases} \rho h \mathbf{u} & \text{in } \Omega_v \\ \mathbf{0} & \text{otherwise} \end{cases}, \tag{4.2.34}$$

$$\mathcal{F}^v_{\text{energy}}(T;\nabla T) = \kappa \nabla T, \tag{4.2.35}$$

and piecewise material parameters are given by

$$\kappa = \begin{cases} \kappa_a & \text{in } \Omega_a \\ \kappa_v & \text{in } \Omega_v \\ \kappa_q & \text{in } \Omega_q \end{cases}, \quad Q = \begin{cases} \sigma_a |\mathbf{E}|^2 & \text{in } \Omega_a \\ P_{\text{ohm}} & \text{in } \Omega_v \\ \sigma_q |\mathbf{E}|^2 & \text{in } \Omega_q \end{cases}. \tag{4.2.36}$$

The natural Neumann condition $\nabla T \cdot \mathbf{n} = 0$ requires that $T_\Gamma(T^+) = T^+$ on $\partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{out}}$. The DG formulation is given by: find $T_h \in V_\ell(\mathcal{T}^h_\Omega)$ such that

$$\begin{aligned} \mathcal{N}^{\text{temperature}}_\Omega(T_h;\xi_{T_h}) := \mathcal{N}^{c,\text{energy}}_\Omega(T_h;\xi_{Th}) + \mathcal{N}^{v,\text{energy}}_\Omega(T_h;\xi_{Th}) \\ - \int_\Omega Q\xi_{Th}\, d\mathbf{x} = 0 \end{aligned} \tag{4.2.37}$$

for all $\xi_{Th} \in V_\ell(\mathcal{T}^h_\Omega)$.

4.2.4 Electron Density

It is assumed that there are no free electrons present on all physical boundaries of the CVD reactor vacuum chamber and at the gas mixture inlet. The conservation of electron particle density is given by the ambipolar diffusion approximation

$$\begin{aligned} \nabla \cdot (\mathcal{F}_{\mathrm{e}}^{c}(n_{\mathrm{e}}) - \mathcal{F}_{\mathrm{e}}^{v}(n_{\mathrm{e}}, \nabla n_{\mathrm{e}})) & \\ + n_{\mathrm{e}} \left(R_0 n_{\mathrm{e}} - A_0 \left| \mathbf{E} \right|^2 \right) = 0 & \qquad \text{in } \Omega_{\mathrm{v}}, \qquad (4.2.38) \\ n_{\mathrm{e}} = 0 & \qquad \text{on } \partial\Omega_{\mathrm{v}} \setminus (\partial\Omega_{\mathrm{axis}} \cup \partial\Omega_{\mathrm{out}}), \qquad (4.2.39) \\ \nabla n_{\mathrm{e}} \cdot \mathbf{u} = 0 & \qquad \text{on } \partial\Omega_{\mathrm{axis}} \cup \partial\Omega_{\mathrm{out}}. \qquad (4.2.40) \end{aligned}$$

Here, the electron generation by ionisation is scaled by the inelastic rate constant A_0, and the loss due to dissociative recombination of electrons with hydrogen ions is scaled by the recombination coefficient R_0. The convective and viscous flux operators are given in terms of ambipolar diffusion coefficient D_a (see [68, 91] for appropriate choices) by

$$\mathcal{F}_{\mathrm{e}}^{c}(n_{\mathrm{e}}) = n_{\mathrm{e}}\mathbf{u}, \qquad (4.2.41)$$

$$\mathcal{F}_{\mathrm{e}}^{v}(n_{\mathrm{e}}; \nabla n_{\mathrm{e}}) = D_a \nabla n_{\mathrm{e}}. \qquad (4.2.42)$$

The Neumann condition requires that $n_{\mathrm{e}\Gamma}(n_{\mathrm{e}}{}^{+}) = n_{\mathrm{e}}{}^{+}$ on $\partial\Omega_{\mathrm{axis}} \cup \partial\Omega_{\mathrm{out}}$. The DG formulation is to find $n_{\mathrm{e}h} \in V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h)$ such that

$$\begin{aligned} \mathcal{N}_{\Omega_{\mathrm{v}}}^{\text{e density}}(n_{\mathrm{e}h}; \xi_{n_{\mathrm{e}}h}) := \mathcal{N}_{\Omega_{\mathrm{v}}}^{c,\mathrm{e}}(n_{\mathrm{e}h}; \xi_{n_{\mathrm{e}}h}) + \mathcal{N}_{\Omega_{\mathrm{v}}}^{v,\mathrm{e}}(n_{\mathrm{e}h}; \xi_{n_{\mathrm{e}}h}) & \\ + \int_{\Omega_{\mathrm{v}}} n_{\mathrm{e}h} \left(R_0 n_{\mathrm{e}h} - A_0 \left| \mathbf{E} \right|^2 \right) \xi_{n_{\mathrm{e}}h} \, \mathrm{d}\mathbf{x} = 0 & \qquad (4.2.43) \end{aligned}$$

for all $\xi_{\mathrm{e}h} \in V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h)$.

4.2.5 Electric Field

The time-harmonic formulation of Maxwell's equations describe the electric field of frequency ω in the CVD reactor with permeability μ, permittivity ε and electric conductivity σ. The permeability of the gas mixture, quartz window and the air filled cavity are all assumed to be equivalent to the permeability of free space μ_0. The permittivity is discontinuous across the subdomains of the reactor; namely,

$$\varepsilon = \begin{cases} \varepsilon_p & \text{in } \Omega_{\mathrm{v}}, \\ \varepsilon_{\mathrm{q}} & \text{in } \Omega_{\mathrm{q}}, \\ \varepsilon_{\mathrm{a}} & \text{in } \Omega_{\mathrm{a}}, \end{cases} \qquad (4.2.44)$$

where in the vacuum region the complex plasma permittivity is given by

$$\varepsilon_p = \varepsilon_0 \left[1 - \frac{\omega_{\text{pe}}^2}{\omega \left(\omega - j\nu_{m_{\text{e}}} \right)} \right], \tag{4.2.45}$$

where $j = \sqrt{-1}$ is the complex unit, ε_0 is the free space permittivity, $\omega_{\text{pe}}^2 = {q_{\text{e}}}^2 n_{\text{e}} / \varepsilon_0 m_{\text{e}}$ is the characteristic plasma frequency and $\nu_{m_{\text{e}}}$ is the electron-neutral collision frequency. In the case of neutral hydrogen atoms we take $\nu_{m_{\text{e}}} \approx 1 \times 10^{10} P/T$ (see [36, 52, 55]). The electric conductivity is discontinuous across the subdomains of the reactor such that

$$\sigma = \begin{cases} \sigma_p & \text{in } \Omega_{\text{v}}, \\ \sigma_{\text{q}} & \text{in } \Omega_{\text{q}}, \\ \sigma_{\text{a}} & \text{in } \Omega_{\text{a}}, \end{cases} \tag{4.2.46}$$

where

$$\sigma_p = \frac{\varepsilon_0 \omega_{\text{pe}}^2}{j\omega + \nu_{m_{\text{e}}}}, \quad \sigma_{\text{q}} = 1.3 \times 10^{-18}, \quad \sigma_{\text{a}} = 3 \times 10^{-15}. \tag{4.2.47}$$

Subject to perfect electric conductor boundary conditions on the reactor walls and excitation $\mathbf{E}_{\text{ant}}$ from the antenna, the time harmonic formulation of Maxwell's equations in the CVD reactor is given by

$$\nabla \times \left(\mu^{-1} \nabla \times \mathbf{E} \right) - \varepsilon \nabla \mathfrak{p} + j\omega \left(\sigma + j\omega\varepsilon \right) \mathbf{E} = \mathbf{0} \qquad \text{in } \Omega, \tag{4.2.48}$$

$$\nabla \cdot (\varepsilon \mathbf{E}) = 0 \qquad \text{in } \Omega, \tag{4.2.49}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{wall}} \cup \partial\Omega_{\text{surf}}, \tag{4.2.50}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{n} \times \mathbf{E}_{\text{ant}} \qquad \text{on } \partial\Omega_{\text{ant}}, \tag{4.2.51}$$

$$\mathbf{n} \times \left(\mu^{-1} \nabla \times \mathbf{E} \right) = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{axis}} \cup \partial\Omega_{\text{in}} \cup \partial\Omega_{\text{out}}. \tag{4.2.52}$$

As shown in Section 3.7, the semilinear DG residual formulation of the Maxwell equations is to find $(\mathbf{E}_h, \mathfrak{p}_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h)$ such that

$$\begin{aligned} \mathcal{N}_\Omega^{\text{E-field}} \left(\mathbf{E}_h, \mathfrak{p}_h; \mathbf{F}_h, \mathfrak{q}_h \right) &:= a_\Omega^{\text{Max}}(\mathbf{E}_h, \mathbf{F}_h) + \mathcal{N}_\Omega^{\varepsilon\nabla\mathfrak{p}} \left(\mathfrak{p}_h; \mathbf{F}_h \right) \\ &\quad + \mathcal{N}_\Omega^{\nabla\cdot(\varepsilon\mathbf{E})} \left(\mathbf{E}_h; \mathfrak{q}_h \right) + \int_\Omega j\omega \left(\sigma + j\omega\varepsilon \right) \mathbf{E}_h \cdot \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} = 0 \end{aligned} \tag{4.2.53}$$

for all $(\mathbf{F}_h, \mathfrak{q}_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h)$.

4.2.6 System Residual

For each of the unknown variables of the MPA-CVD reactor model: the mass average gas flow field $\mathbf{u}$, pressure p, hydrogen molar mass fraction x_{H}, temperature T, electric

field phasor $\mathbf{E}$, Lagrange multiplier $\mathfrak{p}$ and electron density n_{e}, we seek their DG finite element approximation given by: find

$$\begin{aligned}\mathbf{U}_h = (\mathbf{u}_h, p_h, x_{\mathrm{H}h}, T_h, \mathbf{E}_h, \mathfrak{p}_h, n_{\mathrm{e}h}) \in \\ \mathbf{V}_\ell^d(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_{\ell-1}(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_\ell(\mathcal{T}_\Omega^h) \times \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h) \times V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h)\end{aligned} \tag{4.2.54}$$

such that

$$\begin{aligned}\mathcal{N}_\Omega^{\text{system}}(\mathbf{U}_h; \Xi_h) := \mathcal{N}_{\Omega_{\mathrm{v}}}^{\text{gas flow}}(\mathbf{u}_h, p_h; \mathbf{v}_h, q_h) + \mathcal{N}_{\Omega_{\mathrm{v}}}^{\text{mass fraction}}(x_{\mathrm{H}h}; \xi_{x_{\mathrm{H}}h}) \\ + \mathcal{N}_\Omega^{\text{temperature}}(T_h; \xi_{Th}) + \mathcal{N}_{\Omega_{\mathrm{v}}}^{\mathrm{e}}(n_{\mathrm{e}}; \xi_{n_{\mathrm{e}}h}) + \mathcal{N}_\Omega^{\text{E-field}}(\mathbf{E}_h, \mathfrak{p}_h; \mathbf{F}_h, \mathfrak{q}_h) \\ = 0\end{aligned} \tag{4.2.55}$$

for all

$$\begin{aligned}\Xi_h = (\mathbf{v}_h, q_h, \xi_{x_{\mathrm{H}}h}, \xi_{Th}, \mathbf{F}_h, \mathfrak{q}_h, \xi_{n_{\mathrm{e}}h}) \in \\ \mathbf{V}_\ell^d(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_{\ell-1}(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h) \times V_\ell(\mathcal{T}_\Omega^h) \times \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h) \times V_\ell(\mathcal{T}_{\Omega_{\mathrm{v}}}^h).\end{aligned} \tag{4.2.56}$$

Here, all parameters and coefficients including those which are functions of the unknown solution variables stated in (4.2.54) are replaced by their finite element counterparts in each semilinear residual formulation $\mathcal{N}(\cdot;\cdot)$. Furthermore, it is assumed that any geometric discontinuities in parameters are lined up perfectly with the mesh $\mathcal{T}_\Omega^h$.

4.3 Microwave Cavity Resonant Frequency

A key aspect of designing a MPA-CVD reactor is to achieve electric field resonance. Consider the case of the empty reactor geometry represented by Ω. Restricting all boundary components not lying on the axis of symmetry to be perfect electric conductors, the resonant frequencies are calculated from the eigenpair solutions $(\mathbf{0} \neq \mathbf{E}, \gamma^2) \in \mathbb{C}^3 \times \mathbb{C}$ of

$$\nabla \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) = \gamma^2 \mathbf{E} \qquad \text{in } \Omega, \tag{4.3.1}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega \setminus \partial\Omega_{\text{axis}}, \tag{4.3.2}$$

$$\mathbf{n} \times \left(\mu^{-1} \nabla \times \mathbf{E}\right) = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{axis}}. \tag{4.3.3}$$

where $\gamma^2 = -j\omega(\sigma + j\omega\varepsilon)$. In the case of the empty cavity where $(\sigma, \varepsilon, \mu) = (0, \varepsilon_0, \mu_0)$, the resonant frequencies are $\omega_r = \gamma/\sqrt{\varepsilon_0}$. The discrete formulation of the eigen problem (4.3.1) is to find $(\mathbf{0} \neq \mathbf{E}_h, \gamma_h^2) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times \mathbb{C}$ such that

$$a_\Omega^{\text{Max}}(\mathbf{E}_h, \mathbf{F}_h) = \int_\Omega \gamma_h^2 \mathbf{E}_h \cdot \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} \tag{4.3.4}$$

for all $\mathbf{F}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$.

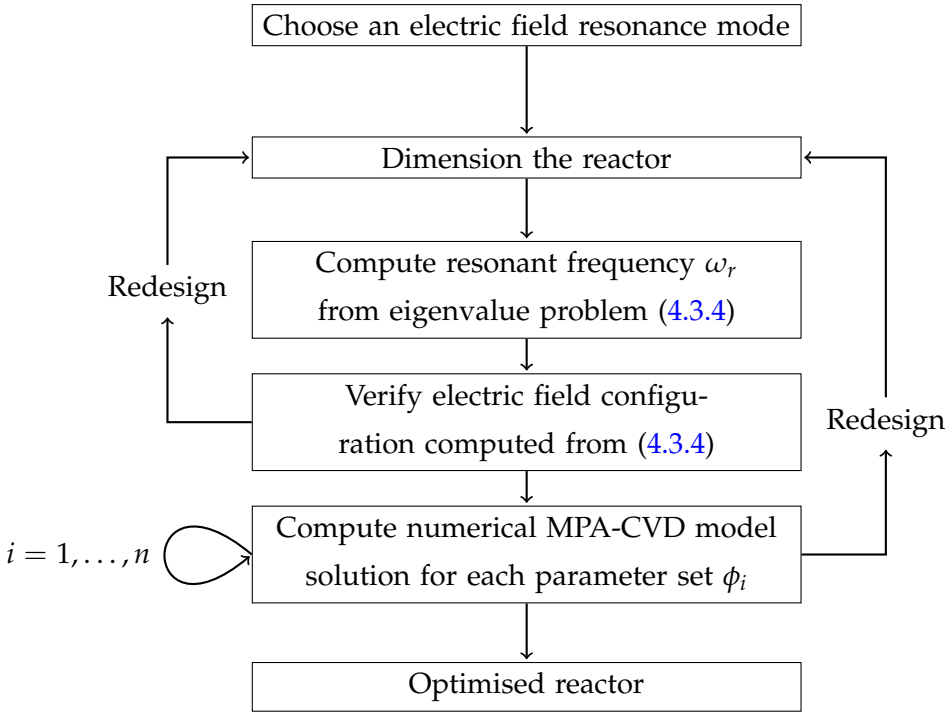


Figure 4.3: MPA-CVD reactor discrete optimisation procedure.

4.4 Optimisation Procedure

Following the choice of the electric field resonance mode, the MPA-CVD reactor system's operational parameters should be optimised. Examples of these parameters include geometric dimensions, mean system pressure, substrate temperature and inlet gas flow rate. Given n parameter sets $\Phi = \{\phi_1, \ldots, \phi_n\}$, the parameter set which maximises the so-called quality factor function $Q_f(\Phi)$ is the optimum MPA-CVD operating condition. In this thesis we employ the quality factor function implemented by Füner et al. [55] which measures the ratio of the magnitude of the electric field in the plasma to the rest of the gas. For a minimum electron number density criterion ϵ_{n_e}, the plasma region is defined such that $\Omega_{\text{plasma}} = \{\kappa : \min_\kappa n_e \geq \epsilon_{n_e},\ \kappa \in \mathcal{T}_\Omega^h\}$ and the remaining vacuum chamber region $\Omega_{\text{E-field}} = \Omega_v \setminus \Omega_{\text{plasma}}$. We seek to optimise the electric field power deposited in the plasma according to quality factor

$$Q_f(\Phi) = \frac{\|\mathbf{E}\|_{L_2(\Omega_{\text{plasma}})}}{\|\mathbf{E}\|_{L_2(\Omega_{\text{E-field}})}}. \tag{4.4.1}$$

In the work presented here, we employ a simple exhaustive method for small parameter sets Φ, choosing the optimum parameter set ϕ_i such that $Q_f(\phi_i) = \max Q_f(\Phi)$. A flow chart of this optimisation procedure is presented in Figure 4.3 (cf. [55]). Here, after choosing the operating electric field resonant mode, the geometry of the reactor is designed. We then verify the resonant electric field configuration of the empty cavity

ensuring that the design meets the requirement for plasma ignition encouraging diamond growth. Once satisfied, we test the performance of the reactor under operation with system configurations specified in the parameter set Φ by computing numerical approximations of the fully self consistent MPA-CVD reactor model.

CHAPTER 5

AptoPy

As we have seen in Chapter 4, the implementation of the DG MPA-CVD reactor model is extremely challenging. A major aspect of the work in this thesis is the development of AptoPy. In this chapter, AptoPy, the computational framework for automatically generating code to solve PDEs using the finite element method is introduced. The fundamental paradigm of AptoPy is that for a given PDE, the code required to calculate its finite element solution should be automatically generated, given a computational symbolic algebra representation of the underlying finite element formulation.

Initially, we introduce the method for computational symbolic algebra in Section 5.1. The use of this symbolic algebra as a means to represent the weak formulation of a PDE is shown, as well as handling finite element function spaces and meshes. The DG discretisation scheme for elliptic and hyperbolic PDE operators is then simplified with their automatic computation in this symbolic algebra framework. The tools which the AptoPy package offers with regards to DG methods, such as automatic computation of the homogeneity tensor and local-Lax Friedrichs flux, are discussed and demonstrated in Section 5.3.

For a given a finite element formulation representation in AptoPy, Section 5.4 demonstrates the solution procedure. Initially the PDE variable indexing and translation to Fortran code and primitive variables is discussed. The translation procedure is then demonstrated, i.e., parsing the Python code representation of a weak formulation in AptoPy to a meaningful representation in the Fortran code for AptoFEM. Lastly, the element and boundary finite element matrix construction procedures are then shown such that they can be applied in an iterative Newton method to compute the finite element solution.

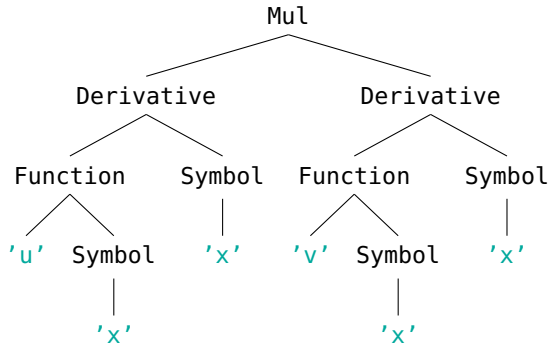


Figure 5.1: Sympy tree structure representing the expression $\partial_x u(x)\partial_x v(x)$. Each class handles: `Mul` multiplication, `Derivative` differentiation, `Function` functions and `Symbol` algebraic symbols.

5.1 Symbolic Representation

5.1.1 Expressions and sympy

AptoPy employs the python symbolic algebra library sympy as the framework for representing and evaluating mathematical expressions and operators [133]. At its core, sympy stores mathematical expressions as trees made up of custom objects for different mathematical concepts. Each of the classes from which these objects are instantiated inherit from a diverse hierarchy, becoming more abstract until reaching the base class `Basic`. Each level of this class hierarchy adds specialisation characteristics; for example, indicating whether the object is an algebraic symbol x or a function such as $\sin(\cdot)$, as well as assigning properties to an expression, such as whether it is differentiable, real, complex, a power expansion series, an infinite sum and so on. Furthermore, these objects serve to overload the standard mathematical operators of Python: addition `+`, subtraction `-`, multiplication $*$, division `/` and exponentiation $**$, providing familiar syntax for their manipulation.

For example, the simple expression '$2+3$' is represented in sympy as `Add(Integer(2), Integer(3))`. Here the object `Add` has two arguments of type `Integer` which in turn have the primitive `int` arguments `2` and `3`, respectively. A graphical representation of the tree required to represent the more complicated expression $\partial_x u(x)\partial_x v(x)$ is given in Figure 5.1.

5.1.2 The Coordinate System Singleton

Vector calculus is an integral part of the construction of a multi-dimensional weak formulation. Operations such as scalar and vector products, as well as tensor contractions must be defined. Furthermore, the encapsulation of operations involving vector gradients in the AptoPy code should evaluate implicitly based on the chosen coordinate system.

AptoPy accommodates any curvilinear coordinate system based on an implementation of the abstract `CoordinateSystem` class. `CoordinateSystem` itself is also the manager of its own singleton instance via the static mutator and corresponding static accessor methods `CoordinateSystem.set()` and `CoordinateSystem.get()`. The `CoordinateSystem` class defines the abstract methods `dim()` and `space_vars()` which, respectively when overridden, provide the number of spatial dimensions and the vector representation of the position vector. Furthermore, the abstract methods `grad()`, `div()` and `curl()` correspond to the respective operations gradient (∇), divergence ($\nabla\cdot$) and curl ($\nabla\times$).

An example of the implementation of this class for the 2D Cartesian coordinate system is the class `CartesianCoordinateSystem2D` as shown in Figure 5.2. This implementation can then be chosen as the coordinate system to be used by simply calling `CoordinateSystem.set(CartesianCoordinateSystem2D())`.

5.1.3 Finite Element Mesh

Given an open bounded Lipschitz domain $\Omega \subset \mathbb{R}^d$, $d \geq 1$, the starting point to introduce the finite element method is to first define the mesh $\mathcal{T}_\Omega^h = \{\kappa\}$, consisting of non-overlapping elements κ such that $\bigcup \overline{\kappa} = \overline{\Omega}$. A simple 2D example of such a meshing procedure is given in Figure 5.3.

The requirement for AptoPy's representation of the mesh is to encapsulate the properties of not only the underlying computational geometry, but also to generate the representation of the unit outward normal vector $\mathbf{n}_\kappa \in \mathbb{R}^d$ on each element $\kappa \in \mathcal{T}_\Omega^h$, the spatial variables $\mathbf{x} \in \mathbb{R}^d$, the element volume integration element d$\mathbf{x}$ and the face integration element ds for both interior and exterior faces.

The dimension d and the spatial variables vector $\mathbf{x}$ of a `Mesh` object are always implicitly specified by the `CoordinateSystem` singleton. In the same implicit fashion, the

```
class CartesianCoordinateSystem2D(CoordinateSystem):

    def __init__(self):
        self.x, self.y = Symbol('x'), Symbol('y')

    def dim(self):
        return 2

    def space_vars(self):
        return Matrix([self.x, self.y])

    def div(self, u):
        return diff(u[0], self.x) + diff(u[1], self.y)

    def grad(self, u):
        if isinstance(u, Matrix):
            return Matrix([[diff(u[0], self.x), diff(u[0], self.y)], \
                           [diff(u[1], self.x), diff(u[1], self.y)]])
        return Matrix([diff(u, self.x), diff(u, self.y)])

    def curl(self, u):
        return diff(u[1], self.x) - diff(u[0], self.y)
```

Figure 5.2: Implementation of a 2D Cartesian coordinate system which overrides the abstract methods `dim`, `space_vars`, `div`, `grad` and `curl` of its base class `CoordinateSystem`.

element face normals' $\mathbf{n}_\kappa$ vector representation is calculated by the `Mesh.face_normals()` method, with each component containing a sympy symbolic representation of each orthogonal component.

The given domain for a finite element problem is represented in the AptoPy `Mesh` class with the restriction that the exterior boundary must be interpolated as a piecewise linear polynomial in $\mathbb{R}^d$. The boundary $\partial\Omega$ of the computational domain Ω is divided into a set of non-overlapping segments $\{\partial\Omega_i\}_{i=1}^{m_\Omega}$, such that $\partial\Omega = \bigcup_i \partial\Omega_i$ and $\bigcap_i \partial\Omega_i = \emptyset$. Each segment $\partial\Omega_i$, $i = 1, \ldots, m_\Omega$, is given a string representing its name; each of these named boundary components is then allocated a piecewise linear expression of the form $f_{\partial\Omega_i}(\mathbf{x})$, such that the condition $\left|f_{\partial\Omega_i}(\mathbf{x})\right| \leq \epsilon$, $0 < \epsilon \ll 1$, ade-

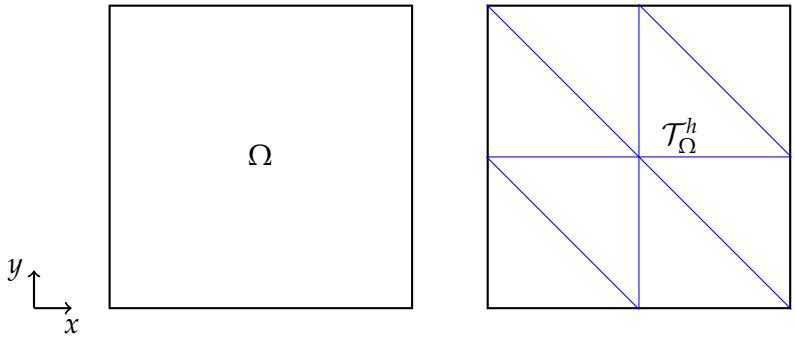


Figure 5.3: Domain $\Omega \subset \mathbb{R}^2$ and its triangulation into conforming elements κ such that the mesh $\mathcal{T}_\Omega^h = \{\kappa\}$.

```
mesh.add_boundary_definition('bottom', Abs(y) <= tol)
mesh.add_boundary_definition('right', Abs(x - 1.0) <= tol)
mesh.add_boundary_definition('top', Abs(y - 1.0) <= tol)
mesh.add_boundary_definition('left', Abs(x) <= tol)
```

Figure 5.4: AptoPy code defining a unit square domain for a given tolerance `tol`.

quately describes whether the spatial location $\mathbf{x}$ lies on the boundary component $\partial\Omega_i$, $i = 1, \ldots, m_\Omega$, within a given numerical tolerance ϵ. For example, the case of the unit square domain is shown in Figure 5.4. Although AptoPy currently requires element faces to be described by piecewise linear functions, potential future functionality is not restricted from incorporating piecewise quadratic and other higher-order curved boundaries.

The `Mesh` class generates a symbolic representation of the whole boundary $\partial\Omega$ of the domain Ω with a call to `dS = mesh.boundary()`. For individual boundaries, the boundary integration elements are constructed by the `Mesh` class by providing the required name in a call to `mesh.get_boundary_element()`. For example, the left side of the unit square $\partial\Omega_{\text{left}}$ defined in the code example in Figure 5.4 is obtained using `dS_left = mesh.get_boundary_element('left')`.

Operations to find the union and exclusion of these boundary elements are implemented via the addition `+` and subtraction `-` operators, respectively. For example, the AptoPy equivalent of $\partial\Omega_{\text{left}} \cup \partial\Omega_{\text{top}}$ would be `dS_left + dS_top`, where `dS_top = mesh.get_boundary_element('top')`, and the equivalent of $\partial\Omega \setminus \partial\Omega_{\text{left}}$ would be `dS - dS_left`. These boundary integration elements can then be used in the implementation of Dirichlet and Neumann boundary conditions, as will be demonstrated in the next sections.

The diverse range of properties of the `Mesh` class required to produce the appropriate code for AptoFEM to generate the mesh (such as element type, external mesh generation package, characteristic lengths, etc.) are stored in a hash table. For example, choosing the element type of a mesh to be simplex would require the key-value definition of `mesh.define_property('element_type', 'simplex')`. This somewhat non-specific design allows for future extensibility, especially when incorporating new external mesh generation libraries into AptoFEM.

5.1.4 Function Spaces and Dirichlet Boundary Conditions

Let the continuous finite element space of piecewise polynomials defined on the partition $\mathcal{T}_\Omega^h$ of the domain Ω be defined by

$$V_{h,\ell}(\Omega) = \left\{ v \in C(\Omega) : v|_\kappa \in \mathcal{P}^\ell(\kappa) \; \forall \kappa \in \mathcal{T}_\Omega^h \right\}, \tag{5.1.1}$$

in which the solution to a finite element problem is sought.

Consider the Poisson equation defined on a domain Ω whose boundary $\partial\Omega$ is split into two components $\partial\Omega_D$ upon which a Dirichlet condition is enforced, and $\partial\Omega_N$, where a natural Neumann condition is specified: find u such that

$$-\nabla^2 u = f \text{ in } \Omega, \tag{5.1.2}$$

$$u = g_D \text{ on } \partial\Omega_D, \tag{5.1.3}$$

$$\nabla u \cdot \mathbf{n} = g_N \text{ on } \partial\Omega_N, \tag{5.1.4}$$

where $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ and $\partial\Omega_D \cap \partial\Omega_N = \emptyset$.

The finite element formulation is to find the finite element solution u_h in the space of piecewise polynomials of a given order ℓ, i.e.,

$$u_h \in V_{h,\ell}^E(\Omega) := \left\{ v \in V_{h,\ell}(\Omega) : v|_{\partial\Omega_D} = g_D \right\} \tag{5.1.5}$$

such that

$$a_h(u_h, v_h) = l_h(v_h) \tag{5.1.6}$$

for all v_h in the space of piecewise polynomials which vanish at the Dirichlet boundary $\partial\Omega_D$, i.e.,

$$v_h \in V_{h,\ell}^{E_0}(\Omega) := \left\{ v \in V_{h,\ell}(\Omega) : v|_{\partial\Omega_D} = 0 \right\}. \tag{5.1.7}$$

Here, the bilinear functional $a_h : V_{h,\ell}^E(\Omega) \times V_{h,\ell}^{E_0}(\Omega) \to \mathbb{R}$ is

$$a_h(u_h, v_h) = \int_\Omega \nabla u_h \cdot \nabla v_h \, \mathrm{d}\mathbf{x}, \tag{5.1.8}$$

and the linear functional $l_h : V_{h,\ell}^{E_0}(\Omega) \to \mathbb{R}$ is given by

$$l_h(v_h) = \int_\Omega f v_h \, \mathrm{d}\mathbf{x} + \int_{\partial\Omega_N} g_N v_h \, \mathrm{d}s. \tag{5.1.9}$$

Mimicking mathematical notation for variational problems, the function spaces of the weak formulation must be appropriately defined with their required bases, polynomial order and whether they are formed based on H^1, L_2, Raviart-Thomas, Nédélec or other types of elements.

In AptoPy the finite element function space is represented by an instantiation of the `FemFunctionSpace` class which requires a given `Mesh` object, polynomial order and finite element type (dictating the numerical scheme) as its construction arguments. For example, the space of continuous piecewise quadratic polynomials,

$$V_{h,2}^E(\Omega) = \left\{ v \in V_{h,2}(\Omega) : v|_{\partial\Omega_D} = g_D \right\} \tag{5.1.10}$$

is created in AptoPy using

```
V_h2 = FemFunctionSpace(mesh, poly_order=2, element_type='CG').
```
(5.1.11)

Any Dirichlet conditions which are to be strongly enforced on this function space must be declared. Referring back to the naming scheme of boundary components of $\partial\Omega$ in the `Mesh` class in Section 5.1.3, a named component of the boundary such as `dS_D = mesh.get_boundary_element('left')` can be chosen. A Dirichlet condition is then enforced in the finite element space by calling `V_h2.dirichlet(dS_D, g_D)`. The trial function, acquired by calling `u = V_h2.trial()`, then satisfies $u_h \in V_{h,2}^E(\Omega)$. Furthermore, this implicitly ensures that the test function associated with $v_h \in V_{h,2}^{E_0}(\Omega)$ acquired from `v = V_h2.test()`, vanishes on the Dirichlet boundary.

For more complicated problems, AptoPy has the classes `FemVectorFunctionSpace` when vector valued trial and test functions are required, along with the complex function spaces `FemComplexFunctionSpace` and `FemComplexVectorFunctionSpace`. In combination with the ability to mix function spaces via the `FunctionSpaceProduct` class which overloads the multiplication operator $*$, finite element spaces such as those required for Taylor Hood [137] elements, e.g., $\mathbf{V}_{h,2}(\Omega) \times V_{h,1}(\Omega)$, can be constructed as shown in Figure 5.5.

```
V = FemVectorFunctionSpace(mesh, poly_order=2, element_type='CG')
Q = FemFunctionSpace(mesh, poly_order=1, element_type='CG')
TH = V * Q
```

Figure 5.5: Example of constructing Taylor Hood finite element function space $\mathbf{V}_{h,2}(\Omega) \times Q_{h,1}(\Omega)$ using AptoPy.

5.1.5 Finite Element Formulation and Neumann Boundary Conditions

AptoPy requires that the finite element formulation be expressed in terms of a residual. For example, the residual formulation of the finite element method defined in (5.1.6) is to find $u_h \in V^E_{h,\ell}(\Omega)$ such that

$$\mathcal{R}_h\left(u_h, v_h\right) = 0 \tag{5.1.12}$$

for all $v_h \in V^{E_0}_{h,\ell}(\Omega)$, where

$$\mathcal{R}_h\left(u_h, v_h\right) \equiv a_h\left(u_h, v_h\right) - l_h\left(v_h\right). \tag{5.1.13}$$

The process by which AptoPy parses a weak formulation to translate to Fortran code for AptoFEM relies on finding coefficient expressions of volume (d$\mathbf{x}$) and boundary (ds) integration elements. This appropriately assumes that every component of a finite element formulation is implicitly part of an integral operation. Combined with the weakly typed nature of Python, this allows for expressive means of representing a finite element formulation, examples of which are given in Figure 5.6.

The specification of Neumann boundaries is declared as part of the weak formulation. For example, the Neumann boundary condition component of the linear functional in (5.1.9) is a component of the weak formulation, and is therefore written in AptoPy as `g_N*v*dS_N`.

5.1.6 Function Encapsulation: The Cost of Symbolic Differentiation

A disadvantage of AptoPy being based on the symbolic algebra package sympy is the computational cost of evaluating the symbolic derivatives of functions and expressions. This cost is prevalent for nonlinear PDEs, where the Gâteaux derivative of the system must be evaluated. AptoPy alleviates this problem by encapsulating user specified expressions into `AptoFunctions`. `AptoFunctions`, when translated for AptoFEM, exist as a set of user friendly and human readable Fortran functions in a single module. This also allows AptoPy to cache the results of operations on `AptoFunctions` to reduce the

```
residual = dot(grad(u), grad(v))*dx - f*v*dx - g_N*v*dS_N
```

```
def a(u, v):
    return dot(grad(u), grad(v))*dx
def l(v):
    return f*v*dx + g_N*v*dS_N
residual = a(u, v) - l(v)
```

Figure 5.6: Examples of the 'expressiveness' of the symbolic representation of the finite element formulation in equation (5.1.6). The first simply writing out the residual equation, and the second emulating the bilinear functional formulation by defining Python functions `a(u, v)` and `l(v)`.

computational expense of generating code to solve a given finite element formulation. The performance benefit offered by the `AptoFunction` optimisation will be discussed in detail later in Section 6.6.

For example, when a function such as `grad()` is called on an `AptoFunction` which encapsulates an expression, a new `AptoFunction` is generated which encapsulates the derivative of its generator's expression as its own. This calculation is only performed once per `AptoFunction` and derivative variable, after which it is stored in a cache. Any further attempts to calculate the same derivative will simply return a reference to the previous evaluation from the cache.

Instantiating an `AptoFunction` requires three arguments: the expression itself, the arguments of the function and a `string` name. For example, consider the simple function $f(x) = x^2$; this can be encapsulated by an `AptoFunction` by calling `f = AptoFunction(x **2, x, 'f')`.

Consider the Poisson equation defined on the domain Ω with diffusion coefficient $D(\mathbf{x})$, i.e.,

$$-\nabla \cdot (D(\mathbf{x})\nabla u) = f \text{ in } \Omega, \tag{5.1.14}$$

$$u = g_D \text{ on } \partial\Omega_D, \tag{5.1.15}$$

$$(D(\mathbf{x})\nabla u) \cdot \mathbf{n} = g_N \text{ on } \partial\Omega_N. \tag{5.1.16}$$

The finite element formulation is given by: find $u_h \in V^E_{h,\ell}(\Omega)$ such that

$$\int_\Omega D\nabla u_h \cdot \nabla v_h \, \mathrm{d}\mathbf{x} = \int_\Omega f v_h \, \mathrm{d}\mathbf{x} + \int_{\partial\Omega_N} g_N v_h \, \mathrm{d}s \tag{5.1.17}$$

```
D = AptoFunction(x**2 + y**2 + 1, (x, y), 'diffusion_coeff')
residual = D*dot(grad(u), grad(v))*dx - f*v*dx - g_N*v*dS
```

```
real(db) function diffusion_coeff(x, y)
    real(db), intent(in) :: x, y
    diffusion_coeff = x**2 + y**2 + 1
end function diffusion_coeff
```

Figure 5.7: An example of the AptoPy implementation of a diffusion coefficient encapsulated by an `AptoFunction` as required by the finite element problem in equation (5.1.17), along with the resulting generated Fortran code.

for all $v_h \in V_{h,\ell}^{E_0}(\Omega)$. An example of choosing the diffusion coefficient to be $D(\mathbf{x}) = x^2 + y^2 + 1$ and declaring it in the AptoPy code as an `AptoFunction` is given in Figure 5.7.

In the case that an expression should have a non-computable derivative, such as for the $\min(\cdot,\cdot)$ and $\max(\cdot,\cdot)$ functions, AptoPy offers `AptoEvaluations`. This class is implemented in exactly the same way as `AptoFunction`, except that every derivative will be nullified to return the symbolic representation of 0. For example, this is used for the dissipation term α of the local-Lax Friedrichs flux (3.4.10).

In the case that custom Fortran code should be inserted into a function, AptoPy offers `AptoCustomFunction`. This class is implemented in the same way as `AptoEvaluation`; however, instead of requiring an expression as the first argument of its constructor, it requires a `string` representation of the Fortran code to be used. Just as with the `AptoFunction` and `AptoEvaluation`, this custom function can be manipulated as a mathematical function in sympy expressions. An example of an `AptoCustomFunction` is given in Figure 5.8.

5.1.7 Parameters

In a similar vein to the `AptoFunction` class, the `AptoParameter` class allows for symbolic representations of real and integer valued parameters. This ensures the symbolic representation of a finite element formulation in AptoPy remains identical irrespective of whether the underlying numeric value assigned to an `AptoParameter` changes. The `AptoParameter` class also allows the underlying numeric value to change between individual calculations of a finite element solution, which is key to the implementation of

```
fortran_code = '''
real(db) function custom_func_name(arg1,arg2)
    real(db) :: arg1, arg2
    custom_func_name = arg1**2 - Max(arg1, arg2)
end function custom_func_name'''
f = AptoCustomFunction(fortran_code, (x, y), 'custom_func_name')
```

Figure 5.8: An example of the AptoPy implementation of a Fortran function encapsulated by an `AptoCustomFunction`. This can then be manipulated as if it were a mathematical function with its given arguments in AptoPy, however; its derivative is defined as zero.

continuation or changing time step sizes in time discretisation methods, for example.

5.1.8 Function Traces

For some discretisation schemes, such as the DG finite element method, traces of functions are required on inter-element boundaries. Consider the bounded domain Ω with boundary $\partial\Omega$ subdivided into a shape regular mesh $\mathcal{T}_\Omega^h$ of non-overlapping elements κ such that $\mathcal{T}_\Omega^h = \{\kappa\}$. The interior boundary $\Gamma_\mathcal{I}$ is chosen to be the union of the common interior faces $\partial\kappa^+ \cap \partial\kappa^-$ of all pairs of neighbouring elements $\kappa^+, \kappa^- \in \mathcal{T}_\Omega^h$. Recalling the definition of the 'broken' Sobolev space from Section 3.2.1

$$H^s(\mathcal{T}_\Omega^h) = \{v \in L_2(\Omega) : v|_\kappa \in H^s(\kappa), \kappa \in \mathcal{T}_\Omega^h\}, \tag{5.1.18}$$

a function $u \in H^1(\mathcal{T}_\Omega^h)$, when evaluated from the interior of element κ^+, is denoted by u^+ and from the interior of the neighbouring element κ^- by u^-. These function traces evaluated on the 'skeleton' of the mesh are symbolically represented in AptoPy by calling the function `elementwise()`.

For example, given a `mesh` upon which a finite element function space is defined `V = FemFunctionSpace(mesh)`, its trial function `u = V.trial()` can have its traces represented by calling `u_p, u_m = elementwise(u)`. These symbolic representations in AptoPy correspond to u^+ = `u_p` and u^- = `u_m`.

The function `elementwise()` is not limited to simple arguments such as test functions. Entire expressions including `AptoFunctions` can be passed as arguments. These expressions are then parsed to find which components can be multivalued on inter-element boundaries, and their local and neighbouring trace representations will be calculated.

5.1.9 Jumps and Averages

We refer to the jump $[\![\cdot]\!]$, average $\{\!\{\cdot\}\!\}$, tensor jump $\underline{[\![\cdot]\!]}$ and tangential jump $[\![\cdot]\!]_{\mathrm{T}}$ operators defined in Section 3.5 which determine their output based on whether they are evaluated on $\partial\Omega$ or the interior $\Gamma_{\mathcal{I}}$ boundary of a mesh. The symbolic representation encapsulates this behaviour in the `jump()`, `avg()`, `tensor_jump()` and `cross_jump()` functions, respectively, based on whether they are parsed to be coefficients of the interior faces integration element `mesh.interior_faces()`, or an exterior component of the boundary `mesh.boundary()`.

5.2 Geometry Representation

5.2.1 The `Polygon` Class

Although the concept of a symbolic representation of a mesh $\mathcal{T}_\Omega^h$ has already been introduced in Section 5.1.3, generation of the geometry of its parent domain Ω is not specified. The use and generation of domains and meshes is not restricted to any one package by AptoPy, however AptoPy offers a means to construct domains $\Omega \subset \mathbb{R}^2$ with the `Polygon` class and thereby construct a mesh of that geometry, $\mathcal{T}_\Omega^h$, by interfacing with the mesh generation package Triangle [124].

The `Polygon` class requires that all exterior boundary faces of a domain can be represented by piecewise linear polynomials, and that all points be specified in a Cartesian geometry $\mathbf{x} = (x, y)$. The `Polygon` class stores and manages pairs of points which describe the boundary $\partial\Omega$ of a domain Ω. Functions can subsequently be applied to all points of the `Polygon` allowing for translation, dilation, rotation and other such geometric operations. An example of implementing the `Polygon.add_line()` and `Polygon.line_to()` functions to construct a unit square is demonstrated in Figure 5.9. For each pair of points $\mathbf{a}$, $\mathbf{b}$, passed to `Polygon.add_line()` the symbolic piecewise linear polynomial bounding equation $f_{\partial\Omega_i}(\mathbf{x})$ of the boundary component $\partial\Omega_i$ can automatically be computed. Here,

$$f_{\partial\Omega_i}(\mathbf{x}) = \begin{cases} y - mx + c & \text{if } \mathbf{b}_x - \mathbf{a}_x > 0, \\ x - \mathbf{b}_x & \text{otherwise,} \end{cases} \tag{5.2.1}$$

where

$$m = \frac{\mathbf{b}_y - \mathbf{a}_y}{\mathbf{b}_x - \mathbf{a}_x}, \quad c = \mathbf{b}_y - m\mathbf{b}_x, \tag{5.2.2}$$

```
poly = Polygon()
poly.add_line((0.0, 0.0), (1.0, 0.0), name='bottom')
poly.line_to((1.0, 1.0), name='right')
poly.line_to((0.0, 1.0), name='top')
poly.line_to((0.0, 0.0), name='left')
```

Figure 5.9: Generating a unit square using the `Polygon` class.

which upon constructing a `Mesh` object is used in calling `Mesh.add_boundary_definition()` for an appropriate numerical tolerance close to machine precision (see Figure 5.4).

5.2.2 Error Control of the Piecewise Linear Boundary Description

Let $\mathcal{K}(x)$ be a function defined on the interval $x \in [a,b]$ which describes a component of the boundary of $\Omega \subset \mathbb{R}^2$, where $a, b \in \mathbb{R}$ and $a < b$. We wish to interpolate $\mathcal{K}$ by continuous piecewise linear polynomials such that $\mathcal{K}$ can be approximated using the `Polygon` class. We denote the subdivision of $[a,b]$ into element subintervals κ each of length h_κ as $\mathcal{T}^h_{[a,b]} = \{\kappa\}$. We define the single continuous piecewise linear polynomial interpolating $\mathcal{K}$ as

$$\mathcal{K}_h := \left\{ v \in C([a,b]) : v|_\kappa \in \mathcal{P}^1(\kappa),\ v(x)|_{\partial\kappa} = \mathcal{K}(x)|_{\partial\kappa}\ \forall \kappa \in \mathcal{T}^h_{[a,b]} \right\}. \tag{5.2.3}$$

The automatic definition of exterior boundary components with pairs of points discussed in Section 5.2.1 allows for error control on curved sections of the exterior boundary geometry. The L_2 error of the interpolation estimate on element κ is denoted by

$$\mathrm{e}_\kappa := \|\mathcal{K} - \mathcal{K}_h\|_{L_2(\kappa)}. \tag{5.2.4}$$

We also introduce the interpolation error on the interval $[a,b]$ by

$$\mathrm{e} := \|\mathcal{K} - \mathcal{K}_h\|_{L_2([a,b])} = \left(\sum_{\kappa \in \mathcal{T}^h_{[a,b]}} \mathrm{e}_\kappa^2 \right)^{\frac{1}{2}}. \tag{5.2.5}$$

We seek an interpolant $\mathcal{K}_h$ of curve $\mathcal{K}$ which satisfies a given numerical tolerance $\mathrm{e} \leq \mathtt{TOL}$. We employ an h-refinement scheme to adapt the subdivision $\mathcal{T}^h_{[a,b]}$ into successively finer meshes until the interpolation error criterion is fulfilled. The outline of this procedure as computed in AptoPy is given in Algorithm 1. An example for the domain

$$\Omega = \left\{ \mathbf{x} : y > x^2,\ \ y < 1,\ \ 0 < x < 1 \right\} \tag{5.2.6}$$

with curve interpolation `add_curve_by_l2_error((0.0, 0.0), (1.0, 1.0), lambda x: x**2, tol=tol)` is demonstrated in Figure 5.10 for various tolerances. Here the arguments `(0.0, 0.0)` and `(1.0, 1.0)` are the start and end points $(a, \mathcal{K}(a))$ and $(b, \mathcal{K}(b))$, respectively, `lambda x: x**2` is the expression $y = x^2$ and `tol` is the numerical tolerance of the interpolation error.

Algorithm 1 Compute exterior boundary geometry interpolation estimate.

while e $\geq$ `TOL` **do**

For each $\kappa \in \mathcal{T}^h_{[a,b]}$ compute e_κ

Choose refinement fraction $\chi \in (0, 1]$

Compute the $\left\lceil \chi \left| \mathcal{T}^h_{[a,b]} \right| \right\rceil$ largest values of e_κ, $\forall \kappa \in \mathcal{T}^h_{[a,b]}$ denoted by E

Let $\mathcal{T}^h_{\text{old}} = \{\kappa : \mathrm{e}_\kappa \in \mathrm{E}\}$

Compute $\mathcal{T}^h_{\text{new}} = \left\{\text{bisect}\,(\kappa)\ \ \forall \kappa \in \mathcal{T}^h_{\text{old}}\right\}$

Compute fine mesh $\mathcal{T}^h_{\text{fine}} = \left(\mathcal{T}^h_{[a,b]} \setminus \mathcal{T}^h_{\text{old}}\right) \cup \mathcal{T}^h_{\text{new}}$

Reassign $\mathcal{T}^h_{[a,b]} = \mathcal{T}^h_{\text{fine}}$

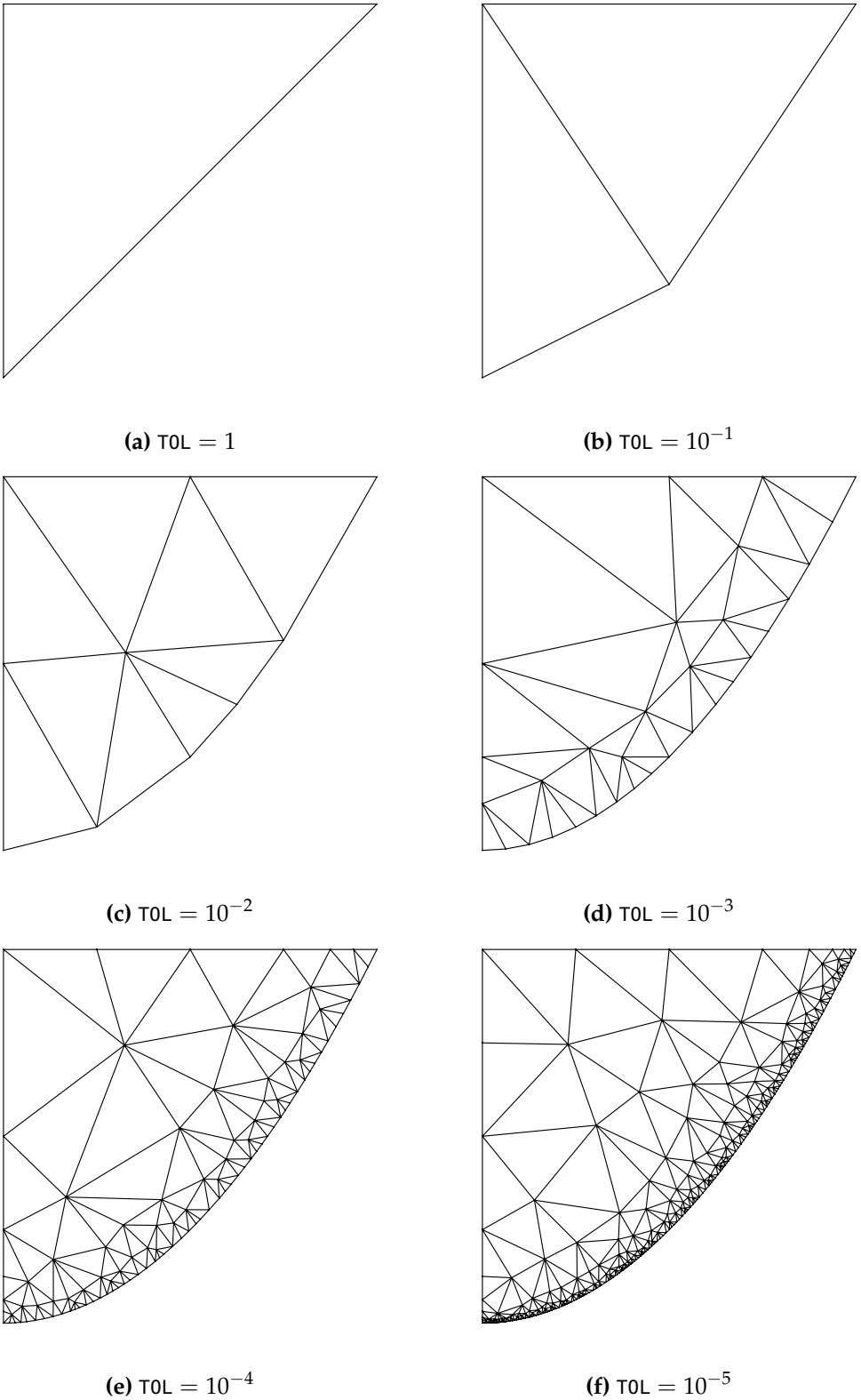
Compute e

end while

5.2.3 Subdomains and Interface Boundaries

In the specification of a `Polygon`, it may be required to indicate relevant subdomains. In the context of the MPA-CVD reactor geometry, these subdomains characterise the air filled cavity, quartz window and hydrogen vacuum regions. `Polygon` allows for specification of subdomains using `int` identifiers by calling `Polygon.define_region(location, region_number)`. Here, the `location` argument is a point in $\mathbb{R}^2$ which is enclosed by the boundary of a subdomain which has been specified in the `Polygon` instance. The `region_number` argument is the `int` identifier assigned to that subdomain. The `Polygon` instance then assumes that when a mesh is generated, all elements in the volume enclosing the `location` specified should be categorised by assigning to each the same identifier `region_nubmer`.

Once a `Mesh` object has been instantiated from the provided `Polygon` with these subregions, the interior interfaces can be requested. For two subdomain regions `subdomain1` and `subdomain2`, the symbolic form of the integration element on their interface is acquired from `dInt_12 = mesh.region_interface(subdomain1, subdomain2)`. AptoPy automatically evaluates any DG jump and average operators as interior or exterior based on whether their arguments exist on both or only one side of the interface. AptoPy



(a) TOL $= 1$

(b) TOL $= 10^{-1}$

(c) TOL $= 10^{-2}$

(d) TOL $= 10^{-3}$

(e) TOL $= 10^{-4}$

(f) TOL $= 10^{-5}$

Figure 5.10: Examples of meshes of the domain $\Omega = \{\mathbf{x} : y > x^2,\ \ y < 1,\ \ 0 < x < 1\}$ generated from a `Polygon` class instance, whose exterior boundary is defined using `Polygon.add_curve_by_l2_error()`. The meshes of the volume of the domain are generated by interfacing with the mesh generation package, Triangle [124].

further handles the orientation of the face normal vectors to automatically point from the region of highest integer identifier into the lower.

5.3 Discontinuous Galerkin Utility Functions

5.3.1 Penalisation Parameter δ

Recall the penalisation parameter of the interior penalty method (3.5.17), repeated here for convenience

$$\delta(\mathbf{u}_h) = C_{\mathrm{IP}} \frac{\ell^2}{h_F} \{\!\{ G(\mathbf{u}_h) \}\!\} \underline{[\![\mathbf{u}_h]\!]}. \tag{5.3.1}$$

The component $\frac{\ell^2}{h_F}$ is explicitly calculated in AptoFEM for a finite element solution of polynomial degree ℓ on each element with diameter measuring h_F. This quantity is represented in AptoPy as a property of the `FemFunctionSpace` class, `FemFunctionSpace.penalisation()`. The positive constant C_{IP} can be selected by the user. For example, choosing $C_{\mathrm{IP}} = 20.0$ requires `20.0*V.penalisation()`. In the case of finite element vector function spaces or function space products, for example $\mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$, `FemFunctionSpace.penalisation()` is the vector

$$\frac{\ell_j^2}{h_F}, \quad j = 1, \ldots, d, \tag{5.3.2}$$

where ℓ_j is the polynomial degree of finite element function space component $\left(\mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)\right)_j$.

5.3.2 Automatic Treatment of Convective Terms

Recall the definition of the local Lax-Friedrichs flux from Section 3.4, i.e.,

$$\mathcal{H}_{\mathrm{LF}}\left(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa\right)\big|_{\partial\kappa} := \tfrac{1}{2}\left(\mathcal{F}^c\left(\mathbf{u}_h^+\right) \cdot \mathbf{n}_\kappa + \mathcal{F}^c\left(\mathbf{u}_h^-\right) \cdot \mathbf{n}_\kappa + \alpha\left(\mathbf{u}_h^+ - \mathbf{u}_h^-\right)\right). \tag{5.3.3}$$

The functional $\mathcal{H}_{\mathrm{LF}}(\cdot,\cdot,\cdot)$ and dissipation parameter α can be automatically generated in AptoPy for the convective components $\mathcal{F}^c(\cdot)$ of a PDE. Calling `H = lax_friedrichs_flux(F_c, alpha)` for a callable function `F_c()` and symbolic representation of `alpha`, the generated function `H()` takes three arguments. These arguments correspond to the mathematical representation of the local Lax-Friedrichs flux $\mathcal{H}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}_\kappa)$. Consider the application of this AptoPy function to the linear advection equation

$$\nabla \cdot (\mathbf{b}u) = f, \tag{5.3.4}$$

where $\mathbf{b} \in \mathbb{R}^d$ and $\mathcal{F}^c(u) = \mathbf{b}u$. Here, the dissipation parameter can be shown to be $\alpha|_{\partial\kappa} = |\mathbf{b} \cdot \mathbf{n}_\kappa|$. The AptoPy code example for this problem is given in Figure 5.11.

```
alpha = Abs(dot(b,n))
def F_c(u): return b*u
H = lax_friedrichs_flux(F_c, alpha)
interior_residual = H(u_p, u_m, n)*(v_p - v_m)*dInt
```

Figure 5.11: Example of the AptoPy automatic calculation and symbolic representation of the local Lax-Friedrichs flux $\mathcal{H}(u^+, u^-, \mathbf{n}_\kappa)$ as required for the linear advection equation shown in (5.3.4)

In some cases the dissipation parameter α cannot be so easily computed analytically. Recall that for system flux Jacobian

$$B(\mathbf{u}, \mathbf{n}_\kappa) := \sum_{i=1}^{d} \frac{\partial \mathbf{f}_i^c}{\partial \mathbf{u}} \mathbf{n}_{\kappa,i}, \tag{5.3.5}$$

the local Lax-Friedrichs dissipation parameter is defined to be the largest magnitude eigenvalue, i.e.,

$$\alpha|_{\partial\kappa} = \max_{\mathbf{w}=\mathbf{u}_h^+,\mathbf{u}_h^-} \left\{ |\lambda \left(B \left(\mathbf{w}, \mathbf{n}_\kappa \right) \right)| \right\}. \tag{5.3.6}$$

AptoPy offers the utility functions `flux_jacobian()` and `flux_jacobian_eigen_values()` to automatically compute the local Lax-Friedrichs dissipation parameter. AptoPy's `flux_jacobian()` function takes three arguments: the hyperbolic flux $\mathcal{F}^c(\cdot)$, the solution vector $\mathbf{u}$ and the element face normal $\mathbf{n}_\kappa$. By calling `flux_jacobian(F_c, u, n)`, AptoPy employs the symbolic differentiation of sympy to formulate the matrix $B(\mathbf{u}, \mathbf{u}_\kappa)$. Subsequently calling `flux_jacobian_eigen_values()`, passing the flux Jacobian matrix as the argument, further employs sympy to compute and solve the characteristic polynomial of the matrix via the Berkowitz algorithm [16]. An example of using this method to compute the local Lax-Friedrichs dissipation parameter for the convective component of the Navier-Stokes equations $\mathcal{F}^c(\mathbf{u}) = \rho \mathbf{u} \otimes \mathbf{u}$ is presented in Figure 5.12.

5.3.3 Automatic Treatment of Viscous Terms

Due to the consistent treatment of the elliptic second order terms of a DG finite element formulation, AptoPy takes advantage of this by offering utility functions to automatically generate their semilinear residual formulations. Recall from Chapter 3 that the viscous component of a PDE can be written as

$$-\nabla \cdot \mathcal{F}^v \left(\mathbf{u}; \nabla \mathbf{u} \right) = \mathbf{0}. \tag{5.3.7}$$

```
def F_c(u):
    return rho*u*u.T
# Calculate the eigenvalues for the dissipation parameter
B = flux_jacobian(F_c, u, n)
lambdas = flux_jacobian_eigen_values(B)
# Calculate the maximum of their magnitudes and encapsulate
lambdas_p, lambdas_m = elementwise(map(Abs, lambdas))
maximum_lambda = Max(*lambdas_p.row_join(lambdas_m))
alpha = apto_evaluation(maximum_lambda, (u_p, u_m, n), 'alpha')
```

Figure 5.12: Automatic symbolic algebra computation of the dissipation parameter required by the local Lax-Friedrichs flux appled to the Navier-Stokes convective flux component $\mathcal{F}^c(\mathbf{u}) = \rho \mathbf{u} \otimes \mathbf{u}$.

```
def F_v(u):
    return grad(u)
vt = DGFemViscousTerm(F_v, u, v, penalty)
```

Figure 5.13: Application of the `DGFemViscousTerm` class to the elliptic operator of the Poisson equation $-\nabla^2 u = f$ where $\mathcal{F}^v(u; \nabla u) = \nabla u$.

The semilinear DG discretisation of the viscous (elliptic) term of this equation, derived in equation (3.5.15), is repeated here for convenience

$$\begin{aligned}\mathcal{N}_\Omega^v(\mathbf{u}_h; \mathbf{v}_h) := &\int_\Omega \mathcal{F}^v(\mathbf{u}_h; \nabla_h \mathbf{u}_h) : \nabla_h \mathbf{v}_h \, \mathrm{d}\mathbf{x} - \int_{\Gamma_\mathcal{I} \cup \partial\Omega} \{\!\{\hat{\sigma}_h\}\!\} : \underline{[\![\mathbf{v}_h]\!]} \, \mathrm{d}s \\ &- \int_{\Gamma_\mathcal{I}} [\![\hat{\sigma}_h]\!] \cdot \{\!\{\mathbf{v}_h\}\!\} \, \mathrm{d}s + \int_{\Gamma_\mathcal{I} \cup \partial\Omega} \underline{[\![\hat{\mathbf{u}}_h - \mathbf{u}_h]\!]} : \{\!\{G^\top(\mathbf{u}_h) \nabla_h \mathbf{v}_h\}\!\} \, \mathrm{d}s \\ &+ \int_{\Gamma_\mathcal{I}} \{\!\{\hat{\mathbf{u}}_h - \mathbf{u}_h\}\!\} \cdot [\![G^\top(\mathbf{u}_h) \nabla_h \mathbf{v}_h]\!] \, \mathrm{d}s = 0. \quad (5.3.8)\end{aligned}$$

This semilinear formulation can be calculated in AptoPy by calling `DGFemViscousTerm (F_v, u, v, penalty)` for scalar or vector trial and test functions `u` and `v`, respectively, DG penalty quantity `penalty` $= C_{\mathrm{IP}} \frac{\ell^2}{h_F}$ = `C_IP*V.penalisation()` used in the automatic formulation of the interior penalty parameter, and callable function

$$\texttt{F\_v(u)} = \mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u}). \quad (5.3.9)$$

An example of the application of the `DGFemViscousTerm` class for the Poisson equation is given in Figure 5.13. Although `DGFemViscousTerm` currently only implements the interior penalty method, this is not a restriction, and future support for other methods is possible.

The `DGFemViscousTerm` automatically generates the homogeneity tensor G, and thereby also the interior penalty parameter δ of equation (3.5.17). Recall that the homogeneity tensor $G(\mathbf{u})$ is defined for $\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) = (\mathbf{f}_1^v, \ldots, \mathbf{f}_d^v)$ by

$$G_{kl}(\mathbf{u}) = \frac{\partial \mathbf{f}_k^v}{\partial (\nabla\mathbf{u})_l}, \quad k, l = 1, \ldots, d, \tag{5.3.10}$$

such that for homogeneity tensor product

$$(G(\mathbf{u})\nabla\mathbf{u})_{ik} = \sum_{j=1}^{m}\sum_{l=1}^{d}(G_{kl}(\mathbf{u}))_{ij}(\nabla\mathbf{u})_{jl}. \tag{5.3.11}$$

AptoPy offers the function `homogeneity_tensor(F_v, u)` to automatically calculate $G(\mathbf{u})$ in equation (3.5.4) according to Algorithm 2 and the function `hyper_tensor_product(G, tau)` to compute the homogeneity tensor product according to Algorithm 3. The `DGFemViscousTerm` class uses this function, automatically constructing the homogeneity tensor storing it as a member of each instance. This is then used along with the trial and test function vectors and the penalisation parameter in the calculation of the semilinear residual stated in equation (3.5.15).

Algorithm 2 Calculating the symbolic algebra representation of the homogeneity tensor $G(\mathbf{u})$.

function HOMOGENEITY_TENSOR($\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})$, $\mathbf{u}$)
  (`F_v, grad_u`) $\leftarrow$ ($\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})$, $\nabla\mathbf{u}$)
  `G` $\leftarrow [[0]^{m\times m}]^{d\times d}$ ▷ Initialise homogeneity tensor
  for $k = 1, \ldots, d$ **do** ▷ Iterate over number of space variables
    for $l = 1, \ldots, d$ **do**
      `g` $\leftarrow [0]^{m\times m}$
      for $r = 1, \ldots, m$ **do** ▷ Iterate over number of PDE variables
        for $c = 1, \ldots, m$ **do** ▷ Perform symbolic differentiation
          `g[r, c]` $\leftarrow$ `Differentiate(F_v[r, k], grad_u[c, l])`
        end for
      end for
      `G[k, l]` $\leftarrow$ `g`
    end for
  end for
  return `G`
end function

Algorithm 3 Calculation of the homogeneity tensor product $G(\mathbf{u})\tau$.

function HOMOGENEITY_TENSOR_PRODUCT($G(\mathbf{u})$, τ)
  (G, tau) $\leftarrow (G(\mathbf{u}), \tau)$
  M $\leftarrow [0]^{m \times d}$
  for $i = 1, \ldots, m$ **do**
    for $k = 1, \ldots, d$ **do**
      xi $\leftarrow 0$
      for $l = 1, \ldots, d$ **do**
        xi $\leftarrow$ xi + MatrixVectorProduct(G[k, l], tau[:, l])[i]
      end for
      M[i, k] $\leftarrow$ M[i, k] + xi
    end for
  end for
  return M
end function

DGFemViscousTerm offers two methods for handling the boundary components of the DG discretisation in (3.5.15). DGFemViscousTerm.interior_residual() automatically generates terms associated with the interior boundaries $\Gamma_{\mathcal{I}}$; DGFemViscousTerm .exterior_residual(u_gamma, dS_i) automatically generates the terms associated with exterior boundary component dS_i with boundary condition $\mathbf{u}_\Gamma(\mathbf{u})$ = u_gamma. An example of the AptoPy code required to generate the semilinear residual form for the nonlinear Poisson equation

$$-\nabla \cdot ((u+1)\nabla u) = f \text{ in } \Omega, \tag{5.3.12}$$

$$u = 0 \text{ on } \partial\Omega, \tag{5.3.13}$$

is demonstrated in Figure 5.14.

In the case of applying the DGFemViscousTerm class in a coordinate system other than the Cartesian coordinate system, the pattern matching offered by sympy required to find components of $\nabla\mathbf{u}$ in $\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})$ when calculating the homogeneity tensor is insufficient. AptoPy therefore offers a more specialised method of calculating $G(\mathbf{u})$ in the function generalised_homogeneity_tensor(F_v_general, u) for vector u and callable function

$$\texttt{F\_v\_general(u, grad\_u)} = \mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}). \tag{5.3.14}$$

When this function is called from generalised_homogeneity_tensor(), its second argument is a matrix whose elements are symbolic representations of the vector derivative

```
def F_v(u): return (u + 1)*grad(u)
vt = DGFemViscousTerm(F_v, u, v, penalty)
residual = dot(F_v(u), grad(v))*dx \
         + vt.interior_residual(dInt) \
         + vt.exterior_residual(0.0, dS) \
         - f*v*dx
```

Figure 5.14: Example of the automatically generated discontinuous Galerkin finite element formulation of the nonlinear Poisson equation in (5.3.12) using the `DGFemViscousTerm` utility class. Compare with the linear case shown previously in Figure 5.13.

```
def F_v_general(u, grad_u):
    return grad_u
G_general = generalised_homogeneity_tensor(F_v_general, u)
vt = DGFemViscousTerm(F_v, u, v, delta, G=G_general)
```

Figure 5.15: Example of the automatically generated homogeneity tensor for the Poisson equation in (5.3.12), and its subsequent use in the instantiation of a `DGFemViscousTerm` object.

such that

$$(\nabla \mathbf{u})_{ij} = \texttt{grad\_u[i, j]}. \tag{5.3.15}$$

This simplifies the symbolic derivative calculation required by AptoPy for each coordinate system. An example of generating the generalised homogeneity tensor for the Poisson equation in (5.3.12) is shown in Figure 5.15.

5.3.4 Automatic Generation of DG Finite Element Formulations

Even with the utility functions provided by AptoPy described in Sections 5.3.2 and 5.3.3, the specification of a DG finite element formulation can be verbose, especially for large systems of PDEs with many boundary conditions. In order to manage specification of large sets of boundary conditions, AptoPy offers the `BoundaryCondition` abstract class from which the classes `DirichletBC` and `NeumannBC` inherit. These implementations simply serve to store the boundary condition and the boundary component over which the condition should be enforced. For example, applying a Dirichlet boundary condition as required by the Poisson equation in (5.1.14) of $u = g_D$ on $\partial\Omega_D$ simply requires an instantiation of `DirichletBC(dS_D, g_D)`. Constructing a series of boundary conditions in this manner and placing them in a list allows for iterative generation of exterior

```
bcs = [DirichletBC(dS_1, bc_1), DirichletBC(dS_2, bc_2), ...]
vt = DGFemViscousTerm(F_v, u, v, penalty)
ext = sum(vt.exterior_residual(bc.boundary(), bc.condition()) for bc in bcs)
```

Figure 5.16: Example of the automatic generation of the exterior residual terms of a given `DGFemViscousTerm` for a list of Dirichlet boundary conditions.

```
bcs = [DirichletBC(dS_D, g_D), NeumannBC(dS_N, g_N)]
poisson_equation = PoissonEquation(mesh, V, bcs)
residual = poisson_equation.generate_fem_formulation()
```

Figure 5.17: Example of implementing the `PoissonEquation` utility class which inherits from the `EllipticOperator` and in turn the abstract base class `FemFormulation`.

boundary terms in a finite element formulation. As an example of a series of Dirichlet boundary conditions being automatically generated using the `DGFemViscousTerm` see Figure 5.16.

This concept is further extended in AptoPy by the introduction of the abstract class `FemFormulation` which prescribes one abstract method `generate_fem_formulation()`. As its arguments, the `FemFormulation` constructor requires the symbolic representations of the PDE boundary value problem mesh, the function space and the list of boundary conditions. For example, the class `EllipticOperator` inherits `FemFormulation` which requires the extra argument at instantiation of the form of $\mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u})$. The implemented overridden method `generate_fem_formulation()` should then be written to automatically generate the volume, interior boundary and exterior boundary integration terms, implementing all of the concepts of utility functions for elliptic operators in Section 5.3.3 and Figure 5.16. In turn, the utility class `PoissonEquation` inherits from `EllipticOperator` which replaces the optional argument of `EllipticOperator` specifying $\mathcal{F}^v(\mathbf{u}; \nabla \mathbf{u})$ to the diffusion coefficient D. An example of the automatic generation of the DG formulation of the Poisson equation (5.1.14) is shown in Figure 5.17.

This class hierarchy scheme for the automated generation of DG FEM formulations implemented in AptoPy is presented in Figure 5.18. Further implementations of each member of this class hierarchy need not only be by inheritance. For example, consider an implementation of the steady state heat equation coupled to the incompressible Navier-Stokes equations. I.e., given domain $\Omega \subset \mathbb{R}^d$ with exterior boundary $\partial\Omega$ we seek the DG numerical approximations of velocity field $\mathbf{u}$ and temperature field T

such that

$$-\nabla^2\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega, \tag{5.3.16}$$

$$\nabla\cdot\mathbf{u} = 0 \quad \text{in } \Omega, \tag{5.3.17}$$

$$\mathbf{u}\cdot\nabla T - \nabla^2 T = 0 \quad \text{in } \Omega, \tag{5.3.18}$$

subject to the boundary conditions for prescribed functions $\mathbf{g}_D$ and T_D

$$\mathbf{u} = \mathbf{g}_D \quad \text{on } \partial\Omega, \tag{5.3.19}$$

$$T = T_D \quad \text{on } \partial\Omega. \tag{5.3.20}$$

The AptoPy implementation could simply be a class inheriting `FemFormulation` and aggregating single instances of the `Poisson`, `NavierStokes` and `HyperbolicOperator` classes. When overriding the abstract `generate_fem_formulation()` method, only the provision of the convective flux of the system enthalpy with the background velocity is required to be coded. The remaining is simple management between the velocity, pressure and temperature function spaces and boundary conditions amongst the aggregated `FemFormulation` members.

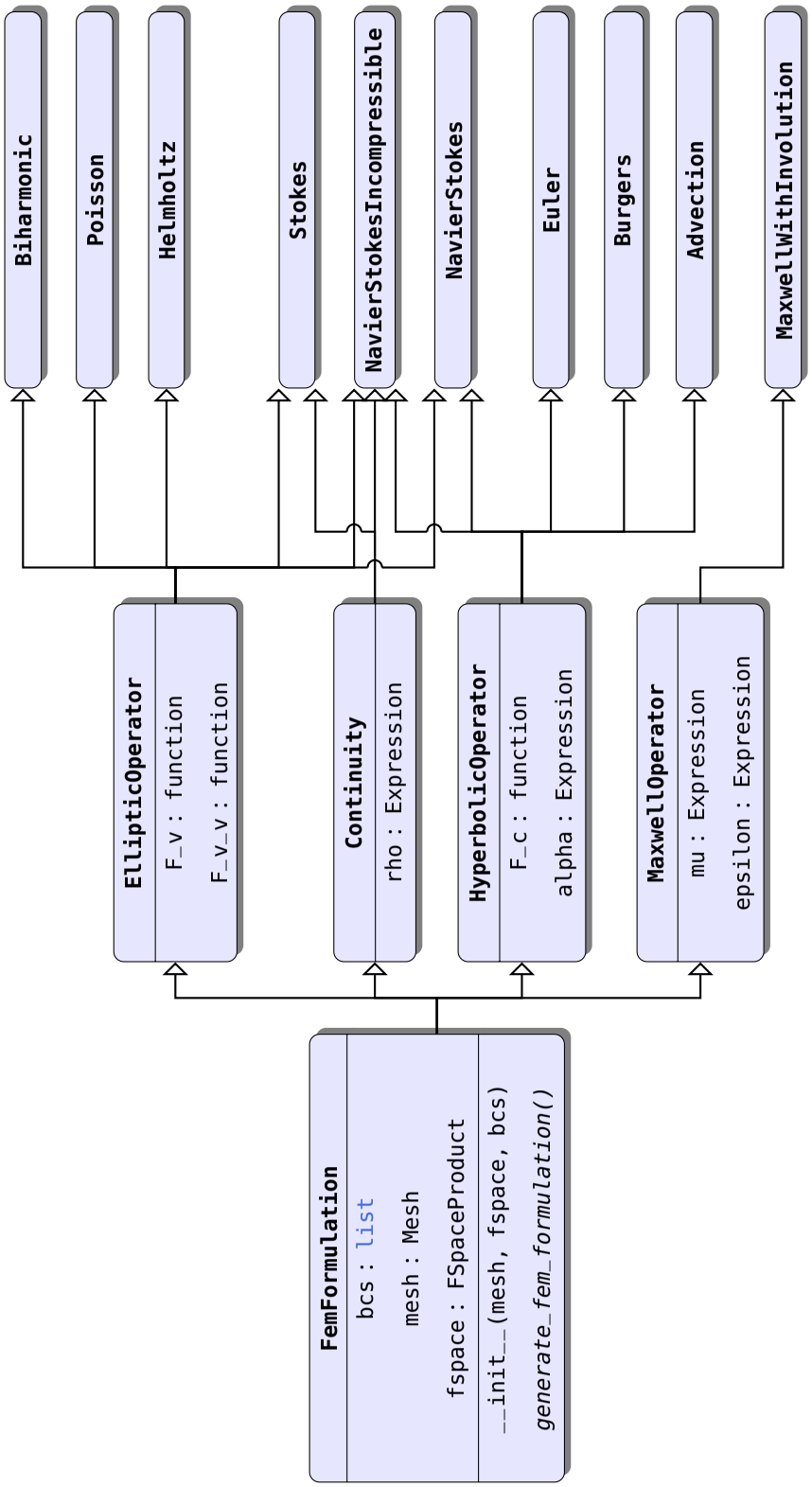


Figure 5.18: Class diagram of the DG FEM formulation hierarchy.

5.4 Solution Procedure

5.4.1 Introduction

By default, AptoPy solves every finite element formulation of a given PDE, as if it were a nonlinear problem by employing a damped Newton iterative method. Given a finite element formulation defined by the semilinear residual functional $\mathcal{N}(\mathbf{u}_h; \mathbf{v}_h)$ and its Gâteaux derivative

$$\mathcal{N}'[\mathbf{w}](\mathbf{u}; \mathbf{v}) := \lim_{\tau \to 0} \frac{\mathcal{N}(\mathbf{u} + \tau \mathbf{w}; \mathbf{v}) - \mathcal{N}(\mathbf{u}; \mathbf{v})}{\tau}, \tag{5.4.1}$$

then for damping parameter $\vartheta \in (0, 1]$, each subsequent solution estimate $\mathbf{u}_h^{n+1}$ is updated according to:

$$\mathbf{u}_h^{n+1} = \mathbf{u}_h^n + \vartheta \mathbf{d}_h^n. \tag{5.4.2}$$

Here, $\mathbf{d}_h^n$ is the update of the previous iterate $\mathbf{u}_h^n$ defined by: find $\mathbf{d}_h^n \in \mathbf{V}_h$ such that

$$\mathcal{N}'[\mathbf{d}_h^n](\mathbf{u}_h^n; v_h) = -\mathcal{N}(\mathbf{u}_h^n; \mathbf{v}_h) \quad \forall \mathbf{v}_h \in \mathbf{V}_h. \tag{5.4.3}$$

AptoFEM provides the appropriate subroutines and data structures to handle the iterative calculation of equation (5.4.2). Therefore AptoPy must parse the computational symbolic algebra representation of the finite element formulation, and generate the necessary Fortran code for compilation against AptoFEM and thereby computation of the FE solution. To this end, in this section an indexing scheme is introduced to match symbols in AptoPy with the corresponding implementation in AptoFEM. The method implemented by AptoPy of parsing a finite element solution from a sympy expression tree and then calculating its Gâteaux derivative is then demonstrated. Lastly the formulation in terms of the basis functions of each finite element function space is introduced and hence the automatic residual vector and Gâteaux derivative matrix construction in AptoFEM. Using these principles, the resulting Fortran code AptoPy generates in this translation procedure is then automatically compiled against the AptoFEM library, and the resulting executable is run to compute the numerical FE solution.

5.4.2 Indexing Function Spaces and their Associated Variables

The various variables and function spaces of a finite element formulation written in AptoPy must be correctly indexed in order to translate the symbolic form to the automatically generated Fortran code necessary for AptoFEM to compute the solution.

In the case of Cartesian coordinates, the spatial variables are indexed in the following manner

$$\mathbf{x} = (x_1, \ldots, x_d)^\top, \tag{5.4.4}$$

which in turn define the indices used in AptoFEM for the spatial coordinate vector, such that $x_j =$ `svars(j)`. Similarly, the element face unit outward normal vector

$$\mathbf{n}_\kappa = (n_1, \ldots, n_d)^\top, \tag{5.4.5}$$

corresponds to the AptoFEM array with indices $n_j =$ `face_normals(j)`.

On a bounded domain Ω with Dirichlet boundary $\partial\Omega_D$ and Neumann boundary $\partial\Omega_N$, where $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$ and $\partial\Omega_D \cap \partial\Omega_N = \emptyset$, a given finite element solution space may be the product of several finite element spaces, i.e.,

$$W(\Omega) = V^1_{h,\ell}(\Omega) \times V^2_{h,\ell}(\Omega) \times \ldots \times V^\alpha_{h,\ell}(\Omega), \tag{5.4.6}$$

where each trial function is indexed such that $u_{h,j} \in V^j_{h,\ell}(\Omega)$, $j = 1, \ldots, \alpha$. The product of finite element spaces which vanish on the Dirichlet boundary $\partial\Omega_D$ is denoted by

$$\hat{W}(\Omega) = \hat{V}^1_{h,\ell}(\Omega) \times \hat{V}^2_{h,\ell}(\Omega) \times \ldots \times \hat{V}^\alpha_{h,\ell}(\Omega), \tag{5.4.7}$$

where each test function is indexed such that $v_{h,j} \in \hat{V}^j_{h,\ell}(\Omega)$, $j = 1, \ldots, \alpha$. Furthermore, derivatives of the functions belonging to these spaces are indexed with the same scheme as used for the spatial variables, for example, $\partial u_{h,j}/\partial x_k$, $j = 1, \ldots, \alpha$, $k = 1, \ldots, d$.

5.4.3 Parsing the Residual Finite Element Formulation

Initially the residual finite element formulation expression tree is parsed to separate element and face contributions. The element components are found in coefficients of the volume integration element $\mathrm{d}\mathbf{x}$ and face components are found in coefficients of the boundary integration element $\mathrm{d}s$ for each boundary component $\partial\Omega_i$, $i = 1, \ldots, m_\Omega$, and interior faces $\Gamma_\mathcal{I}$. An example of this process is shown in Figure 5.19.

The following notation is used to denote the solution variable vector for each trial function $u_{h,j} \in V^j_{h,\ell}(\Omega)$, $j = 1, \ldots, \alpha$

$$\mathbf{u} = (u_{h,1}, \ldots, u_{h,\alpha}) \tag{5.4.8}$$

and test function vector for each test function $v_{h,j} \in \hat{V}^j_{h,\ell}(\Omega)$, $j = 1, \ldots, \alpha$

$$\mathbf{v} = (v_{h,1}, \ldots, v_{h,\alpha}). \tag{5.4.9}$$

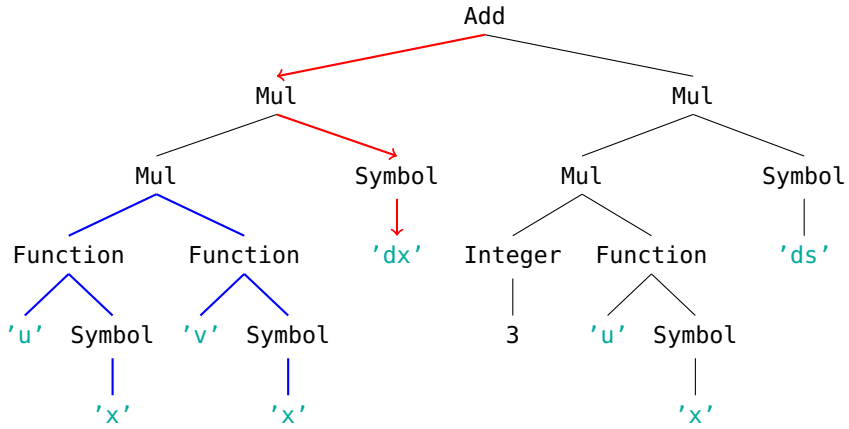


Figure 5.19: The expression tree for the AptoPy code `u*v*dx + 3*v*dS`, which is parsed for the volume integration element `dx` by walking the tree as shown by red branches. The coefficient subtree of `dx` is then extracted as shown by the blue branches.

For each function space employed in the definition of the product space $W(\Omega)$, each subtree is parsed to find coefficients of test functions $v_{h,j}$, $j = 1, \ldots, \alpha$, which are stored in the tensor $E_j^v(\mathbf{u})$ for element contribution and $B_j^{v,\partial\Omega_i}(\mathbf{u})$ for boundary contributions. Furthermore, coefficients of test function derivatives $\partial v_{h,j}/\partial x_k$, $j = 1, \ldots, \alpha$, $k = 1, \ldots, d$, are stored in the element contribution tensor $E_{j,k}^{\nabla v}(\mathbf{u})$ and boundary contribution tensor $B_{j,k}^{\nabla v,\partial\Omega_i}(\mathbf{u})$. This then allows for the semi-linear residual formulation to be written: find $\mathbf{u}_h \in W(\Omega)$ such that

$$\begin{aligned}\mathcal{N}(\mathbf{u}_h;\mathbf{v}_h) = &\int_\Omega \sum_{j=1}^{\alpha} \left(E_j^{\mathbf{v}}(\mathbf{u}_h) v_{h,j} + \sum_{k=1}^{d} E_{j,k}^{\nabla\mathbf{v}}(\mathbf{u}_h) \frac{\partial v_{h,j}}{\partial x_k} \right) \mathrm{d}\mathbf{x} \\ &+ \sum_{i=1}^{m_\Omega} \int_{\partial\Omega_i} \sum_{j=1}^{\alpha} \left(B_j^{\mathbf{v},\partial\Omega_i}(\mathbf{u}_h) v_{h,j} + \sum_{k=1}^{d} B_{j,k}^{\nabla\mathbf{v},\partial\Omega_i}(\mathbf{u}_h) \frac{\partial v_{h,j}}{\partial x_k} \right) \mathrm{d}s \end{aligned} \tag{5.4.10}$$

for all $\mathbf{v}_h \in \hat{W}(\Omega)$, where m_Ω is the total number of boundary components.

5.4.4 Calculating the Gâteaux Derivative

We introduce the tensors:

$$\begin{aligned}
\mathfrak{E}_{j,l}^{\mathbf{v},\mathbf{u}} &= \frac{\partial E_j^{\mathbf{v}}}{\partial u_{h,l}}, & \mathfrak{E}_{j,l,k}^{\mathbf{v},\nabla\mathbf{u}} &= \frac{\partial E_j^{\mathbf{v}}}{\partial \frac{\partial u_{h,l}}{\partial x_k}}, & \mathfrak{E}_{j,k,l}^{\nabla\mathbf{v},\mathbf{u}} &= \frac{\partial E_{j,k}^{\nabla\mathbf{v}}}{\partial u_{h,l}}, & \mathfrak{E}_{j,k_v,l,k_u}^{\nabla\mathbf{v},\nabla\mathbf{u}} &= \frac{\partial E_{j,k_v}^{\nabla\mathbf{v}}}{\partial \frac{\partial u_{h,l}}{\partial x_{k_u}}}, \\
\mathfrak{B}_{j,l}^{\mathbf{v},\mathbf{u}} &= \frac{\partial B_j^{\mathbf{v}}}{\partial u_{h,l}}, & \mathfrak{B}_{j,l,k}^{\mathbf{v},\nabla\mathbf{u}} &= \frac{\partial B_j^{\mathbf{v}}}{\partial \frac{\partial u_{h,l}}{\partial x_k}}, & \mathfrak{B}_{j,k,l}^{\nabla\mathbf{v},\mathbf{u}} &= \frac{\partial B_{j,k}^{\nabla\mathbf{v}}}{\partial u_{h,l}}, & \mathfrak{B}_{j,k_v,l,k_u}^{\nabla\mathbf{v},\nabla\mathbf{u}} &= \frac{\partial B_{j,k_v}^{\nabla\mathbf{v}}}{\partial \frac{\partial u_{h,l}}{\partial x_{k_u}}},
\end{aligned} \tag{5.4.11}$$

```
V = FemFunctionSpace(mesh)
u, v = V.trial(), V.test()
residual = dot(grad(u), grad(v))*dx - f*v*dx
gd = GateauxDerivative(residual, V)
```

Figure 5.20: AptoPy python code computing the Gâteaux derivative of the finite element residual formulation of the Poisson equation.

where $j, l = 1, \ldots, \alpha$ and $k, k_u, k_v = 1, \ldots, d$. These tensors can each be calculated using sympy, the Gâteaux derivative of the residual expression in (5.4.10) is given by,

$$\begin{aligned}
\mathcal{N}'\left[\mathbf{w}_h\right](\mathbf{u}_h;\mathbf{v}_h) = &\int_\Omega \sum_{j=1}^{\alpha}\sum_{l=1}^{\alpha}\left(\mathfrak{E}^{\mathbf{v},\mathbf{u}}_{j,l}(\mathbf{u}_h)w_{h,l}v_{h,j} + \sum_{k=1}^{d}\mathfrak{E}^{\mathbf{v},\nabla\mathbf{u}}_{j,l,k}(\mathbf{u}_h)\frac{\partial w_{h,l}}{\partial x_k}v_{h,j}\right)\,\mathrm{d}\mathbf{x}\\
&+\int_\Omega \sum_{j=1}^{\alpha}\sum_{l=1}^{\alpha}\sum_{k_v=1}^{d}\left(\mathfrak{E}^{\nabla\mathbf{v},\mathbf{u}}_{j,k_v,l}(\mathbf{u}_h)w_{h,l}\frac{\partial v_{h,j}}{\partial x_{k_v}} + \sum_{k_u=1}^{d}\mathfrak{E}^{\nabla\mathbf{v},\nabla\mathbf{u}}_{j,k_v,l,k_u}(\mathbf{u}_h)\frac{\partial w_{h,l}}{\partial x_{k_u}}\frac{\partial v_{h,j}}{\partial x_{k_v}}\right)\,\mathrm{d}\mathbf{x}\\
&+\sum_{i=1}^{m_\Omega}\int_{\partial\Omega_i}\sum_{j=1}^{\alpha}\sum_{l=1}^{\alpha}\left(\mathfrak{B}^{\mathbf{v},\mathbf{u},\partial\Omega_i}_{j,l}(\mathbf{u}_h)w_{h,l}v_{h,j} + \sum_{k=1}^{d}\mathfrak{B}^{\mathbf{v},\nabla\mathbf{u},\partial\Omega_i}_{j,l,k}(\mathbf{u}_h)\frac{\partial w_{h,l}}{\partial x_k}v_{h,j}\right)\,\mathrm{d}s\\
&+\sum_{i=1}^{m_\Omega}\int_{\partial\Omega_i}\sum_{j=1}^{\alpha}\sum_{l=1}^{\alpha}\sum_{k_v=1}^{d}\left(\mathfrak{B}^{\nabla\mathbf{v},\mathbf{u},\partial\Omega_i}_{j,k_v,l}(\mathbf{u}_h)w_{h,l}\frac{\partial v_{h,j}}{\partial x_{k_v}} + \sum_{k_u=1}^{d}\mathfrak{B}^{\nabla\mathbf{v},\nabla\mathbf{u},\partial\Omega_i}_{j,k_v,l,k_u}(\mathbf{u}_h)\frac{\partial w_{h,l}}{\partial x_{k_u}}\frac{\partial v_{h,j}}{\partial x_{k_v}}\right)\,\mathrm{d}s.
\end{aligned} \tag{5.4.12}$$

Although seemingly verbose, this formulation reduces the computational complexity of calculating the symbolic Gâteaux derivative of the residual form by breaking up the residual equation into smaller segments. Once translated to Fortran code, optimised linear algebra libraries can also take advantage of the matrix product calculations between the arrays storing the residual and Gâteaux derivative tensors and the arrays storing the basis of the test functions and their derivatives. AptoPy implements the calculation and storage of the symbolic Gâteaux derivative in the `GateauxDerivative` class. An example of the automatic computation of the Gâteaux derivative residual for the finite element formulation of the Poisson equation is demonstrated in Figure 5.20.

We will show later in Section 6.6 that computing the Gâteaux derivative of a finite element formulation is the most costly computation performed by AptoPy. To alleviate this issue, each computed Gâteaux derivative can be serialised and cached to the persistent memory of the hard disk. These cached Gâteaux derivatives are indexed by meta-data generated from the residual from which they were computed, such that AptoPy will load the corresponding formulation when required on subsequent code generation.

5.4.5 Applying Newton's Method: The Residual Vector and Jacobi Matrix

We write $N_j(h) = \dim V^j_{h,\ell}(\Omega)$, $j = 1, \ldots, \alpha$, to denote the dimension of the discrete solution space $V^j_{h,\ell}(\Omega)$, and let $V^j_{h,\ell}(\Omega) = \text{span}\left\{\phi^j_1, \ldots, \phi^j_{N_j(h)}\right\}$ for linearly independent basis functions ϕ^j_n, $n = 1, \ldots, N_j(h)$. Thereby, we may write $u_{h,j}$ in the following form

$$u_{h,j} = \sum_{m=1}^{N_j(h)} U^j_m \phi^j_m, \quad j = 1, \ldots, \alpha, \tag{5.4.13}$$

for solution vector U^j and denote the complete solution vector by

$$\mathfrak{U} = \left(U^1, \ldots, U^\alpha\right). \tag{5.4.14}$$

This solution vector is represented in AptoPy by the `SolutionVector` class whose constructor arguments require the function space of the problem. For example, given the function space of a finite element problem $V^1_{h,\ell}(\Omega) \times V^2_{h,\ell}(\Omega)$, an appropriate solution vector $\mathfrak{U}$ is constructed by calling `U = SolutionVector(V1 * V2)`. The residual formulation of equation (5.4.10) is then constructed in the vector

$$\mathfrak{R}(\mathfrak{U}) := \mathcal{N}\left(\left(\sum_{m=1}^{N_1(h)} U^1_m \phi^1_m, \ldots, \sum_{m=1}^{N_\alpha(h)} U^\alpha_m \phi^\alpha_m\right); \phi^j_n\right), \tag{5.4.15}$$

for $j = 1, \ldots, \alpha$ and $n = 1, \ldots, N_j(h)$. The Gâteaux derivative is constructed in the matrix

$$\mathfrak{R}'(\mathfrak{U}) := \mathcal{N}'\left[\mathbf{d}_h\right]\left(\left(\sum_{m=1}^{N_1(h)} U^1_m \phi^1_m, \ldots, \sum_{m=1}^{N_\alpha(h)} U^\alpha_m \phi^\alpha_m\right); \phi^j_n\right) \tag{5.4.16}$$

for $j = 1, \ldots, \alpha$ and $n = 1, \ldots, N_j(h)$.

The damped iterative Newton method then employs the vector in (5.4.15) and the matrix in (5.4.16) for each iterative solution vector approximation $\mathfrak{U}_n$ of the Newton method, i.e.,

$$\mathfrak{U}_{n+1} = \mathfrak{U}_n - \vartheta\left[\mathfrak{R}'\left(\mathfrak{U}_n\right)\right]^{-1}\mathfrak{R}\left(\mathfrak{U}_n\right) \tag{5.4.17}$$

with initial starting guess $\mathfrak{U}_0 = \left(U^1_0, \ldots, U^\alpha_0\right)$.

The data-structures and subroutines required to set up and store the vectors $\mathfrak{U}$ and $\mathfrak{R}$ and the sparse matrix $\mathfrak{R}'$ are provided by AptoFEM, along with subroutines for evaluating the iterative method in (5.4.17). The AptoPy symbolic representation for computing a solution to a finite element residual formulation in `SolutionVector U` is simply to call `newton_solve(residual, U)`. In this case the Gâteaux derivative of the residual is computed implicitly based on the properties of the `SolutionVector`; a custom `GateauxDerivative` argument can be passed as an optional argument if required.

5.4.6 Numerical Quadrature in AptoFEM

In order to evaluate the integration operations required in the finite element formulation, a quadrature method is employed. On each element κ of the mesh $\mathcal{T}_\Omega^h$, the quadrature points $\mathbf{x}_{q_k}$ and quadrature weights w_{q_k}, $q_k = 1, \ldots, N(q_k)$, where $N(q_k)$ indicates the total number of quadrature points, are provided by AptoFEM. This allows for the integral operations over the domain Ω to be expressed as a sum of integrals over every element

$$\int_\Omega f(\mathbf{x}) \, \mathrm{d}\mathbf{x} \approx \sum_{\kappa \in \mathcal{T}_\Omega^h} \sum_{q_k=1}^{N(q_k)} w_{q_k} f(\mathbf{x}_{q_k}). \tag{5.4.18}$$

AptoFEM also calculates the basis functions and their derivatives of each finite element function space at each quadrature point in the arrays

$$\begin{aligned} \phi_n^j(\mathbf{x}_{q_k}) &= \texttt{phi(j, qk, n)}, \\ \frac{\partial \phi_n^j(\mathbf{x}_{q_k})}{\partial x_k} &= \texttt{grad\_phi(j, qk, k, n)}. \end{aligned} \tag{5.4.19}$$

Furthermore, AptoFEM calculates the finite element solution and its derivatives at a given quadrature point $\mathbf{x}_{q_k}$, $q_k = 1, \ldots, N(q_k)$, on an element $\kappa \in \mathcal{T}_\Omega^h$. These values are then stored in the arrays

$$\begin{aligned} u_{h,j}(\mathbf{x}_{q_k}) &= \texttt{var(j, qk)}, \\ \frac{\partial u_{h,j}(\mathbf{x}_{q_k})}{\partial x_k} &= \texttt{dvdr(j, k, qk)}. \end{aligned} \tag{5.4.20}$$

AptoPy thereby ensures that the numerical integration of the trial and test functions from `V.trial()` and `V.test()`, respectively, align with the function space and quadrature point indexing of AptoFEM.

5.4.7 Residual Vector and Jacobi Matrix Construction in AptoFEM

AptoFEM assembles the residual vector and Jacobi matrix by iterating over all elements in the mesh and adding the local contributions of each element to the global system. Once AptoPy has constructed each of the tensors $E^{\mathbf{v}}(\mathbf{u})$, $E^{\nabla\mathbf{v}}(\mathbf{u})$, $B^{\mathbf{v}}(\mathbf{u})$ and $B^{\nabla\mathbf{v}}(\mathbf{u})$ and computed their Gâteaux derivatives in the tensors defined in (5.4.11), they are each translated to arrays in AptoFEM of the appropriate dimension at each quadrature point of a single element. Thereby AptoFEM computes the each element contribution in the system assembly. For example, the element residual contribution arrays

$$\begin{aligned} E_j^{\mathbf{v}} &= \texttt{e(j)}, \\ E_{j,k}^{\nabla\mathbf{v}} &= \texttt{er(j, k)}, \end{aligned} \tag{5.4.21}$$

and the element Jacobi matrix contribution arrays

$$\mathfrak{E}^{\mathbf{v},\mathbf{u}}_{j,l} = \texttt{dedv(j, l)}, \qquad \mathfrak{E}^{\mathbf{v},\nabla\mathbf{u}}_{j,l,k} = \texttt{dedvr(j, l, k)},$$
$$\mathfrak{E}^{\nabla\mathbf{v},\mathbf{u}}_{j,k,l} = \texttt{derdv(j, k, l)}, \qquad \mathfrak{E}^{\nabla\mathbf{v},\nabla\mathbf{u}}_{j,k_v,l,k_u} = \texttt{derdvr(j, k\_v, l, k\_u)}. \tag{5.4.22}$$

An example of the automatically generated Fortran code to form these arrays at a single quadrature point for the finite element formulation of the Poisson equation is shown in Figure 5.21.

The vector $\mathfrak{R}$ and matrix $\mathfrak{R}'$ can then be constructed by writing the integral over the domain Ω in (5.4.15) and (5.4.16) as a sum of numerical quadrature integral calculations over all elements $\kappa \in \mathcal{T}^h_\Omega$. An example of the automatically generated Fortran code to construct these matrices is shown in Figures 5.22 and 5.23, respectively.

```
f = 1
residual = dot(grad(u), grad(v))*dx - f*v*dx
```

```
subroutine residual_vector(var, dvdr, svars, e, er, ...)
    ...
    e(1) = -1.0d0
    er(1,1) = dvdr(1,1)
    er(1,2) = dvdr(1,2)
end subroutine residual_vector
subroutine jacobi_matrix(var, dvdr, svars, dedv, derdv, dedvr, derdvr, ...)
    ...
    derdvr(1,1,1,1) = 1.0d0
    derdvr(1,2,1,2) = 1.0d0
end subroutine jacobi_matrix
```

Figure 5.21: AptoPy python code representing the finite element residual of the Poisson equation $-\nabla^2 u = 1$, and the resulting automatically generated Fortran subroutines forming the residual vector and Jacobi matrix arrays. These functions are called for finite element solution interpolation `var(:)`, its derivatives `dvdr(:,:)`, spatial variables `svars(:)` and residual/Jacobi tensors at a given quadrature point `qk`.

```
! Element Residual Vector
do j = 1,no_pdes
    do n = 1,no_dofs_per_variable(j)
        do qk = 1,no_quad_points

            element_residual(j,n) = element_residual(j,n) &
                - integral_weighting(qk)*( &
                e(j,qk)*phi(j,qk,n) &
                + dot_product(er(j,:,qk), grad_phi(j,qk,:,n)))

        end do
    end do
end do
```

Figure 5.22: Fortran code generated for AptoFEM by AptoPy to calculate contributions to the element residual vector $\mathfrak{R}$ given the residual tensors `e(:,qk)` and `er(:,:,qk)` at each quadrature point provided by AptoFEM `qk`.

```
! Element Jacobi Matrix
do j = 1,no_pdes
    do l = 1,no_pdes
        do qk = 1,no_quad_points
            do n = 1,no_dofs_per_variable(j)
                do m = 1,no_dofs_per_variable(l)
                    element_matrix(j,l,n,m) = element_matrix(j,l,n,m) &
                        + integral_weighting(qk)*( &
                        dedv(j,l,qk)*phi(j,qk,n)*phi(l,qk,m) &
                        + dot_product(derdv(j,1:problem_dim,l,qk), &
                        grad_phi(j,qk,:,n))*phi(l,qk,m) &
                        + dot_product(dedvr(j,l,1:problem_dim,qk), &
                        grad_phi(l,qk,:,m))*phi(j,qk,n) &
                        + dot_product(grad_phi(j,qk,:,n), &
                        matmul(derdvr(j,1:problem_dim,l,1:problem_dim,qk), &
                        grad_phi(l,qk,:,m))))
                end do
            end do
        end do
    end do
end do
```

Figure 5.23: Fortran code generated for AptoFEM by AptoPy to calculate contributions to the element Jacobi matrix $\mathfrak{R}'$ given the Jacobian tensors `dedv(:,:,qk)`, `dedvr(:,:,:,qk)`, `derdv(:,:,:,qk)` and `derdvr(:,:,:,:,qk)` at each quadrature point provided by AptoFEM `qk`.

CHAPTER 6

AptoPy Validation and Performance

Validation of mathematical code is essential to ensure its correctness. To this end, the underlying mathematical operators of AptoPy are verified by the unit test framework of the sympy library. As an extension to these tests, AptoPy employs its own unit test suite verifying results of its computations of DG FEM operators, vector calculus operations, symbolic mesh generation and finite element space construction. A further suite of regression tests validate the Fortran code generated for AptoFEM by testing *a priori* error convergence rates.

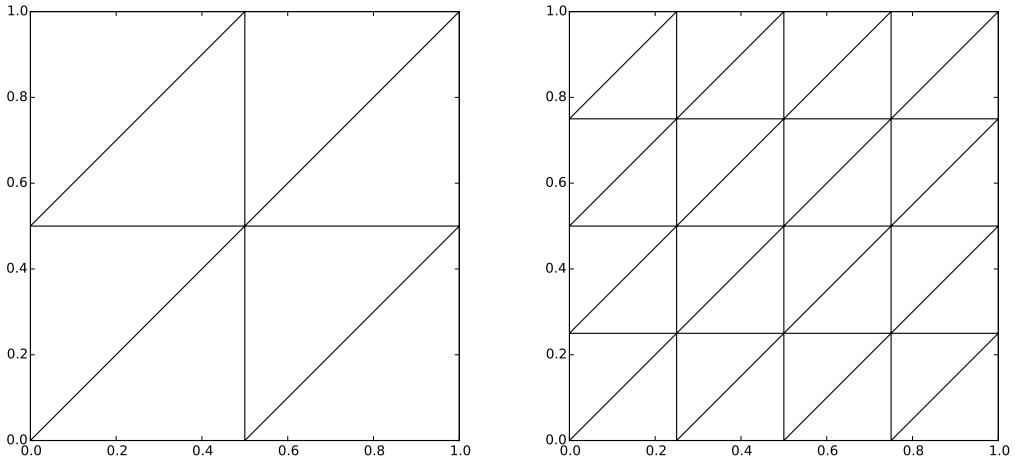
In this chapter we demonstrate four of these regression tests featuring boundary value problems relevant to the MPA-CVD model. We test error convergence rates of the finite element solution calculated with the code generated by AptoPy, compared against a known analytical solution. For each of these tests we adopt a cylindrical azimuthally symmetric coordinate system. We consider the convection-diffusion-reaction equation in Section 6.2, the incompressible Navier-Stokes equations in Section 6.3, the homogeneous cylindrical waveguide in Section 6.4 and the homogeneous empty cylindrical microwave resonator cavity in Section 6.5.

6.1 Convergence Rates

In each of the following sections, we test for optimal error convergence rates of the DG symmetric interior penalty method. For an analytical $\mathbf{u} \in \left[H^{\ell+1}(\Omega)\right]^m$ which is approximated by the DG finite element solution $\mathbf{u}_h \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^h)$, we expect the error to converge in the L_2 and H^1 norms according to

$$\|\mathbf{u} - \mathbf{u}_h\|_{L_2(\Omega)} \leq C_h^{L_2} h^{\ell+1} \left|\mathbf{u}\right|_{H^{\ell+1}(\Omega)}, \tag{6.1.1}$$

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)} \leq C_h^{H^1} h^{\ell} \left|\mathbf{u}\right|_{H^{\ell+1}(\Omega)}, \tag{6.1.2}$$



(a) $\mathcal{T}_\Omega^{h_\mathrm{c}}$, $h_\mathrm{c}^2 = \dfrac{1}{2}$ **(b)** $\mathcal{T}_\Omega^{h_\mathrm{f}}$, $h_\mathrm{f}^2 = \dfrac{1}{8}$

Figure 6.1: Examples of a coarse mesh and a successively finer mesh employed by a numerical error convergence rate test.

respectively, for positive constants $C_h^{L_2}$ and $C_h^{H^1}$, and element diameter h. In essence, the error in the L_2 norm should converge at the optimal rate $\mathcal{O}(h^{\ell+1})$ and in the H^1 norm at the optimal rate $\mathcal{O}(h^\ell)$.

Consider a domain Ω which is subdivided into two separate structured shape regular meshes, one coarse $\mathcal{T}_\Omega^{h_\mathrm{c}}$ and one fine $\mathcal{T}_\Omega^{h_\mathrm{f}}$, such that $h_\kappa = h_\mathrm{c}$ for all $\kappa \in \mathcal{T}_\Omega^{h_\mathrm{c}}$, $h_\kappa = h_\mathrm{f}$ for all $\kappa \in \mathcal{T}_\Omega^{h_\mathrm{f}}$ and $h_\mathrm{c} > h_\mathrm{f}$. An example of a coarse and fine mesh for the domain $\Omega = (0,1) \times (0,1)$ is shown in Figure 6.1. Let $\mathbf{u}_{h_\mathrm{c}} \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^{h_\mathrm{c}})$ be the finite element solution computed on the coarse mesh and $\mathbf{u}_{h_\mathrm{f}} \in \mathbf{V}_\ell^m(\mathcal{T}_\Omega^{h_\mathrm{f}})$ be the finite element solution computed on the fine mesh. The relative rate at which the finite element solution converges to the analytical solution between these successive meshes can be calculated. To this end, we have convergence rates

$$\mathrm{R}_{L_2} := \frac{\log\left(\|\mathbf{u} - \mathbf{u}_{h_\mathrm{c}}\|_{L_2(\Omega)} / \|\mathbf{u} - \mathbf{u}_{h_\mathrm{f}}\|_{L_2(\Omega)}\right)}{\log\left(h_\mathrm{c}/h_\mathrm{f}\right)} \sim \ell + 1 \tag{6.1.3}$$

and

$$\mathrm{R}_{H^1} := \frac{\log\left(\|\mathbf{u} - \mathbf{u}_{h_\mathrm{c}}\|_{H^1(\Omega)} / \|\mathbf{u} - \mathbf{u}_{h_\mathrm{f}}\|_{H^1(\Omega)}\right)}{\log\left(h_\mathrm{c}/h_\mathrm{f}\right)} \sim \ell. \tag{6.1.4}$$

For the case of the optimal convergence rates of the electromagnetic field solution computed from the DG FEM approximation we refer to [80]. As such, we further seek optimal convergence rates of the electromagnetic field approximation in the $H(\mathrm{curl};\Omega)$ norm such that

$$\mathrm{R}_{H(\mathrm{curl})} := \frac{\log\left(\|\mathbf{E} - \mathbf{E}_{h_\mathrm{c}}\|_{H(\mathrm{curl};\Omega)} / \|\mathbf{E} - \mathbf{E}_{h_\mathrm{f}}\|_{H(\mathrm{curl};\Omega)}\right)}{\log\left(h_\mathrm{c}/h_\mathrm{f}\right)} \sim \ell, \tag{6.1.5}$$

where $\|\mathbf{E}\|^2_{H(\text{curl};\Omega)} = \|\mathbf{E}\|^2_{L_2(\Omega)} + \|\nabla \times \mathbf{E}\|^2_{L_2(\Omega)}$.

Regarding optimal convergence rates of the eigenvalue computations from the DG FEM symmetric interior penalty method applied to the Maxwell operator, we refer to Buffa et al. [25, Theorem 4.3]. We expect the absolute error of the approximate eigenvalue λ_h to converge to the true value λ at a rate of $\mathcal{O}(h^{2\ell})$, i.e., we compute

$$\mathrm{R}_\lambda := \frac{\log\left(|\lambda - \lambda_{h_\mathrm{c}}| / |\lambda - \lambda_{h_\mathrm{f}}|\right)}{\log\left(h_\mathrm{c}/h_\mathrm{f}\right)} \sim 2\ell. \tag{6.1.6}$$

6.2 Convection-Diffusion-Reaction

Adopting a cylindrical coordinate system $\mathbf{x} = (r, \theta, z)$, let $\Omega = (0,1) \times \{0\} \times (0,1)$ with boundary $\partial\Omega$. We let $\mathcal{T}^h_\Omega$ be the subdivision of Ω into a structured shape regular mesh of two dimensional simplices taking advantage of azimuthal symmetry. We seek the solution u to the convection-diffusion-reaction equation with diffusion coefficient A, convection vector coefficient $\mathbf{b}$ and reaction coefficient c, namely,

$$-\nabla \cdot (A \nabla u) + \nabla \cdot (\mathbf{b} u) + cu = f \qquad \text{in } \Omega, \tag{6.2.1}$$

$$u = 0 \qquad \text{on } \partial\Omega, \tag{6.2.2}$$

where $c(\mathbf{x}) - \frac{1}{2}\nabla \cdot \mathbf{b}(\mathbf{x}) \geq 0,\ \mathbf{x} \in \overline{\Omega}$. Constructing a test case, we select

$$u = r^2 \sin(\pi r) \sin(\pi z) \tag{6.2.3}$$

and formulate the source term f for coefficients

$$A = (r+1)^2 + (z+1)^2, \tag{6.2.4}$$

$$\mathbf{b} = \left(\sin(\pi r)^2,\ 0,\ \cos(\pi z)^2\right)^\top, \tag{6.2.5}$$

$$c = r^2 \sin(\pi r) \sin(\pi z) + 2. \tag{6.2.6}$$

It is clear that equation (6.2.1) can be rewritten in terms of the convective and viscous operators

$$\nabla \cdot (\mathcal{F}^c(\mathbf{u}) - \mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})) + cu = f \qquad \text{in } \Omega, \tag{6.2.7}$$

$$u = 0 \qquad \text{on } \partial\Omega, \tag{6.2.8}$$

where

$$\mathcal{F}^c(\mathbf{u}) = \mathbf{b}u, \quad \mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) = A\nabla u. \tag{6.2.9}$$

The DG FEM formulation is given by: find $u_h \in V_\ell(\mathcal{T}^h_\Omega)$ such that

$$\mathcal{N}^c_\Omega(u_h; v_h) + \mathcal{N}^v_\Omega(u_h; v_h) + \int_\Omega (c u_h v_h - f v_h)\ \mathrm{d}\mathbf{x} = 0 \ \ \forall v_h \in V_\ell(\mathcal{T}^h_\Omega). \tag{6.2.10}$$

Using this formulation and the concepts outlined in Section 5.3 for automatic computation of DG FEM formulations, the AptoPy code to construct the DG finite element formulation of this problem is presented in Figure 6.2. We refer to Section 5.1.3 regarding handling the symbolic form of the mesh, its spatial variables and the boundary and volume integration elements. The convergence rates of the finite element approximation to the solution u are listed in Table 6.1, showing clear agreement with the convergence rates predicted by DG finite element approximation theory stated in (6.1.3) and (6.1.4).

```
# Boundary condition and function space
V = FemFunctionSpace(mesh, poly_order=1, element_type='DG')
u, v = V.trial(), V.test()
bcs = [DirichletBC(dS, 0.0)]

# Elliptic Term
A = AptoFunction((r + 1)**2 + (z + 1)**2, (r, z), 'A_coeff')
def F_v(u):
    return A*grad(u)
def F_v_v(u, grad_u):
    return A*grad_u
eo = EllipticOperator(mesh, V, bcs, F_v=F_v, F_v_v=F_v_v)

# Hyperbolic Term
b = AptoFunction(Matrix([sin(pi*r)**2, 0, cos(pi*z)**2]), (r, z), 'b_coeff')
def F_c(u):
    return b*u
ho = HyperbolicOperator(mesh, V, bcs, F_c=F_c, alpha=Abs(dot(b, n)))

# Reaction and source terms
u_soln = r**2*sin(pi*r)*sin(pi*z)
c = AptoFunction(u_soln + 2, (r, z), 'c_coeff')
f = div(F_c(u_soln) - F_v(u_soln)) + c*u_soln

res = eo.generate_fem_formulation() + ho.generate_fem_formulation() \
        + c*u*v*dx - f*v*dx

# Cylindrical coordinate system integration element
res *= r
```

Figure 6.2: AptoPy automatic computation of the DGFEM formulation of the convection-diffusion-reaction equation in (6.2.1).

$\ell = 1$					$\ell = 2$				
h^2	$\|u - u_h\|_{L_2(\Omega)}$	R_{L_2}	$\|u - u_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|u - u_h\|_{L_2(\Omega)}$	R_{L_2}	$\|u - u_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	$6.28400 \cdot 10^{-2}$	0.00000	0.63962	0.00000	0.50000	$1.84710 \cdot 10^{-2}$	0.00000	0.26422	0.00000
0.12500	$2.37870 \cdot 10^{-2}$	1.40151	0.38138	0.74599	0.12500	$2.40520 \cdot 10^{-3}$	2.94103	$7.57800 \cdot 10^{-2}$	1.80185
$3.12500 \cdot 10^{-2}$	$7.13700 \cdot 10^{-3}$	1.73678	0.20104	0.92375	$3.12500 \cdot 10^{-2}$	$2.99920 \cdot 10^{-4}$	3.00351	$1.98130 \cdot 10^{-2}$	1.93537
$7.81250 \cdot 10^{-3}$	$1.91640 \cdot 10^{-3}$	1.89692	0.10212	0.97722	$7.81250 \cdot 10^{-3}$	$3.76450 \cdot 10^{-5}$	2.99405	$5.02790 \cdot 10^{-3}$	1.97842
$1.95313 \cdot 10^{-3}$	$4.92770 \cdot 10^{-4}$	1.95941	$5.13160 \cdot 10^{-2}$	0.99278	$1.95313 \cdot 10^{-3}$	$4.72270 \cdot 10^{-6}$	2.99477	$1.26350 \cdot 10^{-3}$	1.99253
$4.88281 \cdot 10^{-4}$	$1.24670 \cdot 10^{-4}$	1.98280	$2.57050 \cdot 10^{-2}$	0.99736	$4.88281 \cdot 10^{-4}$	$5.91590 \cdot 10^{-7}$	2.99694	$3.16470 \cdot 10^{-4}$	1.99729
$\ell = 3$					$\ell = 4$				
h^2	$\|u - u_h\|_{L_2(\Omega)}$	R_{L_2}	$\|u - u_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|u - u_h\|_{L_2(\Omega)}$	R_{L_2}	$\|u - u_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	$3.25760 \cdot 10^{-3}$	0.00000	$6.48470 \cdot 10^{-2}$	0.00000	0.50000	$5.56980 \cdot 10^{-4}$	0.00000	$1.34910 \cdot 10^{-2}$	0.00000
0.12500	$2.29490 \cdot 10^{-4}$	3.82731	$9.24190 \cdot 10^{-3}$	2.81078	0.12500	$1.98800 \cdot 10^{-5}$	4.80824	$9.57610 \cdot 10^{-4}$	3.81642
$3.12500 \cdot 10^{-2}$	$1.42210 \cdot 10^{-5}$	4.01234	$1.18030 \cdot 10^{-3}$	2.96904	$3.12500 \cdot 10^{-2}$	$6.47630 \cdot 10^{-7}$	4.94000	$6.17030 \cdot 10^{-5}$	3.95603
$7.81250 \cdot 10^{-3}$	$8.72460 \cdot 10^{-7}$	4.02679	$1.47540 \cdot 10^{-4}$	2.99998	$7.81250 \cdot 10^{-3}$	$2.05220 \cdot 10^{-8}$	4.97993	$3.88480 \cdot 10^{-6}$	3.98943
$1.95313 \cdot 10^{-3}$	$5.39390 \cdot 10^{-8}$	4.01569	$1.83990 \cdot 10^{-5}$	3.00341	$1.95313 \cdot 10^{-3}$	$6.43970 \cdot 10^{-10}$	4.99403	$2.43130 \cdot 10^{-7}$	3.99804
$4.88281 \cdot 10^{-4}$	$3.35280 \cdot 10^{-9}$	4.00789	$2.29590 \cdot 10^{-6}$	3.00250	$4.88281 \cdot 10^{-4}$	$2.01490 \cdot 10^{-11}$	4.99821	$1.51970 \cdot 10^{-8}$	3.99987

Table 6.1: Numerical convergence rate computations using AptoPy generated code appled to the convection-diffusion-reaction equation (6.2.1).

6.3 Navier-Stokes Equations

In this section we validate the Fortran code generated by AptoPy when applied to the incompressible Navier-Stokes equations in cylindrical coordinates. Let $\Omega = (1,2) \times \{0\} \times (0,1)$ with boundary $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$, where $\partial\Omega_N = \{2\} \times \{0\} \times [0,1]$ and $\partial\Omega_D = \partial\Omega \setminus \partial\Omega_N$. As in Section 6.2 let $\mathcal{T}_\Omega^h$ be the subdivision of Ω into a structured shape regular mesh of two dimensional simplices exploiting azimuthal symmetry. We seek the solutions to the velocity field $\mathbf{u}$ and pressure field p of the Navier-Stokes equations for a given density ρ and viscosity η, i.e.,

$$\nabla \cdot (\mathcal{F}^c(\mathbf{u}) - \mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u})) = \mathbf{f} \qquad \text{in } \Omega, \tag{6.3.1}$$

$$\nabla \cdot (\rho\mathbf{u}) = 0 \qquad \text{in } \Omega, \tag{6.3.2}$$

$$\mathbf{u} = \mathbf{g}_D \qquad \text{on } \partial\Omega_D, \tag{6.3.3}$$

$$\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) \cdot \mathbf{n} = \mathbf{g}_N \qquad \text{on } \partial\Omega_N, \tag{6.3.4}$$

where

$$\mathcal{F}^c(\mathbf{u}) = \rho\mathbf{u} \otimes \mathbf{u}, \tag{6.3.5}$$

$$\mathcal{F}^v(\mathbf{u}; \nabla\mathbf{u}) = \eta \left(\nabla\mathbf{u} + \nabla\mathbf{u}^\top - \frac{2}{3} (\nabla \cdot \mathbf{u}) \underline{\mathbf{I}} \right) - p\underline{\mathbf{I}}. \tag{6.3.6}$$

We select $\mathbf{f}$ such that the solution for $\rho = 1$ and $\mu = 1$ is given by

$$\mathbf{u} = \begin{pmatrix} \frac{1}{r}\sin(\pi r)\sin(\pi z) \\ \sin(\pi r)\sin(\pi z) \\ \frac{1}{r}\cos(\pi r)\cos(\pi z) \end{pmatrix}, \quad p = e^{r+z}; \tag{6.3.7}$$

in this case we prescribe the boundary conditions

$$\mathbf{g}_D = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{r}\cos(\pi r)\cos(\pi z) \end{pmatrix}, \tag{6.3.8}$$

$$\mathbf{g}_N = \left(\nabla\mathbf{g}_D + \nabla\mathbf{g}_D^\top - \frac{2}{3} (\nabla \cdot \mathbf{g}_D) \underline{\mathbf{I}} - e^{r+z}\underline{\mathbf{I}} \right) \cdot \mathbf{n}. \tag{6.3.9}$$

The DG FEM formulation is to find $(\mathbf{u}_h, p_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell-1}(\mathcal{T}_\Omega^h)$ such that

$$\mathcal{N}_\Omega^c(\mathbf{u}_h; \mathbf{v}_h) + \mathcal{N}_\Omega^v(\mathbf{u}_h; \mathbf{v}_h) + \mathcal{N}_\Omega^{\text{cont}}(\mathbf{u}_h; q_h) = 0, \tag{6.3.10}$$

for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell-1}(\mathcal{T}_\Omega^h)$.

The AptoPy code used to generate the DG FEM formulation of this problem is presented in Figure 6.3. The convergence rates of the velocity and pressure field variables are presented in Tables 6.2 and 6.3, respectively. The computed convergence rates agree well with equations (6.1.3) and (6.1.4). We also note that the piecewise constant representation ($\ell - 1 = 0$) of the DG approximation to the pressure does not converge to the analytical solution in the H^1 norm. This is expected, as the gradient of p_h is zero everywhere in this case.

```
# Function spaces
Q = FemFunctionSpace(mesh, poly_order=1, element_type='DG')
p = Q.trial()
V = FemVectorFunctionSpace(mesh, poly_order=2, element_type='DG')

# Manufactured solution
gD = Matrix([1/r*sin(pi*r)*sin(pi*z),
             sin(pi*r)*sin(pi*z),
             1/r*cos(pi*r)*cos(pi*z)])
pD = exp(r+z)
gN = dot(grad(gD) + grad(gD).T - 2.0/3.0*div(gD)*eye(3), n) - pD*n

# Boundary conditions
bcs = [DirichletBC(dSD, gD), NeumannBC(dSN, gN)]

# Source term construction
def F_c(u):
    return u*u.T
def F_v(u):
    return grad(u) + grad(u).T - 2.0/3.0*(div(u))*eye(3) - p*eye(3)
f = div(F_c(gD) - F_v(gD)) + grad(pD)

# DG FEM residual construction
pe = NavierStokesEquationsIncompressible(mesh, V, Q, bcs)
residual = pe.generate_fem_formulation() - pe.generate_forcing_formulation(f)

# Cylindrical coordinate system integration element
residual *= r
```

Figure 6.3: AptoPy automatic computation of the DG FEM formulation of the Navier-Stokes equations in equation (6.3.1).

$\ell = 1$					$\ell = 2$				
h^2	$\|\mathbf{u} - \mathbf{u}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|\mathbf{u} - \mathbf{u}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	0.22030	0.00000	2.13240	0.00000	0.50000	$7.45090 \cdot 10^{-2}$	0.00000	0.81353	0.00000
0.12500	$8.70330 \cdot 10^{-2}$	1.33984	1.22040	0.80512	0.12500	$8.44200 \cdot 10^{-3}$	3.14176	0.21439	1.92396
$3.12500 \cdot 10^{-2}$	$2.95520 \cdot 10^{-2}$	1.55831	0.62487	0.96573	$3.12500 \cdot 10^{-2}$	$8.88430 \cdot 10^{-4}$	3.24825	$5.20640 \cdot 10^{-2}$	2.04188
$7.81250 \cdot 10^{-3}$	$8.64210 \cdot 10^{-3}$	1.77380	0.30898	1.01604	$7.81250 \cdot 10^{-3}$	$1.00150 \cdot 10^{-4}$	3.14910	$1.26550 \cdot 10^{-2}$	2.04058
$1.95313 \cdot 10^{-3}$	$2.28330 \cdot 10^{-3}$	1.92026	0.15318	1.01229	$1.95313 \cdot 10^{-3}$	$1.20460 \cdot 10^{-5}$	3.05554	$3.12550 \cdot 10^{-3}$	2.01755
$4.88281 \cdot 10^{-4}$	$5.79430 \cdot 10^{-4}$	1.97841	$7.63750 \cdot 10^{-2}$	1.00406	$4.88281 \cdot 10^{-4}$	$1.48830 \cdot 10^{-6}$	3.01682	$7.77910 \cdot 10^{-4}$	2.00641
$\ell = 3$					$\ell = 4$				
h^2	$\|\mathbf{u} - \mathbf{u}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|\mathbf{u} - \mathbf{u}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	$1.10320 \cdot 10^{-2}$	0.00000	0.18067	0.00000	0.50000	$1.75580 \cdot 10^{-3}$	0.00000	$3.30270 \cdot 10^{-2}$	0.00000
0.12500	$6.56580 \cdot 10^{-4}$	4.07058	$2.33310 \cdot 10^{-2}$	2.95304	0.12500	$5.35320 \cdot 10^{-5}$	5.03558	$2.13040 \cdot 10^{-3}$	3.95445
$3.12500 \cdot 10^{-2}$	$3.72040 \cdot 10^{-5}$	4.14144	$2.81290 \cdot 10^{-3}$	3.05212	$3.12500 \cdot 10^{-2}$	$1.60410 \cdot 10^{-6}$	5.06057	$1.33710 \cdot 10^{-4}$	3.99395
$7.81250 \cdot 10^{-3}$	$2.14450 \cdot 10^{-6}$	4.11674	$3.39100 \cdot 10^{-4}$	3.05228	$7.81250 \cdot 10^{-3}$	$4.90550 \cdot 10^{-8}$	5.03122	$8.35860 \cdot 10^{-6}$	3.99970
$1.95313 \cdot 10^{-3}$	$1.28550 \cdot 10^{-7}$	4.06024	$4.15980 \cdot 10^{-5}$	3.02712	$1.95313 \cdot 10^{-3}$	$1.51810 \cdot 10^{-9}$	5.01406	$5.22400 \cdot 10^{-7}$	4.00003
$4.88281 \cdot 10^{-4}$	$7.88930 \cdot 10^{-9}$	4.02629	$5.15650 \cdot 10^{-6}$	3.01205	$4.88281 \cdot 10^{-4}$	$4.72290 \cdot 10^{-11}$	5.00645	$3.26500 \cdot 10^{-8}$	4.00000

Table 6.2: Numerical convergence rate computations of the velocity vector field DG FEM approximation using AptoPy generated code applied to the Navier-Stokes equations in (6.3.1).

$\ell - 1 = 0$					$\ell - 1 = 1$				
h^2	$\|p - p_h\|_{L_2(\Omega)}$	R_{L_2}	$\|p - p_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|p - p_h\|_{L_2(\Omega)}$	R_{L_2}	$\|p - p_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	3.89160	0.00000	16.27800	0.00000	0.50000	2.41400	0.00000	20.87500	0.00000
0.12500	2.70700	0.52367	16.03600	0.02161	0.12500	0.65897	1.87314	10.35800	1.01103
$3.12500 \cdot 10^{-2}$	1.68420	0.68463	15.89500	0.01274	$3.12500 \cdot 10^{-2}$	0.18150	1.86024	5.33800	0.95637
$7.81250 \cdot 10^{-3}$	0.94404	0.83514	15.83400	0.00555	$7.81250 \cdot 10^{-3}$	$4.78150 \cdot 10^{-2}$	1.92443	2.74400	0.96002
$1.95313 \cdot 10^{-3}$	0.49626	0.92775	15.81400	0.00182	$1.95313 \cdot 10^{-3}$	$1.22270 \cdot 10^{-2}$	1.96739	1.39180	0.97933
$4.88281 \cdot 10^{-4}$	0.25344	0.96945	15.80800	0.00055	$4.88281 \cdot 10^{-4}$	$3.08690 \cdot 10^{-3}$	1.98584	0.70067	0.99014
$\ell - 1 = 2$					$\ell - 1 = 3$				
h^2	$\|p - p_h\|_{L_2(\Omega)}$	R_{L_2}	$\|p - p_h\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|p - p_h\|_{L_2(\Omega)}$	R_{L_2}	$\|p - p_h\|_{H^1(\Omega)}$	R_{H^1}
0.50000	0.54864	0.00000	8.09560	0.00000	0.50000	$8.61880 \cdot 10^{-2}$	0.00000	2.05510	0.00000
0.12500	$8.62010 \cdot 10^{-2}$	2.67008	2.86500	1.49860	0.12500	$5.28260 \cdot 10^{-3}$	4.02817	0.25833	2.99192
$3.12500 \cdot 10^{-2}$	$1.30070 \cdot 10^{-2}$	2.72842	0.89859	1.67280	$3.12500 \cdot 10^{-2}$	$2.87590 \cdot 10^{-4}$	4.19916	$2.84180 \cdot 10^{-2}$	3.18434
$7.81250 \cdot 10^{-3}$	$1.78210 \cdot 10^{-3}$	2.86764	0.24935	1.84949	$7.81250 \cdot 10^{-3}$	$1.65210 \cdot 10^{-5}$	4.12164	$3.29690 \cdot 10^{-3}$	3.10762
$1.95313 \cdot 10^{-3}$	$2.31600 \cdot 10^{-4}$	2.94387	$6.51300 \cdot 10^{-2}$	1.93678	$1.95313 \cdot 10^{-3}$	$9.89470 \cdot 10^{-7}$	4.06150	$3.97640 \cdot 10^{-4}$	3.05158
$4.88281 \cdot 10^{-4}$	$2.94380 \cdot 10^{-5}$	2.97588	$1.65940 \cdot 10^{-2}$	1.97266	$4.88281 \cdot 10^{-4}$	$6.05620 \cdot 10^{-8}$	4.03017	$4.88750 \cdot 10^{-5}$	3.02429

Table 6.3: Numerical convergence rate computations of the pressure scalar field DG FEM approximation using AptoPy generated code applied to the Navier-Stokes equations in (6.3.1).

6.4 Cylindrical Homogeneous Waveguide

We refer to Section 2.3.2 regarding the electromagnetic vector field solution of the time harmonic formulation of Maxwell's equations applied to a cylindrical waveguide. Let $\Omega = (0, a) \times \{0\} \times (0, d)$, $a, d \in \mathbb{R}$, with boundary $\partial\Omega$ whose unit outward normal vector is denoted by $\mathbf{n}$. This domain is the representation of an axial slice of the cylindrical waveguide geometry. We subdivide the exterior boundary into the axis of symmetry, the open waveguide port, the field source port and the waveguide wall such that $\partial\Omega = \partial\Omega_{\text{sym}} \cup \partial\Omega_{\text{o}} \cup \partial\Omega_{\text{s}} \cup \partial\Omega_D$, where

$$\partial\Omega_{\text{sym}} = \{0\} \times \{0\} \times [0, 1], \tag{6.4.1}$$

$$\partial\Omega_{\text{o}} = [0, a] \times \{0\} \times \{d\}, \tag{6.4.2}$$

$$\partial\Omega_{\text{s}} = [0, a] \times \{0\} \times \{0\}, \tag{6.4.3}$$

$$\partial\Omega_D = \{a\} \times \{0\} \times [0, 1]. \tag{6.4.4}$$

We let $\mathcal{T}_\Omega^h$ be the subdivision of Ω into a structured shape regular mesh of two dimensional simplices of granularity h exploiting azimuthal symmetry. We seek the TM_{0p} solutions of the electric field $\mathbf{E}$ and Lagrange multiplier $\mathfrak{p}$ of the time harmonic formulation of Maxwell's equations in the empty waveguide $(\sigma, \varepsilon, \mu) = (0, \varepsilon_0, \mu_0)$, subject to the boundary conditions of symmetry on $\partial\Omega_{\text{sym}}$, the excited port boundary on $\partial\Omega_{\text{s}}$, the open port boundary on $\partial\Omega_{\text{o}}$ (see [86]) and the perfectly conducting wall $\partial\Omega_D$, i.e., $(\mathbf{E}, \mathfrak{p})$ satisfies

$$\nabla \times (\nabla \times \mathbf{E}) - \mu_0\varepsilon_0\nabla\mathfrak{p} - k_0^2\mathbf{E} = \mathbf{0} \qquad \text{in } \Omega, \tag{6.4.5}$$

$$\nabla \cdot (\mathbf{E}) = 0 \qquad \text{in } \Omega, \tag{6.4.6}$$

$$\mathbf{n} \times (\nabla \times \mathbf{E}) = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{sym}}, \tag{6.4.7}$$

$$\mathbf{n} \times (\nabla \times \mathbf{E}) + j\frac{k_0^2}{\beta_{np}} (\mathbf{n} \times \mathbf{E}_{\text{w}}) \times \mathbf{n} = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{s}}, \tag{6.4.8}$$

$$\mathbf{n} \times (\nabla \times \mathbf{E}) - j\frac{k_0^2}{\beta_{np}} (\mathbf{n} \times \mathbf{E}) \times \mathbf{n} = \mathbf{0} \qquad \text{on } \partial\Omega_{\text{o}}, \tag{6.4.9}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega_D, \tag{6.4.10}$$

where $k_0 = \sqrt{\varepsilon_0\mu_0}\omega$. The electric vector field solution is

$$\mathbf{E}_{\text{w}} = \left(-j\frac{\beta_{0p}}{k_c^2}\frac{\partial E_z}{\partial r},\ 0,\ E_z, \right)^\top, \tag{6.4.11}$$

where

$$E_z = E_{0z} J_0\left(\frac{X_{0p}}{a} r\right) e^{-j\beta_{0p}z},\quad \beta_{0p} = \sqrt{\mu_0\varepsilon_0\omega^2 - \left(\frac{X_{0p}}{a}\right)^2},\quad k_c = \frac{X_{0p}}{a}. \tag{6.4.12}$$

We choose an empty waveguide with radius $a = 5\,\text{cm}$, enclosing an electromagnetic field operating at frequency $f = 2.45\,\text{GHz}$ in a TM_{01} mode which satisfies the waveguide propagation condition. We further set $d = 2.25\pi/\beta_{01}$, i.e., $9/8$ wavelengths of the incident field. The DG FEM formulation is to find $(\mathbf{E}_h, \mathfrak{p}_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h)$ such that

$$a_\Omega^{\text{Max}}(\mathbf{E}_h, \mathbf{F}_h) + \mathcal{N}_\Omega^{\varepsilon\nabla\mathfrak{p}}(\mathfrak{p}_h; \mathbf{F}_h) + \mathcal{N}_\Omega^{\nabla\cdot(\varepsilon\mathbf{E})}(\mathbf{E}_h; \mathfrak{q}_h) - \int_\Omega k^2 \mathbf{E}_h \cdot \overline{\mathbf{F}}_h \, \mathrm{d}\mathbf{x} = 0 \tag{6.4.13}$$

for all $(\mathbf{F}_h, \mathfrak{q}_h) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times V_{\ell+1}(\mathcal{T}_\Omega^h)$.

The AptoPy code required to generate the DG FEM formulation of this problem is shown in Figure 6.4. The convergence rates of the electric field and Lagrange multiplier variables are presented in Tables 6.4 and 6.5, respectively. As required to verify the automatically generated code, the computed convergence rates match those predicted by the theory stated in the L_2 norm in equation (6.1.3) and the $H(\text{curl})$ norm stated in equation (6.1.5).

```
# Physical and waveguide constants
X_01 = 2.404825557695772
omega = 2*pi*2.45e9
k_0 = sqrt(mu_0*eps_0)*omega
beta_01 = sqrt(k_0**2 - (X_01/a)**2)
kc = X_01/a

# Analytical solution - see Liao 1990, Microwave Devices & Circuits
Ez = besselj(0, X_01*r/a)*exp(-sympy.I*beta_01*z)
Er = -sympy.I*beta_01/kc**2*diff(Ez, r)
E_wg = Matrix([Er, 0.0, Ez])

V = FemComplexVectorFunctionSpace(mesh, poly_order=1, element_type='DG')
E, F = V.trial(), conj(V.test())

Q = FemComplexFunctionSpace(mesh, poly_order=2, element_type='DG')

# Port boundary conditions - see Jin 2012, The Finite Element Method in
    Electromagnetics
port_o_pc = NeumannBC(dS_p2, sympy.I*k_0**2/beta_01*cross(cross(n, E), n))
port_s_bc = NeumannBC(dS_p1, -sympy.I*k_0**2/beta_01*cross(cross(n, E_wg), n))
symmetry_bc = NeumannBC(dSsym, 0.0)
conductor_bc = DirichletBC(dSD, 0.0)
bcs = [port_s_bc, port_o_pc, symmetry_bc, conductor_bc]

pe = MaxwellOperatorWithInvolution(mesh, V, Q, bcs)
residual = pe.generate_fem_formulation() - k_0**2*dot(E, F)*dx
residual *= r
```

Figure 6.4: AptoPy automatic computation of the DG FEM formulation of the time harmonic formulation of Maxwell's equations applied to a cylindrical waveguide.

$\ell = 1$					$\ell = 2$				
h^2	$\|\mathbf{E}_w - \mathbf{E}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{E}_w - \mathbf{E}_h\|_{H(\text{curl};\Omega)}$	$R_{H(\text{curl})}$	h^2	$\|\mathbf{E}_w - \mathbf{E}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{E}_w - \mathbf{E}_h\|_{H(\text{curl};\Omega)}$	$R_{H(\text{curl})}$
0.5000	0.1661	0.0000	6.0639	0.0000	0.5000	$8.0650 \cdot 10^{-2}$	0.0000	2.8598	0.0000
0.1250	$8.2312 \cdot 10^{-2}$	1.0129	3.0227	1.0044	0.1250	$9.1898 \cdot 10^{-3}$	3.1336	0.4321	2.7265
$3.1250 \cdot 10^{-2}$	$4.2607 \cdot 10^{-2}$	0.9500	1.6156	0.9038	$3.1250 \cdot 10^{-2}$	$9.4697 \cdot 10^{-4}$	3.2786	$8.8995 \cdot 10^{-2}$	2.2795
$7.8125 \cdot 10^{-3}$	$1.5216 \cdot 10^{-2}$	1.4855	0.6422	1.3310	$7.8125 \cdot 10^{-3}$	$8.7847 \cdot 10^{-5}$	3.4303	$2.1463 \cdot 10^{-2}$	2.0519
$1.9531 \cdot 10^{-3}$	$4.1447 \cdot 10^{-3}$	1.8762	0.2347	1.4520	$1.9531 \cdot 10^{-3}$	$9.3860 \cdot 10^{-6}$	3.2264	$5.3508 \cdot 10^{-3}$	2.0040
$4.8828 \cdot 10^{-4}$	$1.0580 \cdot 10^{-3}$	1.9699	$9.9853 \cdot 10^{-2}$	1.2331	$4.8828 \cdot 10^{-4}$	$1.1139 \cdot 10^{-6}$	3.0749	$1.3369 \cdot 10^{-3}$	2.0009
$\ell = 3$					$\ell = 4$				
h^2	$\|\mathbf{E}_w - \mathbf{E}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{E}_w - \mathbf{E}_h\|_{H(\text{curl};\Omega)}$	$R_{H(\text{curl})}$	h^2	$\|\mathbf{E}_w - \mathbf{E}_h\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathbf{E}_w - \mathbf{E}_h\|_{H(\text{curl};\Omega)}$	$R_{H(\text{curl})}$
0.5000	$1.3178 \cdot 10^{-2}$	0.0000	0.5519	0.0000	0.5000	$2.0372 \cdot 10^{-3}$	0.0000	0.1171	0.0000
0.1250	$6.6098 \cdot 10^{-4}$	4.3174	$5.6578 \cdot 10^{-2}$	3.2861	0.1250	$5.0226 \cdot 10^{-5}$	5.3420	$7.0406 \cdot 10^{-3}$	4.0561
$3.1250 \cdot 10^{-2}$	$3.3570 \cdot 10^{-5}$	4.2994	$6.9275 \cdot 10^{-3}$	3.0298	$3.1250 \cdot 10^{-2}$	$1.5013 \cdot 10^{-6}$	5.0642	$4.3767 \cdot 10^{-4}$	4.0078
$7.8125 \cdot 10^{-3}$	$1.9878 \cdot 10^{-6}$	4.0779	$8.6336 \cdot 10^{-4}$	3.0043	$7.8125 \cdot 10^{-3}$	$4.6277 \cdot 10^{-8}$	5.0198	$2.7102 \cdot 10^{-5}$	4.0134
$1.9531 \cdot 10^{-3}$	$1.2266 \cdot 10^{-7}$	4.0184	$1.0761 \cdot 10^{-4}$	3.0042	$1.9531 \cdot 10^{-3}$	$1.4407 \cdot 10^{-9}$	5.0055	$1.6827 \cdot 10^{-6}$	4.0095

Table 6.4: Numerical convergence rate computations of the TM_{01} waveguide electric field vector field DG FEM approximation using AptoPy generated code applied to the time harmonic formulation of Maxwell's equations (6.4.5).

$\ell+1=2$					$\ell+1=3$				
h^2	$\|\mathfrak{p}\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathfrak{p}\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|\mathfrak{p}\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathfrak{p}\|_{H^1(\Omega)}$	R_{H^1}
0.5000	$9.2716 \cdot 10^{-4}$	0.0000	0.1148	0.0000	0.5000	$2.4498 \cdot 10^{-5}$	0.0000	$1.9097 \cdot 10^{-3}$	0.0000
0.1250	$6.0613 \cdot 10^{-5}$	3.9351	$1.6648 \cdot 10^{-2}$	2.7853	0.1250	$1.9204 \cdot 10^{-6}$	3.6732	$2.7609 \cdot 10^{-4}$	2.7901
$3.1250 \cdot 10^{-2}$	$6.2880 \cdot 10^{-6}$	3.2690	$2.9678 \cdot 10^{-3}$	2.4879	$3.1250 \cdot 10^{-2}$	$9.4193 \cdot 10^{-8}$	4.3496	$3.0134 \cdot 10^{-5}$	3.1957
$7.8125 \cdot 10^{-3}$	$7.4759 \cdot 10^{-7}$	3.0723	$6.1856 \cdot 10^{-4}$	2.2624	$7.8125 \cdot 10^{-3}$	$3.9280 \cdot 10^{-9}$	4.5838	$3.4157 \cdot 10^{-6}$	3.1411
$1.9531 \cdot 10^{-3}$	$8.8630 \cdot 10^{-8}$	3.0764	$1.3895 \cdot 10^{-4}$	2.1543	$1.9531 \cdot 10^{-3}$	$1.7274 \cdot 10^{-10}$	4.5071	$4.0134 \cdot 10^{-7}$	3.0893
$4.8828 \cdot 10^{-4}$	$1.0800 \cdot 10^{-8}$	3.0368	$3.3182 \cdot 10^{-5}$	2.0661	$4.8828 \cdot 10^{-4}$	$4.6551 \cdot 10^{-11}$	1.8917	$4.9896 \cdot 10^{-8}$	3.0078

$\ell+1=4$					$\ell+1=5$				
h^2	$\|\mathfrak{p}\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathfrak{p}\|_{H^1(\Omega)}$	R_{H^1}	h^2	$\|\mathfrak{p}\|_{L_2(\Omega)}$	R_{L_2}	$\|\mathfrak{p}\|_{H^1(\Omega)}$	R_{H^1}
0.5000	$2.5377 \cdot 10^{-6}$	0.0000	$2.7239 \cdot 10^{-4}$	0.0000	0.5000	$3.1701 \cdot 10^{-7}$	0.0000	$5.2702 \cdot 10^{-5}$	0.0000
0.1250	$8.2540 \cdot 10^{-8}$	4.9423	$1.7595 \cdot 10^{-5}$	3.9524	0.1250	$4.8514 \cdot 10^{-9}$	6.0300	$1.4003 \cdot 10^{-6}$	5.2340
$3.1250 \cdot 10^{-2}$	$2.5452 \cdot 10^{-9}$	5.0192	$1.0973 \cdot 10^{-6}$	4.0031	$3.1250 \cdot 10^{-2}$	$5.0686 \cdot 10^{-10}$	3.2587	$6.9033 \cdot 10^{-8}$	4.3423
$7.8125 \cdot 10^{-3}$	$7.8412 \cdot 10^{-11}$	5.0206	$6.8397 \cdot 10^{-8}$	4.0039	$7.8125 \cdot 10^{-3}$	$4.0781 \cdot 10^{-10}$	0.3137	$8.4566 \cdot 10^{-8}$	-0.2928
$1.9531 \cdot 10^{-3}$	$2.6687 \cdot 10^{-11}$	1.5549	$7.9664 \cdot 10^{-8}$	-0.2200	$1.9531 \cdot 10^{-3}$	$5.5347 \cdot 10^{-11}$	2.8813	$4.3736 \cdot 10^{-8}$	0.9513

Table 6.5: Numerical convergence rate computations of the TM_{01} waveguide Lagrange multipler DG FEM approximation using AptoPy generated code applied to the time harmonic formulation of Maxwell's equations (6.4.5).

6.5 Cylindrical Cavity Resonator

We refer to Section 2.3.3 for the derivation of the eigenvalue solution of the cylindrical cavity resonator employed in this convergence test. As in the waveguide test case, let the domain $\Omega = (0, a) \times \{0\} \times (0, d)$ with boundary $\partial\Omega$, whose unit outward normal vector is denoted by $\mathbf{n}$, be the representation of an axial slice of the cylindrical resonator geometry. We subdivide the boundary into the axis of symmetry and the perfect conductor cavity walls $\partial\Omega = \partial\Omega_N \cup \partial\Omega_D$. We construct the test case for the empty cavity of dimensions $(a, d) = (1, \pi)$ and material properties $(\sigma, \varepsilon, \mu) = (0, \varepsilon_0, \mu_0)$, enclosing an azimuthally symmetric electromagnetic field operating in a TM_{0pq} mode. The resonant frequencies ω_r are calculated from the eigenpair solutions $(\mathbf{0} \neq \mathbf{E}, k_0^2) \in \mathbb{R}^3 \times \mathbb{R}$ of

$$\nabla \times (\nabla \times \mathbf{E}) = k_0^2 \mathbf{E} \qquad \text{in } \Omega, \tag{6.5.1}$$

$$\mathbf{n} \times \mathbf{E} = \mathbf{0} \qquad \text{on } \partial\Omega \setminus \partial\Omega_D, \tag{6.5.2}$$

$$\mathbf{n} \times (\nabla \times \mathbf{E}) = \mathbf{0} \qquad \text{on } \partial\Omega_N. \tag{6.5.3}$$

Here, each resonant frequency $\omega_r = {}^{k_0}/_{\sqrt{\varepsilon_0 \mu_0}}$ of the TM_{0pq} mode is analytically determined by

$$\omega_r = \frac{1}{\sqrt{\mu_0 \varepsilon_0}} \sqrt{X_{0p}^2 + q^2}, \quad p \geq 1,\ q \geq 0. \tag{6.5.4}$$

Let $\mathcal{T}_\Omega^h$ be the subdivision of Ω into a structured shape regular mesh of two dimensional simplices of diameter h. The discrete form of the eigen problem (6.5.1) is to find $(\mathbf{0} \neq \mathbf{E}_h, k_{0h}^2) \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h) \times \mathbb{R}$ such that

$$a_\Omega^{\mathrm{Max}}(\mathbf{E}_h, \mathbf{F}_h) = \int_\Omega k_{0h}^2 \mathbf{E}_h \cdot \mathbf{F}_h \, \mathrm{d}\mathbf{x} \tag{6.5.5}$$

for all $\mathbf{F}_h \in \mathbf{V}_\ell^d(\mathcal{T}_\Omega^h)$.

The AptoPy code required to generate the DG FEM formulation of the eigenpair problem is shown in Figure 6.5. Examples of the convergence rates of the computed eigenvalues are presented in Figure 6.6. The computed convergence rate matches well with that stated in equation (6.1.6), where we note that the convergence rate of the TM_{010} (first) eigenvalue with approximating finite element polynomial order $\ell = 4$ is stalled by machine precision on the finer meshes. We also note the numerical approximation error of the computed eigenvalue increases with the eigenvalue index as expected, cf. Buffa et al. [25].

```
V = FemVectorFunctionSpace(mesh, poly_order=1, element_type='DG')
E, F = V.trial(), V.test()
sigma = 10.0*V.penalisation()[0]

def A(u, v):
    val = dot(curl(u), curl(v))*r*dx \
          - dot(cross_jump(u), avg(curl(v)))*r*(dInt + dS) \
          - dot(cross_jump(v), avg(curl(u)))*r*(dInt + dS) \
          + sigma*dot(cross_jump(u), cross_jump(v))*r*(dInt + dS)
    return val

lhs = A(E, F)
rhs = dot(E, F)*r*dx

U = eigen_solve(mesh, lhs, rhs, V)
output_eigenvalues(U, lambda ev: ev if ev > 1e-4 else None)
```

Figure 6.5: AptoPy automatic computation of the DG FEM formulation of the eigenvalue problem of the Maxwell operator.

6.6 Performance

A key issue in the design of AptoPy, whilst ensuring robustness, is the consideration of computation time. In this section we demonstrate the computation time undertaken by AptoPy on a standard laptop computer featuring a 2.3 GHz Intel Core i7-3615QM processor. For a series of equations, a tabulation of example run times required for AptoPy to automatically compute the finite element formulation and output the required Fortran code for use with AptoFEM is presented in Table 6.6.

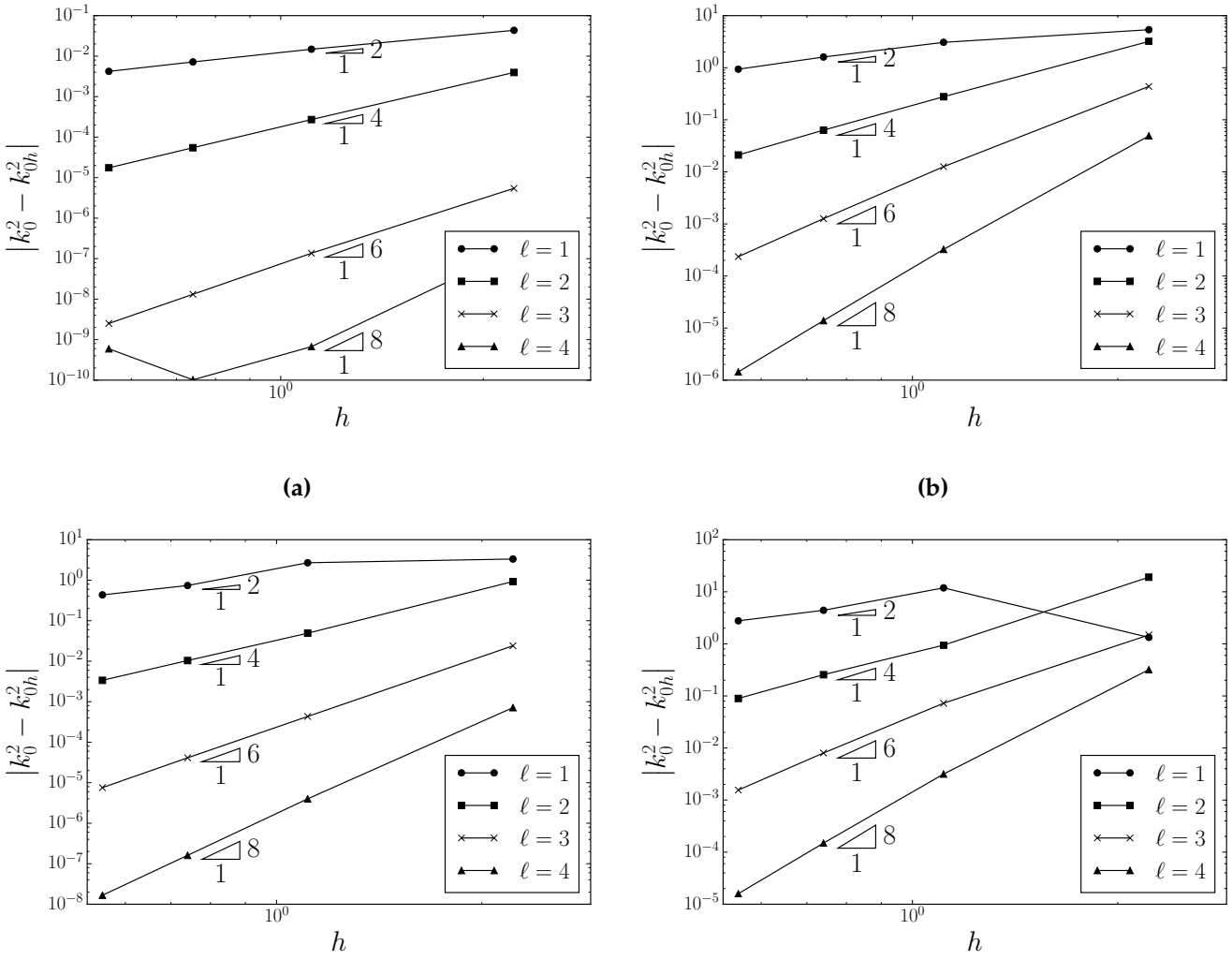


Figure 6.6: Numerical computations of eigenvalue convergence rates of the a) TM_{010}, b) TM_{013}, c) TM_{020} and d) TM_{023} electromagnetic field harmonic modes corresponding to the 1st, 4th, 6th and 10th non-zero eigenvalues respectively.

Finite Element Formulation	Variables	Gâteaux Derivative	Total
CG Poisson	1	0.013 023	0.098 761
CG Poisson cylindrical	1	0.031 952	0.139 858
CG Stokes	3	0.175 607	0.329 000
CG incompressible Navier-Stokes	3	0.178 140	0.328 386
DG Poisson	1	0.295 699	0.447 525
DG Poisson cylindrical	1	0.373 568	0.547 341
DG Poisson 3D cube	1	0.577 185	0.799 675
DG Maxwell eigenvalue (real valued) [25]	2	0.932 15	1.117 097
DG Stokes [39]	3	1.582 049	1.903 392
DG incompressible Navier-Stokes	3	2.766 409	3.372 381
DG Maxwell waveguide [80]	4	5.428 481	6.171 783
DG linearised magnetohydrodynamics [82]	6	9.011 347	9.863 774
DG Maxwell waveguide (Lagrange multiplier) [80]	6	12.911 531	14.014 545
DG compressible Navier-Stokes [66]	4	28.173 629	34.051 378

Table 6.6: Examples of the total time elapsed (s) for AptoPy to calculate the finite element formulation and output the Fortran code for use by AptoFEM, measured using Python's `time` module. Note the most expensive operation is the computation of the Gâteaux derivative. All boundary value problems are calculated on the 2D unit square in Cartesian coordinates unless stated otherwise. We note the large growth in computation time of the Gâteaux derivative for the DG compressible Navier-Stokes equations. This is due in part to the greater number of solution variables, but primarily the high degree of nonlinearity. We investigate this issue in detail in Section 6.6.3.

6.6.1 Gâteaux Derivative

As already briefly mentioned in Section 5.1.6, the cost of symbolic computational algebra is high, particularly differentiation. As the complexity and number of variables in a given partial differential problem grows, so does the combinatoric complexity of the sympy expression tree.

To demonstrate the caching optimisation offered by the `AptoFunction` construct, consider the semilinear PDE: find u such that

$$-\nabla \cdot (A(u)\nabla u) = 0 \qquad \text{in } \Omega, \tag{6.6.1}$$

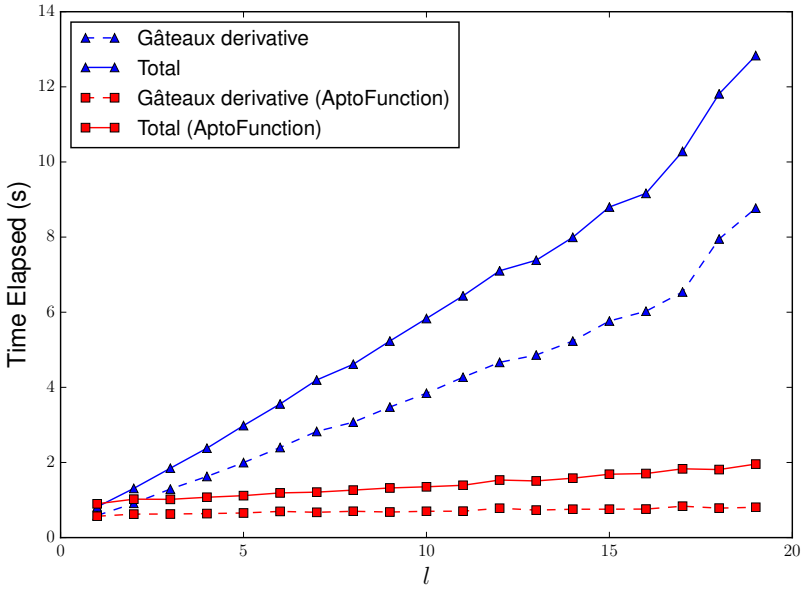
$$u = 1 \qquad \text{on } \partial\Omega. \tag{6.6.2}$$

Selecting $A(u)$ to be a power series, the computation time taken for AptoPy to compute and generate the Fortran code required to solve the DG FEM formulation of (6.6.1) is shown in Figure 6.7. Two cases are shown, one constructing $A(u)$ as a power series and the other as a product series, both truncated at the l^{th} term. Growth of computation time of the Gâteaux derivative is expected to be linear for the power series coefficient. Exponential growth for the product series is expected as the computation of the chain and product rules of differentiation become geometrically more expensive as the multiplication operators in the sympy expression tree become more nested. The use of the `AptoFunction` requires that these derivatives each need only be computed once through the entire process of computing the system Gâteaux derivative, thereby vastly reducing computational cost.

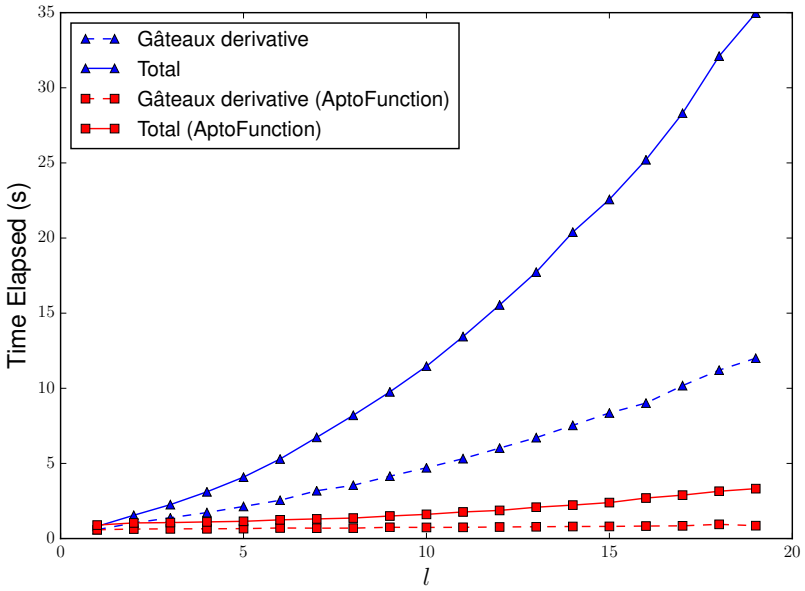
Once the Gâteaux derivative is computed, the remaining computation time is dominated by forming and generating the required Fortran code. As the complexity of the coefficient $A(u)$ grows, so does the length of the required Fortran code. The encapsulation of $A(u)$ into an `AptoFunction` (which at the Fortran level is wrapped in a `function`) ensures that the verbosity of code required to compute $A(u)$ is generated only once. In the context of solving the MPA-CVD reactor model where many coefficients are power series of solution variables, employing AptoPy's caching optimisation is a necessity.

6.6.2 Number of Variables

It is evident that the greater the number of variables, and therefore equations, the greater the computational complexity of computing the finite element formulation.



(a) $A(u) = \sum_{k=1}^{l} \left(u^k + k\right)^k$, `A = sum((u**k + k)**k for k in range(1, l+1))`



(b) $A(u) = \prod_{k=1}^{l} \left(u^k + k\right)^k$, `A = 1; for k in range(1, l+1): A *= (u**k + k)**k`

Figure 6.7: Computation time comparison between the caching of sympy and the specialised caching of AptoPy using `AptoFunction`. Here $A(u)$ is the nonlinear diffusion coefficient of the Poisson equation (6.6.1). Note that even with the caching optimisations offered by AptoPy, the time taken to generate the Fortran code still rises with the complexity of $A(u)$. In each case, the `AptoFunction` is generated by calling `A_af = AptoFunction(A, u, 'A_%d'%l)`.

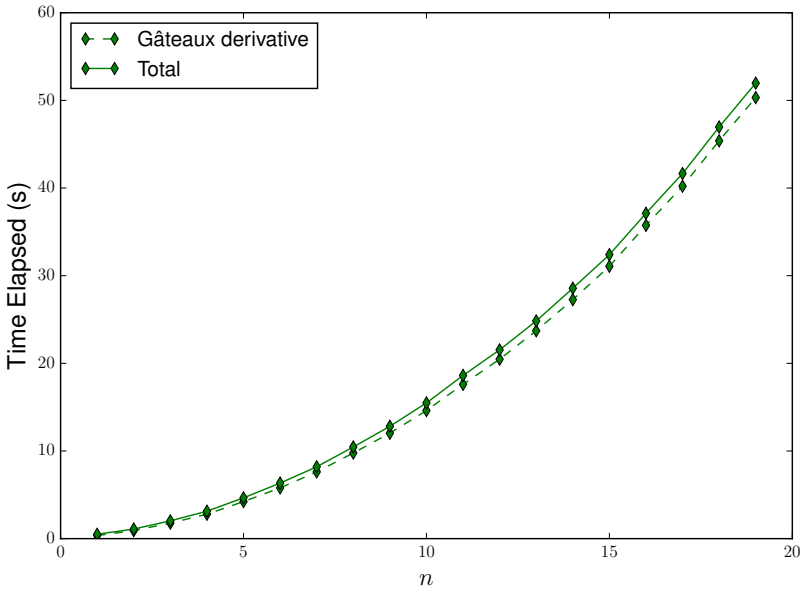


Figure 6.8: Computation time for AptoPy to generate the DG FEM formulation of a system of n equations (6.6.3).

Given a domain Ω with boundary $\partial\Omega$, consider the series of n simultaneous equations:

$$-\nabla \cdot \mathcal{F}^v(u_i) = 0 \qquad \text{in } \Omega, \tag{6.6.3}$$

$$u_i = 0 \qquad \text{on } \partial\Omega, \tag{6.6.4}$$

$i = 1, \ldots, n$, where $\mathcal{F}^v(u) = \nabla u$. The DG FEM formulation is to find $(u_1, \ldots, u_n) \in V_\ell(\mathcal{T}_\Omega^h) \times \ldots \times V_\ell(\mathcal{T}_\Omega^h)$ such that

$$\sum_{i=1}^{n} \mathcal{N}_\Omega^v(u_i; v_i) = 0 \tag{6.6.5}$$

for all $(v_1, \ldots, v_n) \in V_\ell(\mathcal{T}_\Omega^h) \times \ldots \times V_\ell(\mathcal{T}_\Omega^h)$. Here, $\mathcal{T}_\Omega^h$ is the triangulation of Ω. The number of equations in this linear setting affects the computation time required by AptoPy to formulate and generate the required code for AptoFEM as shown in Figure 6.8. The exponential growth of the computation time of the Gâteaux derivative is expected as the number of derivatives required in its calculation is $\mathcal{O}(n^2)$.

6.6.3 Degree of Interdependence

Consider again the system of n Poisson equations (6.6.3). Serving as a test case for AptoPy's performance, we make the modification

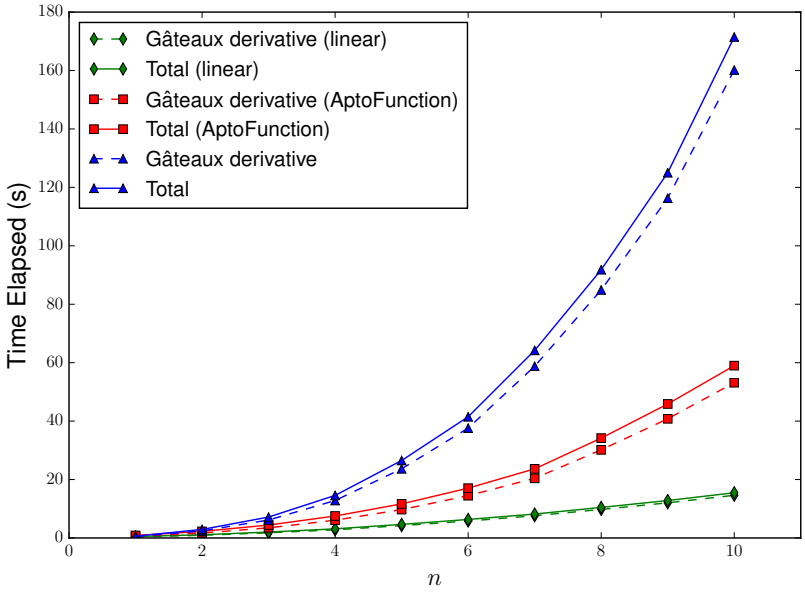
$$\mathcal{F}^v(u) = D\nabla u, \tag{6.6.6}$$

where the nonlinear diffusion coefficient

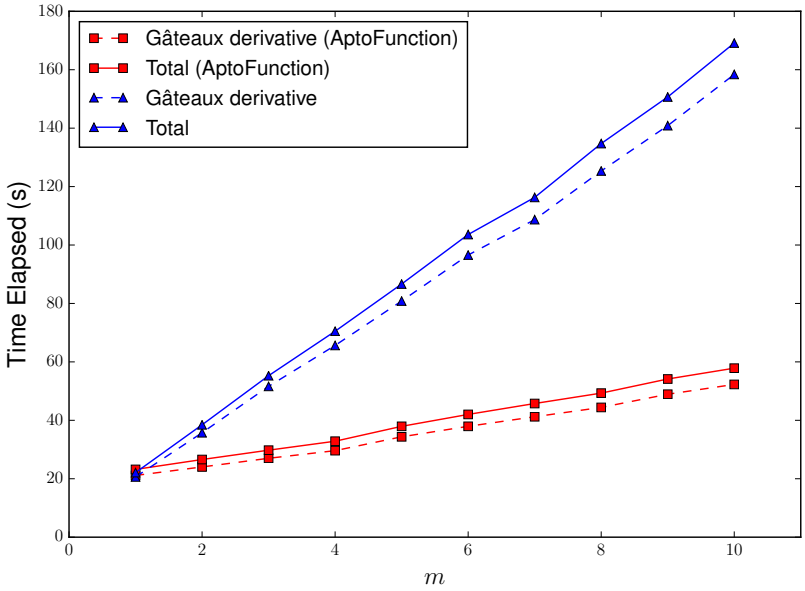
$$D = \sum_{k=1}^{m} u_k, \quad m \leq n. \tag{6.6.7}$$

Here, m introduces a degree of interdependence in the set of simultaneous PDEs which affects the computation time required to formulate the Gâteaux derivative of the finite element formulation. Computation times for a growing number of equations at maximum interdependence $m = n$, and a fixed number of equations with growing interdependence $n = 10$, $1 \leq m \leq 10$, are shown in Figure 6.9.

The rate of exponential growth of computation time with the number of equations at maximum interdependence is drastically increased compared with that of the linear setting. The use of the `AptoFunction` construct alleviates this cost, although does not reduce the growth of computation time to a linear rate. This is due to the number of symbolic derivatives required to be computed is still $\mathcal{O}(n^2)$. For the same reasons as the evaluation of the power series coefficient examined in Section 6.6.1, the degree of interdependence affects the computation time linearly at fixed n.



(a) $m = n$



(b) $n = 10$

Figure 6.9: Computation time for AptoPy's generation of Fortran code required to solve n simultaneous PDEs with viscous operator (6.6.6). The characterisation of the interdependence of the simultaneous PDEs is determined by diffusion coefficient $D = \sum_{k=1}^{m} u_k, \quad m \leq n$.

CHAPTER 7

Numerical experiments

In this chapter we present numerical results for the DG approximation for various CVD reactor geometries and operating conditions. For each of the geometries implemented in this thesis, the finite element mesh $\mathcal{T}_\Omega^h$ is generated using Triangle [124], the DG finite element formulation is generated using AptoPy with underlying sympy framework [133] and AptoPy automatically generates the Fortran code such that the finite element matrix assembly is computed by AptoFEM [1]. The underlying linear systems are solved using the MUltifrontal Massively Parallel solver [3–5] and the eigenpair solutions of microwave cavity resonance are computed using the Arnoldi Package (ARPACK) [95]. The chemical data relating to the dissociation of hydrogen and thermal properties of fused silica for all coefficients and parameters were obtained from the National Institute of Standards and Technology chemistry database [35, 43]. Furthermore, we set the underlying polynomial order $\ell = 1$ for every element in the generated meshes.

We first present numerical examples demonstrating results of the DG solution of the model equations in the simplified CVD geometry shown in Figure 7.1. We then present results calculated for reactor geometries inspired by the ellipsoidal AIXTRON reactor and the LIMHP reactor, both of which are summarised in [9]. Finally, we demonstrate an example geometry optimisation procedure using the LIMHP reactor inspired design as a reference. In each case, we initially solve the DG MPA-CVD model on a coarse mesh, ensuring that when solving on a finer mesh the solution converges to a more precise numerical approximation.

Each reactor presented here is designed for operation with incident electromagnetic field operating at 2.45 GHz. For each geometry, the electromagnetic field will be computed for the case of the empty, air-filled, reactor. The peaks in the magnitude of the

electric field solution will provide an indication of the location of the ignited plasma when the hydrogen vacuum portion of the reactor is implemented. Given a satisfactory electromagnetic field configuration, we then proceed to compute a series of solutions to the DG MPA-CVD model at hydrogen gas mixture pressures of 50 torr, 150 torr and 250 torr and input power in the range 200 W to 1200 W. The quantities of interest are the free electron density in the plasma n_{e}, the number density of atomic hydrogen n_{H}, the reactor temperature T and the microwave power deposition in the plasma P_{ohm}.

7.1 Example 1: A Simple Cylindrical MPA-CVD Reactor

In our first example, we take the arbitrarily designed geometry Ω and mesh $\mathcal{T}_{\Omega}^{h}$ presented in Figure 7.1. The reactor radius from the axis of symmetry to the outer wall is set at $r_{\mathrm{reactor}} = 0.12\,\mathrm{m}$ and the height of the chamber from the chamber floor to its ceiling is set at $h_{\mathrm{reactor}} = 0.24\,\mathrm{m}$. Configuring the electromagnetic field frequency to run at 2.45 GHz it is intended to excite the transversal magnetic TM_{022} harmonic mode. The substrate surface has radius 3 cm at height 5 mm from the chamber floor. In order to confine the ignited plasma to reside above the substrate surface, the base of the quartz window whose width is set at 5 mm is situated $6\,\mathrm{cm} = \frac{1}{4}h_{\mathrm{reactor}}$ from the floor of the chamber. This position of the quartz window exploits the local maximum of the electric field TM_{022} configuration at the base of the reactor. The inlet gas nozzle is situated on the outer chamber wall 1 cm from the quartz window and has width 3 mm.

The mesh is constructed with a higher density of elements in the quartz window, gas inlet and outlet pipes, substrate surface and microwave antenna. The intention is to better resolve the permittivity transition between the vacuum and the air filled cavity, the re-entrant corners of the geometry, the region with a high density of electrons in the plasma and the singularities in the electromagnetic field. The resulting mesh for this simple geometry consists of 44 935 elements with 67 040 interior and 725 boundary faces. The resulting DG finite element formulation with $\ell = 1$ for the MPA CVD model in this configuration consists of 1 358 520 unknown degrees of freedom. This is split between: the velocity field variables $\mathbf{u}$ and p, 78 225; the reactor temperature T, 134 805; the species densities x_{H} and n_{e}, 33 525 each; and the electric field variables $\mathbf{E}$ and $\mathfrak{p}$, 1 078 440.

Solving for the electric field in the empty air-filled reactor geometry, cf. Figure 7.2, yields a similar distribution to the cylindrical cavity resonator operating in a TM_{022}

mode (cf. Figure 2.2). This indicates that the plasma will be ignited as intended above the diamond substrate surface. Furthermore, the lack of a peak in the electric field in the hydrogen gas mixture close to the quartz window should eliminate the possibility of damage to the window from unwanted heating.

With this reactor design, we apply zero inlet gas velocity $\mathbf{u}_{\text{inlet}} = \mathbf{0}$, such that diffusive transport effects dominate. The spatial distribution close to the substrate surface of the plasma density and its properties are presented for varying power input and working pressures in Figures 7.3, 7.4, 7.5 and 7.6. The exploitation of the resonant electric field structure is clear with the shape and peak of the plasma density residing above the substrate surface. Combined with the ignited plasma's shape and position, this leads to the localised peak in the microwave power deposition above the substrate surface. At larger power input the spatial gas temperature distribution is around the temperature required for efficient production of H from H_2 by dissociation, i.e., $T > 3200\,\text{K}$ at power input of 900 W. Figure 7.7 shows the temperature distribution in the quartz window, which does not exceed 650 K, well below its softening point $\approx 1343\,\text{K}$.

The electric field resonance property of the CVD reactor is affected by the permittivity of the generated plasma. Figure 7.8 demonstrates this variation in the electric permittivity in the presence of the plasma. The attenuation of the electric field in the plasma is clear if we consider that it is damped as $\sim e^{-\Im(\varepsilon)\mathbf{x}}$ (see Section 2.3.2). In depth analysis of this coupling is performed in [63].

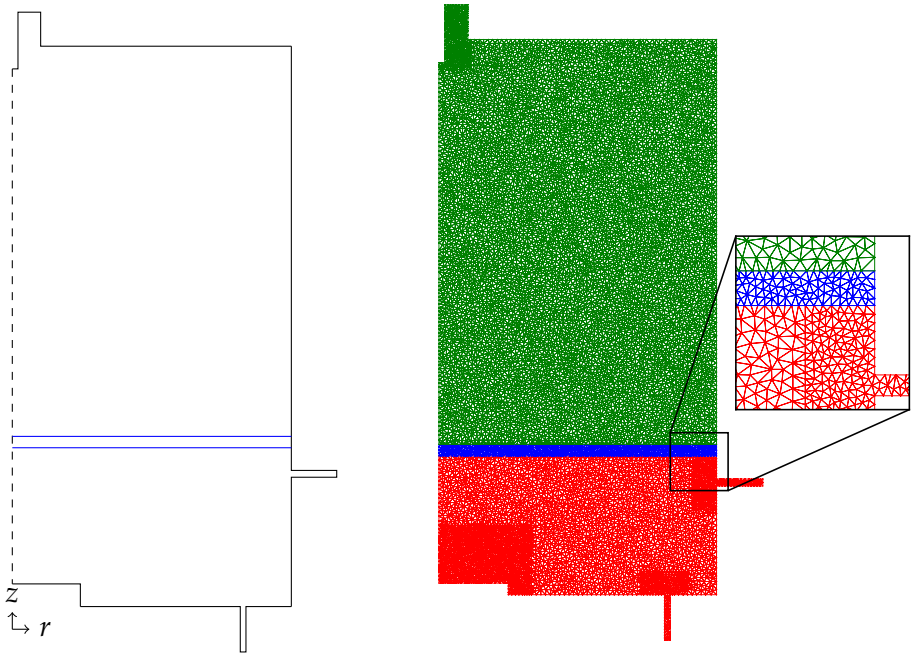


Figure 7.1: Axial slice of a simple chemical vapour deposition reactor geometry at azimuth $\theta = 0$. The lower hydrogen gas vacuum and upper air filled cavity regions are separated by a quartz window highlighted in blue.

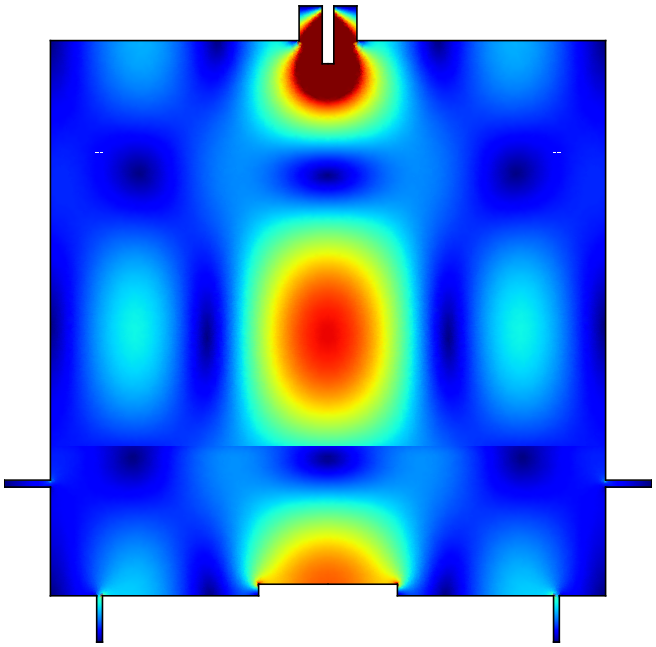


Figure 7.2: Electric field magnitude in the empty simple geometry reactor.

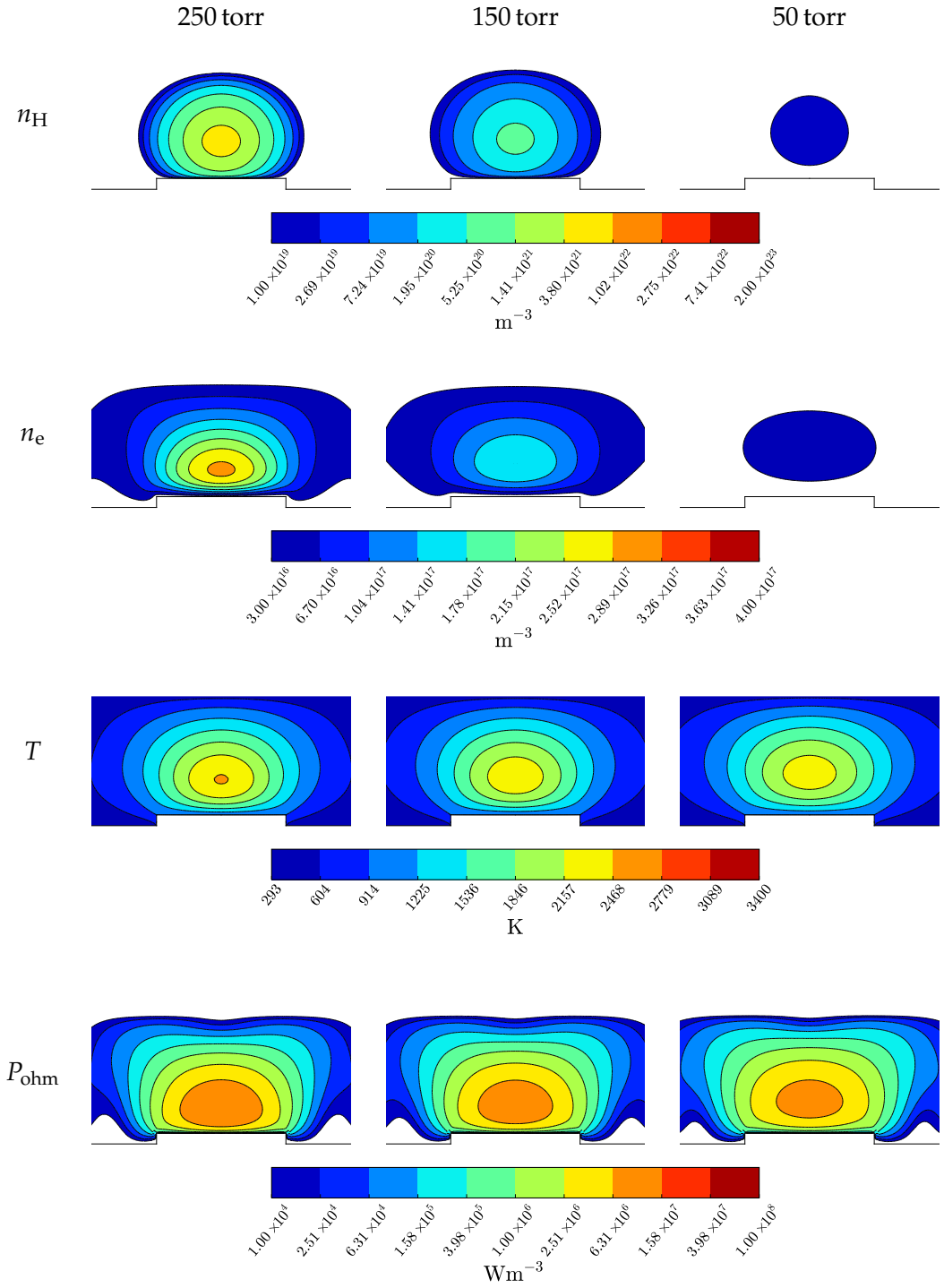


Figure 7.3: The basic reactor plasma shape operating at 500 W.

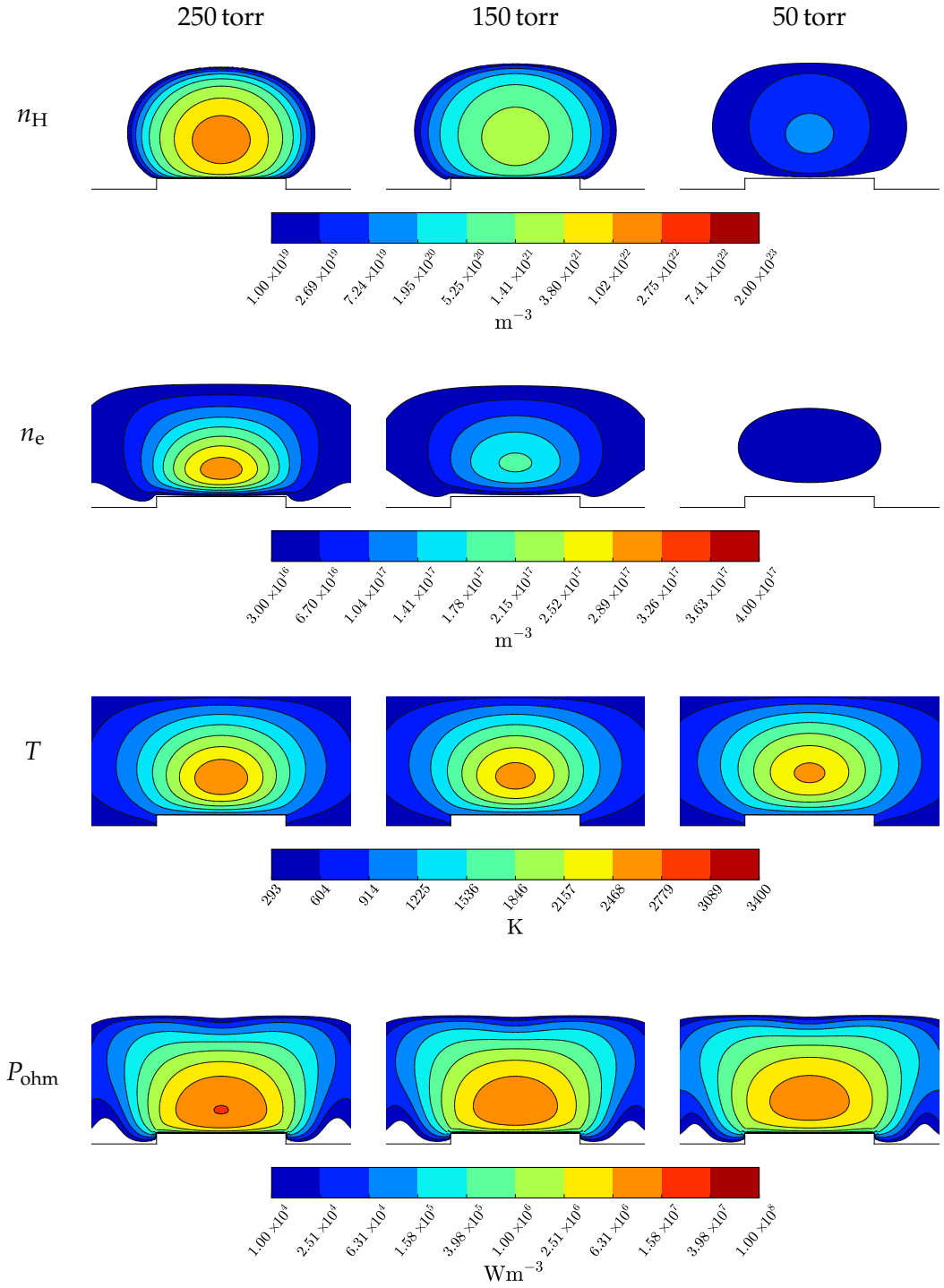


Figure 7.4: The basic reactor plasma shape operating at 600 W.

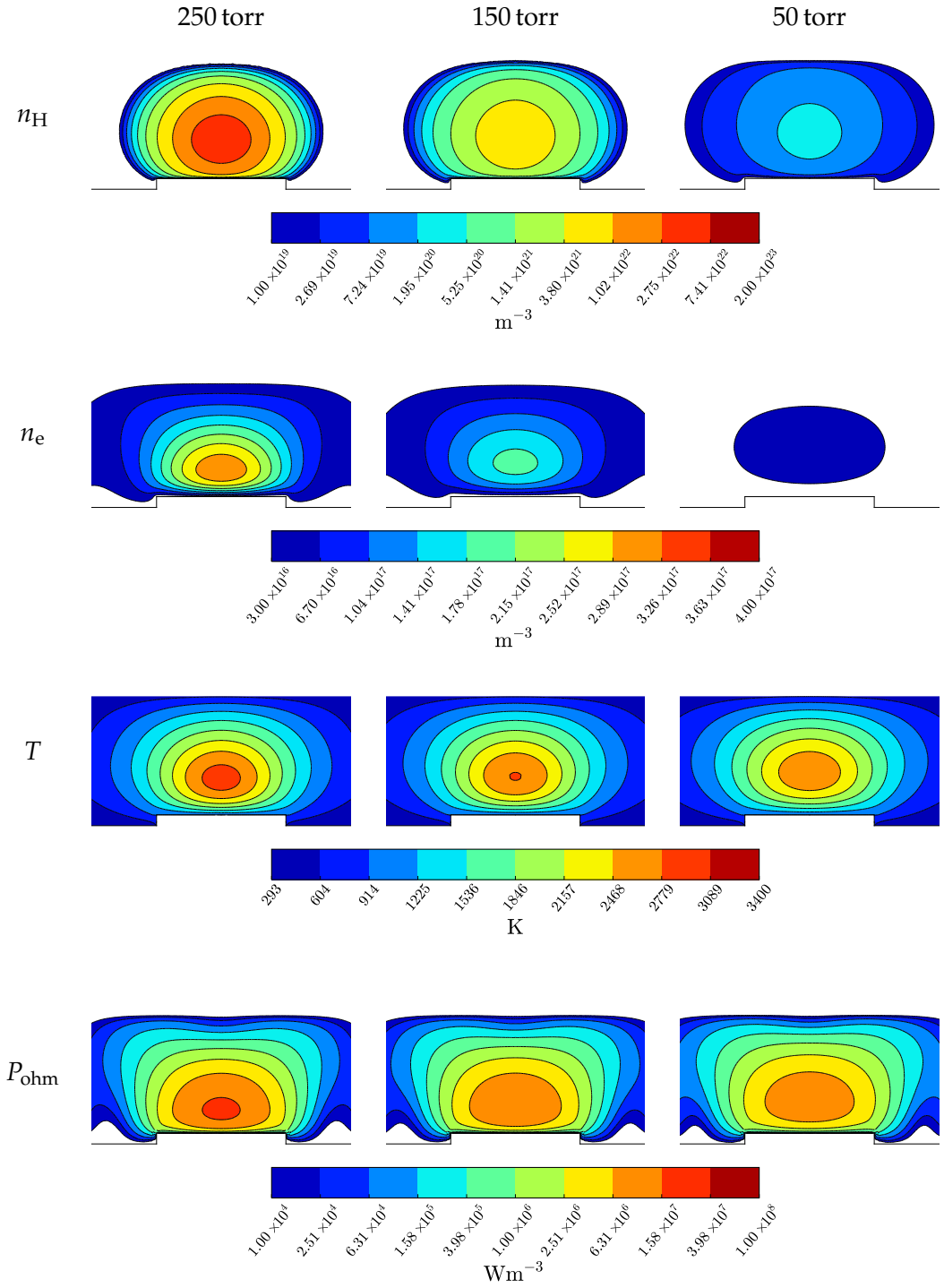


Figure 7.5: The basic reactor plasma shape operating at 700 W.

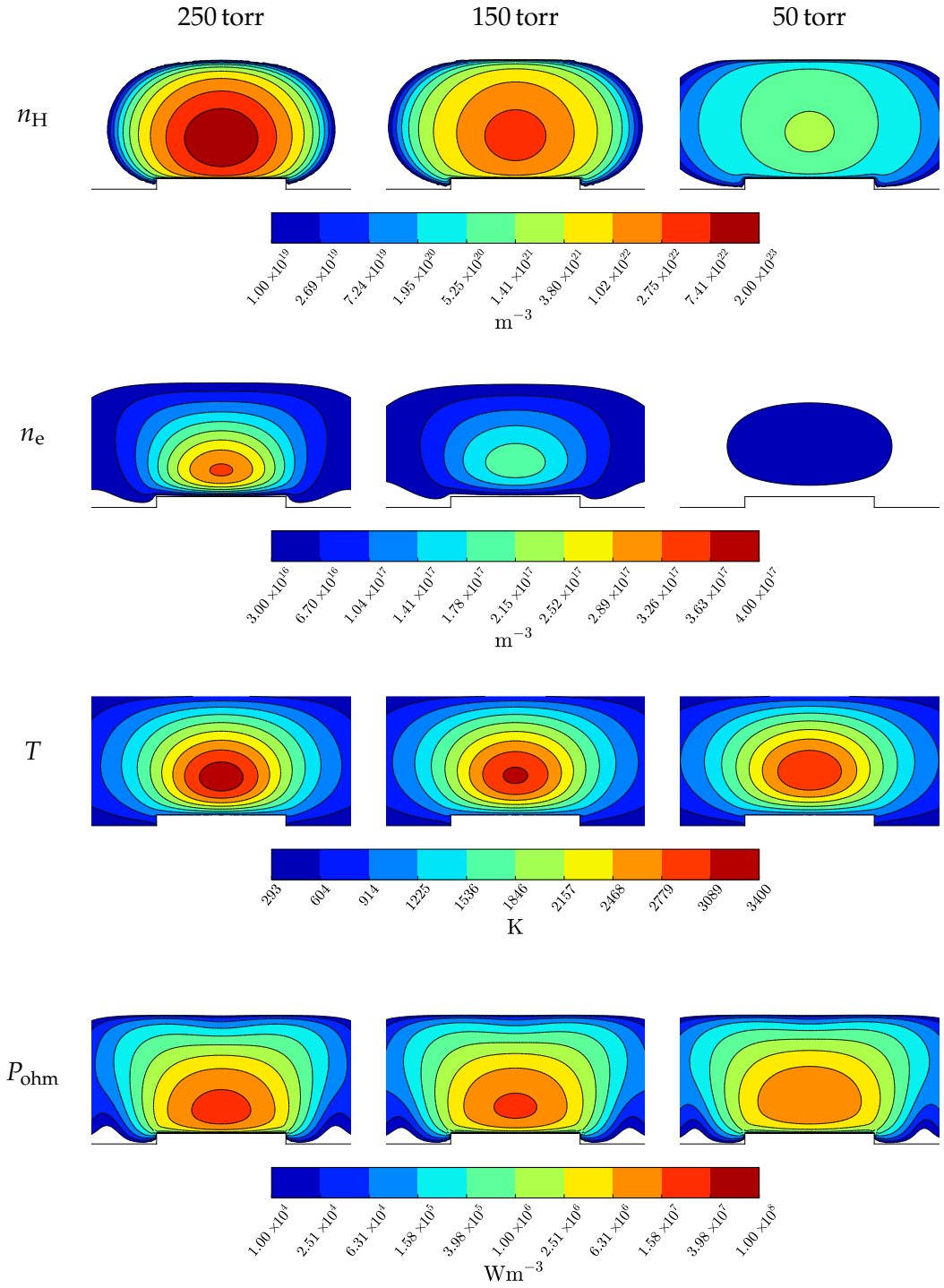


Figure 7.6: The basic reactor plasma shape operating at 900 W.

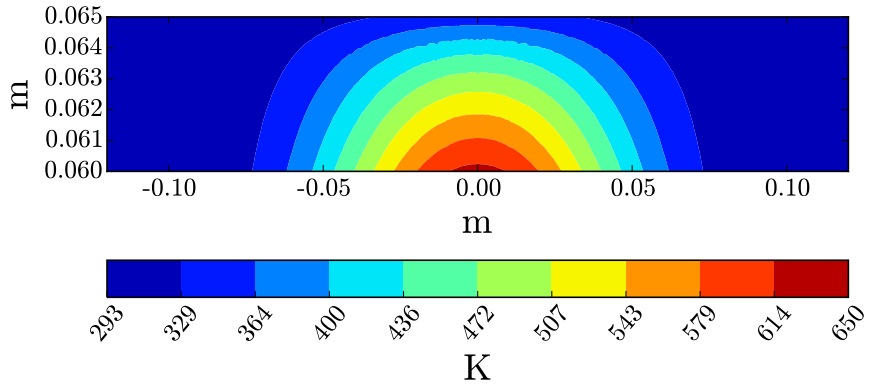


Figure 7.7: Temperature distribution in the quartz window of the simple CVD reactor geometry in Figure 7.1 when operating at a working pressure of 250 torr and power input of 900 W.

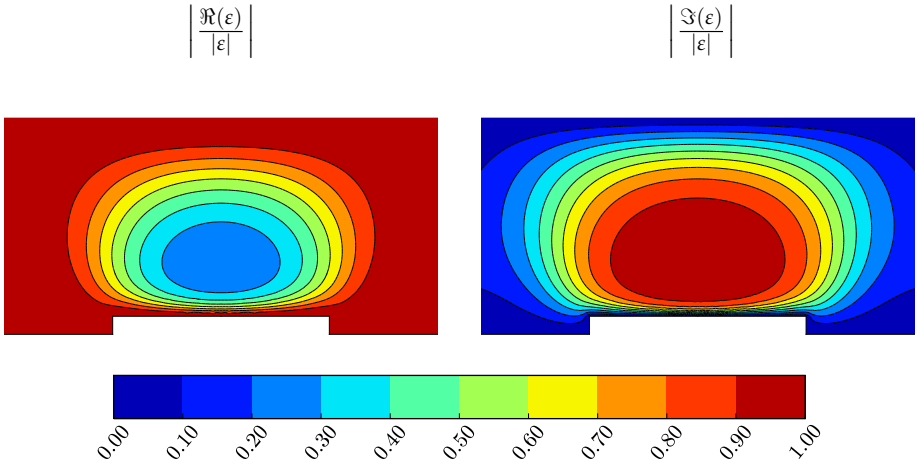


Figure 7.8: The real and imaginary parts of the dielectric constant in the simple reactor geometry operating at a working pressure of 250 torr and power input of 900 W.

7.2 Example 2: The AIXTRON Reactor

Here we apply the DG MPA-CVD model to a design inspired by the ellipsoidal AIXTRON reactor depicted in Figure 7.9. The ellipsoid design is intended to take advantage of its two focal points concentrating the microwave radiation at one point above the substrate surface. Configuring the electric field to run at a frequency of 2.45 GHz in a TM_{036} mode, the ellipse is constructed such that the minor radius $a = 228$ mm and major radius $b = \frac{4}{3}a$. The base of the reactor is located 208 mm from its ellipsoid centre. The antenna is positioned into the cavity at the focal point $\sqrt{b^2 - a^2}$. The diamond substrate surface has radius 2.5 cm and is raised to a height of 5 mm from the reactor floor.

We employ the same procedure as with the basic reactor geometry for meshing. The meshing of the AIXTRON inspired reactor domain is performed with the intention to have greater density of elements close to regions of interest and where large changes in the computed solution are expected. These regions are the quartz bell jar, in order to resolve the change in refractive index of the medium, the gas inlet and outlet pipes, and the region above the substrate to accurately resolve the plasma. The generated mesh consists of 34 243 triangle elements and 51 029 interior and 671 exterior faces. The DG MPA-CVD model computed on this mesh consists of 1 011 102 degrees of freedom.

The DG electric field solution in the empty air-filled cavity is presented in Figure 7.10. It is clear that the peak in the resonant electric field mode lies on the diamond substrate surface. In the subsequent computation of the full DG MPA-CVD reactor simulations, we therefore expect the plasma to reside as intended above the substrate. Furthermore, the small electric field amplitudes around the quartz bell jar should remove the risk of damage through unwanted microwave power deposition and hence heating.

With zero gas inlet velocity $\mathbf{u}_{\text{inlet}} = \mathbf{0}$, we present the array of spatial distributions of the plasma density, microwave plasma deposition, gas temperature and atomic hydrogen located close to the substrate surface in Figures 7.11, 7.12, 7.13 and 7.14. The shape of the plasma is improved by the AIXTRON inspired design over the simple geometry. The peaks in density of the plasma lie closer to the substrate surface and are more evenly spread. Furthermore, the temperature of the quartz bell jar does not exceed 400 K. We note that the results presented here for the AIXTRON inspired reactor are in agreement with those in [127].

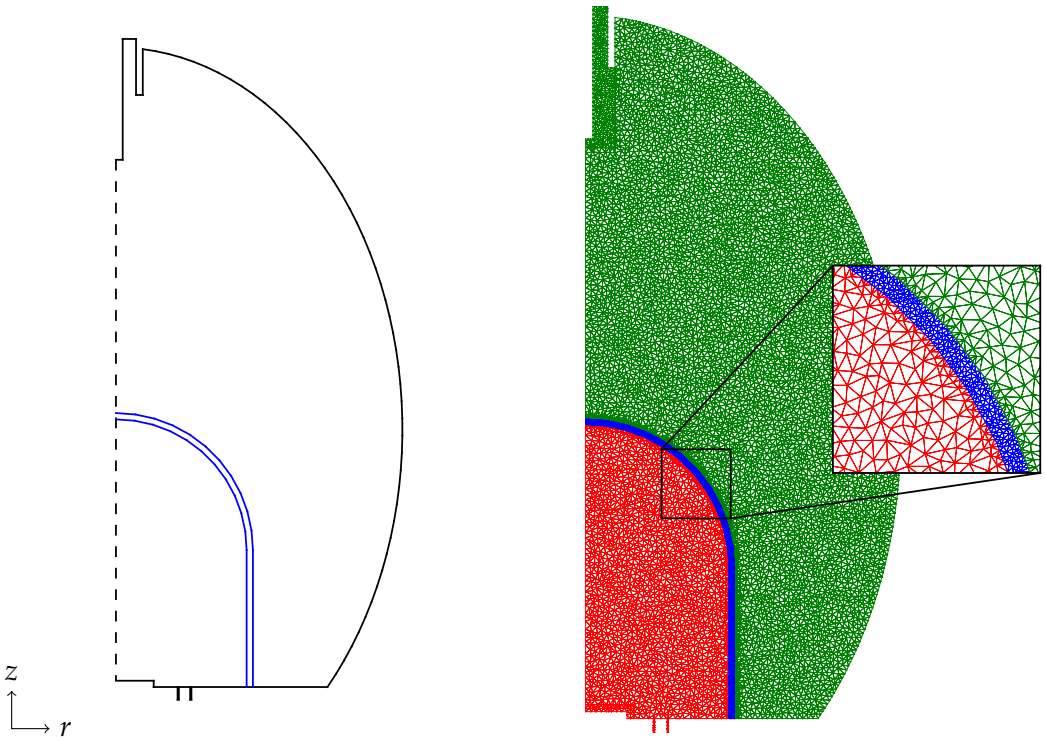


Figure 7.9: Computational domain and generated mesh of the ellipsoidal AIXTRON reactor geometry.

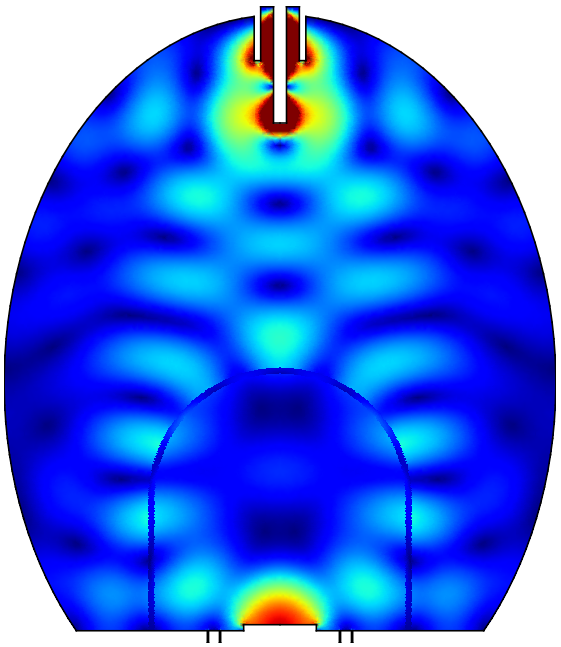


Figure 7.10: Electric field magnitude in the empty AIXTRON reactor.

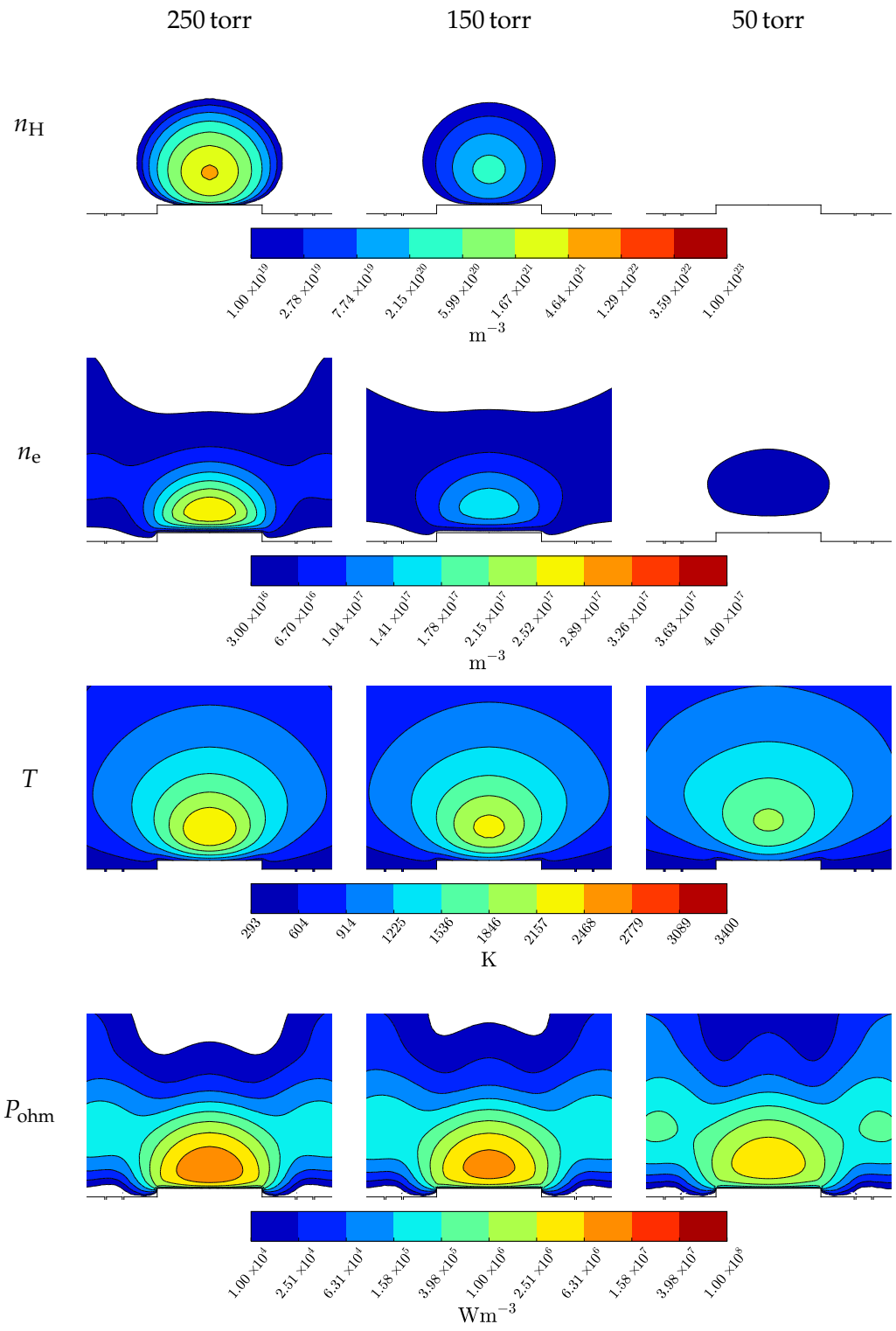


Figure 7.11: The AIXTRON inspired reactor plasma shape operating at 700 W.

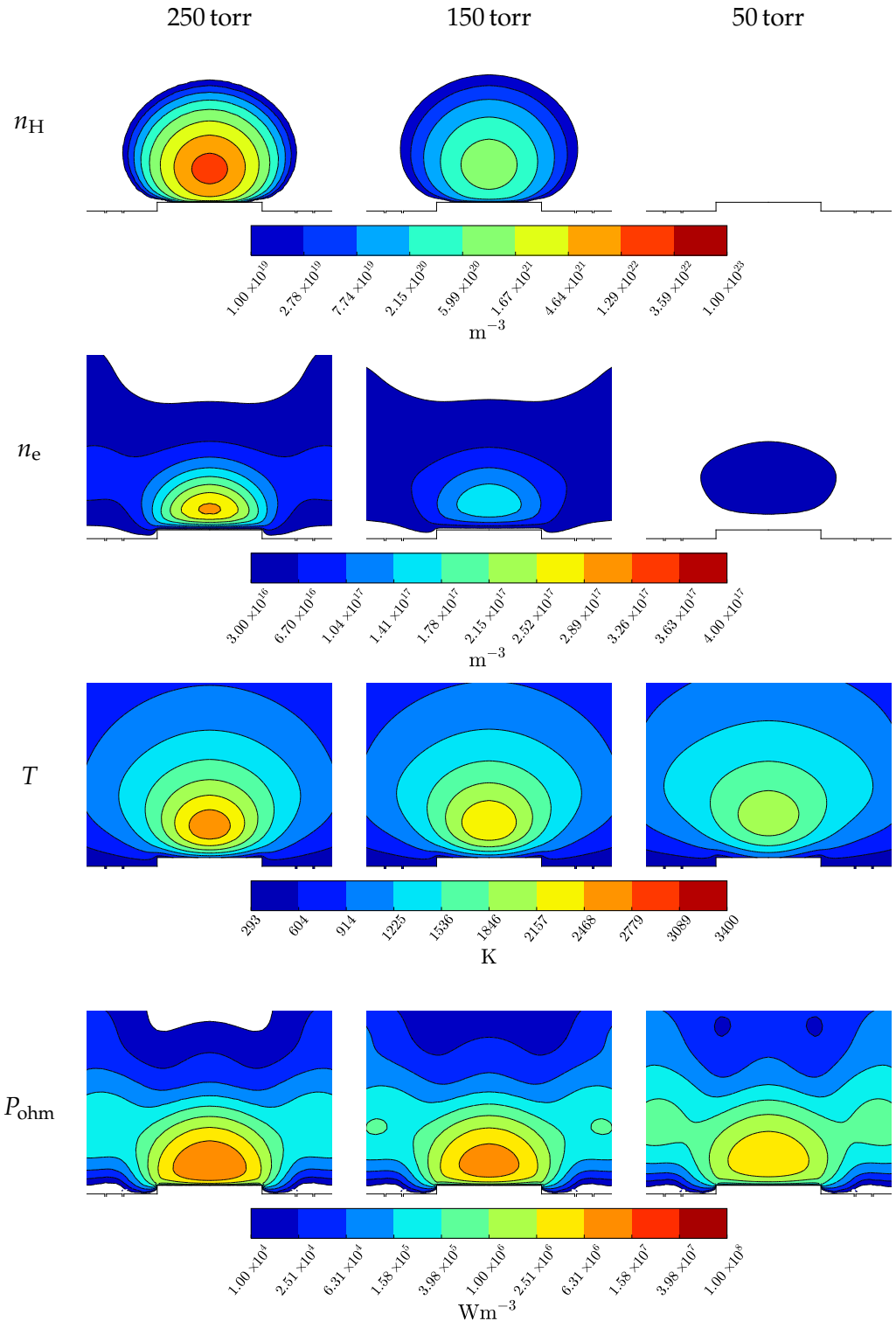


Figure 7.12: The AIXTRON inspired reactor plasma shape operating at 800 W.

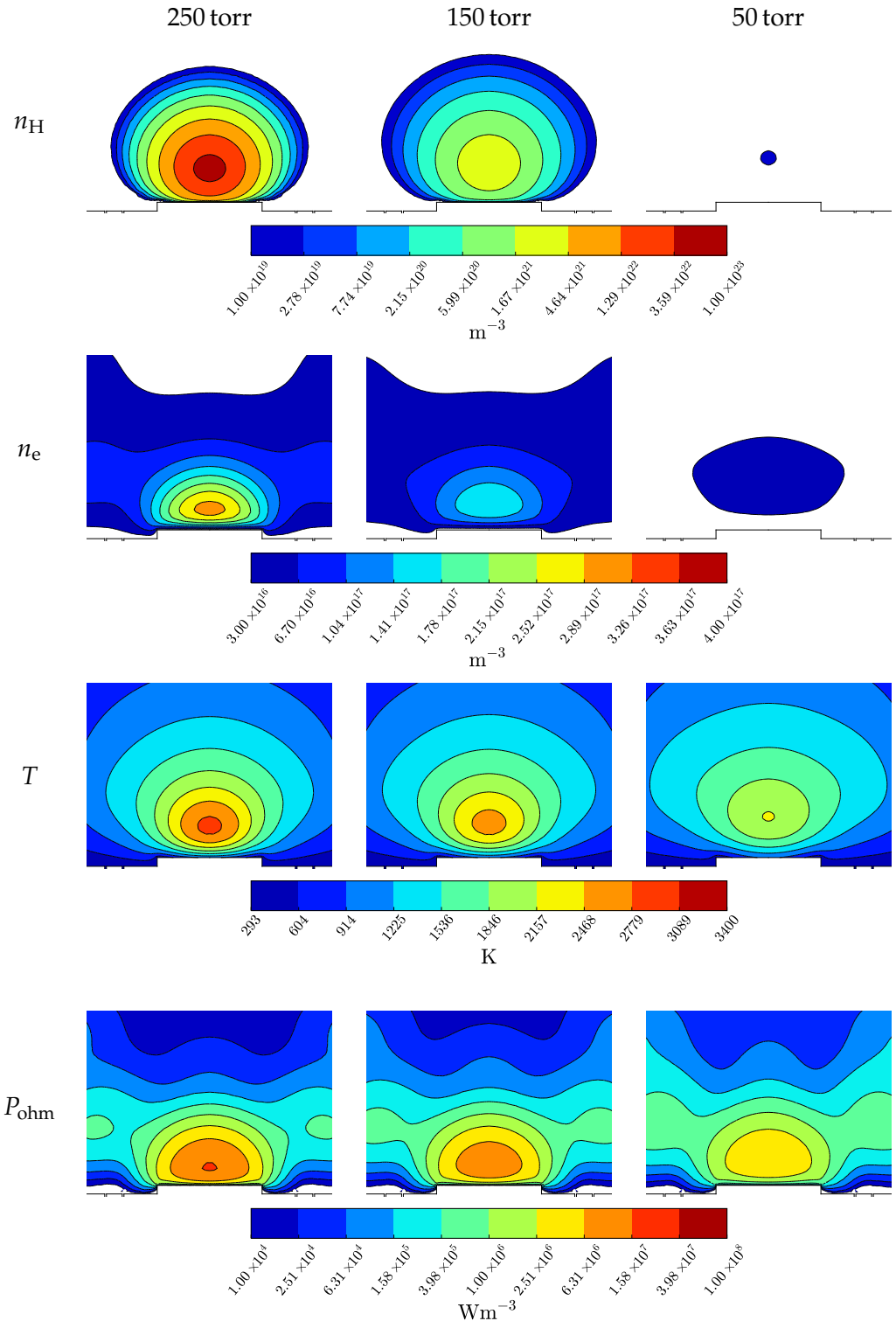


Figure 7.13: The AIXTRON inspired reactor plasma shape operating at 1000 W.

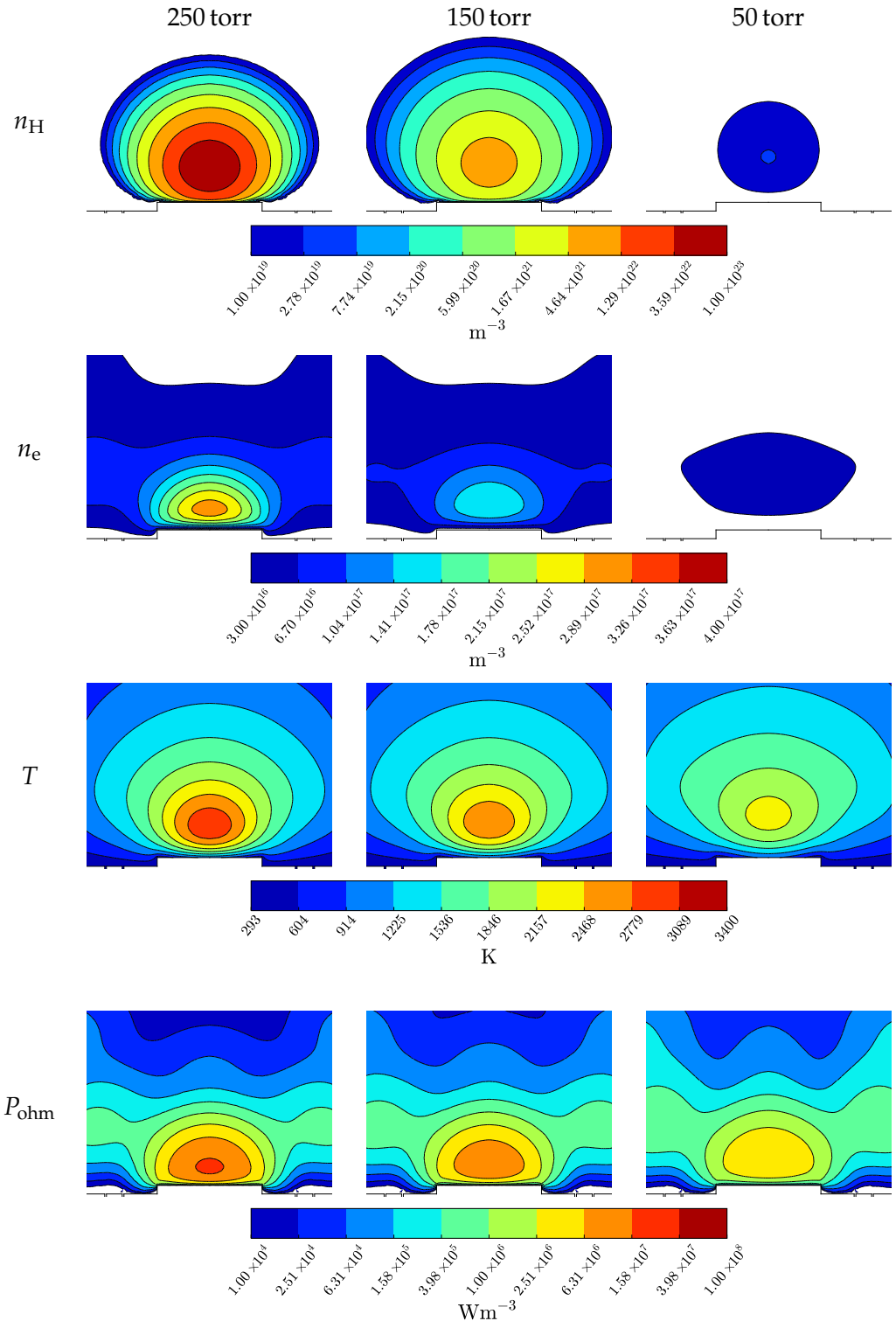


Figure 7.14: The AIXTRON inspired reactor plasma shape operating at 1100 W.

7.3 Example 3: The LIMHP Bell Jar Reactor

In this section, the MPA-CVD model is applied to the geometry shown in Figure 7.15 which approximates the first generation LIMHP bell jar reactor developed by LSPM Paris and Plassys analysed in [127]. In the geometry employed here, the reactor's 26 cm diameter Faraday cage houses a CVD chamber of height 35 cm containing a 10 cm diameter quartz bell jar and 50 mm diameter substrate surface. The reactor is designed to operate at a mean pressure of 20 torr to 120 torr, power of 0.6 kW to 6 kW and microwave frequency of 2.45 GHz exciting the TM_{023} cavity mode.

The mesh generated for the LIMHP domain outline consists of 15 836 elements with 23 432 interior and 644 exterior faces. As usual, regions of interest receive priority for a higher density of elements such as the quartz bell jar, the microwave antenna and the substrate surface. The DG MPA-CVD reactor model on this mesh consists of 467 742 degrees of freedom.

The empty air-filled electric field DG solution in the LIMHP reactor is shown in Figure 7.16. There is a clear peak in the amplitude of the electric field at the substrate surface, however, attention should also be drawn to the peak lying just above the top of the quartz bell jar. The primary peak in the electric field indicates that plasma ignition will occur as desired above the substrate surface, however, increasing the input power in the electric field does lead to a second plasma ball being generated at the top of the quartz bell jar. The first generation LIMHP reactor is known to exhibit this double plasma ball phenomenon [70].

Computed results of the DG MPA-CVD reactor model are presented in Figures 7.17, 7.18, 7.19 and 7.20 with zero inlet gas velocity, $\mathbf{u}_{\text{inlet}} = \mathbf{0}$. These results show agreement with the results presented in [127]. Likewise with the AIXTRON inspired reactor design, the plasma density is distributed more evenly across the substrate surface, as well as its peak residing in closer proximity to the carbon seed. However, as the power input and density increases, the formation of the second plasma ball at the top of quartz bell jar becomes prominent. The occurance of this discharge transition is explained in [63]. Due to both the peak in the electric field at the top of the quartz bell jar and the creeping temperature increase in the same region, a secondary plasma ball is generated. The detrimental effect of this secondary plasma ball is to shield the electric potential at the substrate surface, leading to suboptimal electron densities in the primary plasma.

The attenuation experienced by the electric field is evident from the distribution of the dielectric constant shown in Figure 7.21.

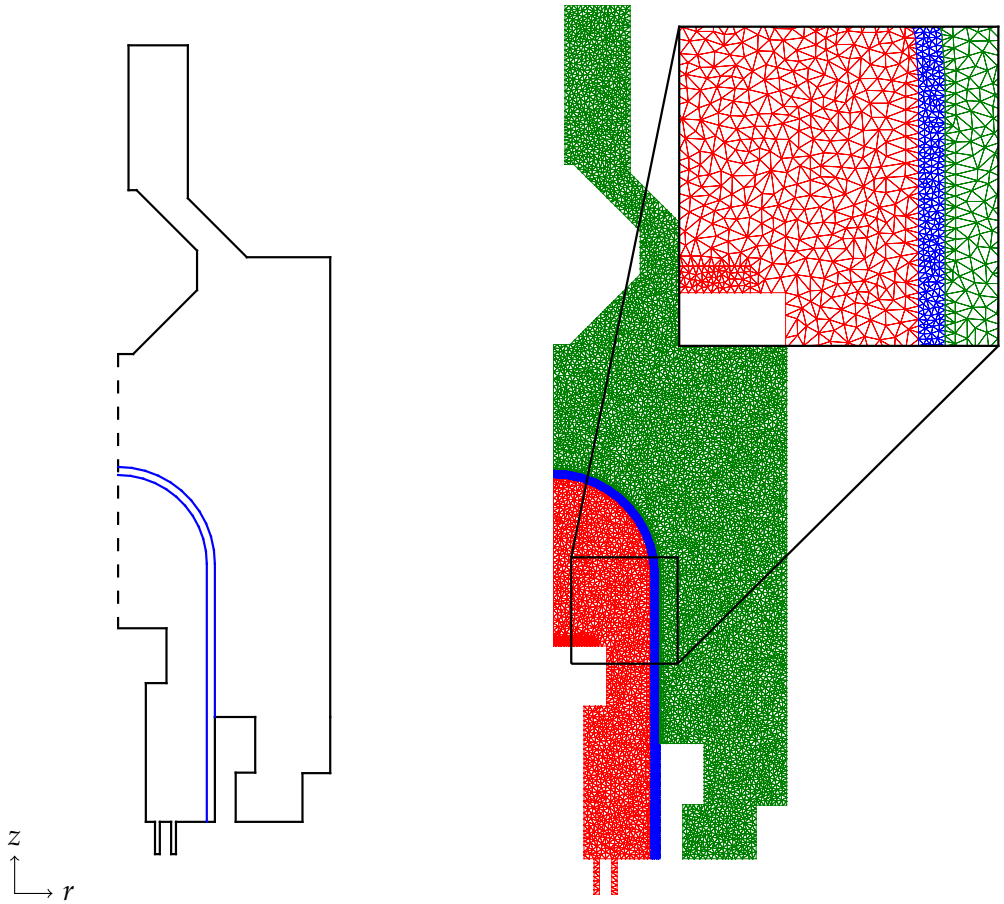


Figure 7.15: Computational domain and generated mesh of the ellipsoidal LIMHP reactor geometry.

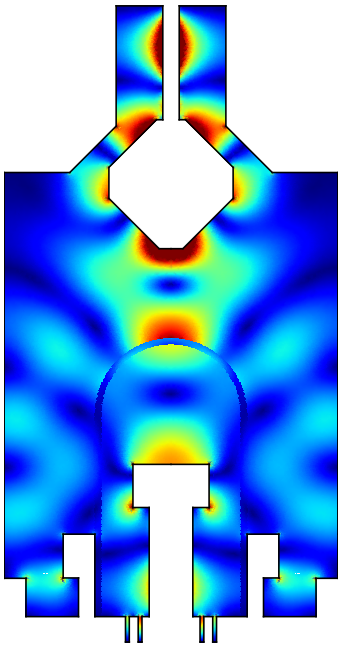


Figure 7.16: Electric field magnitude in the empty LIMHP reactor.

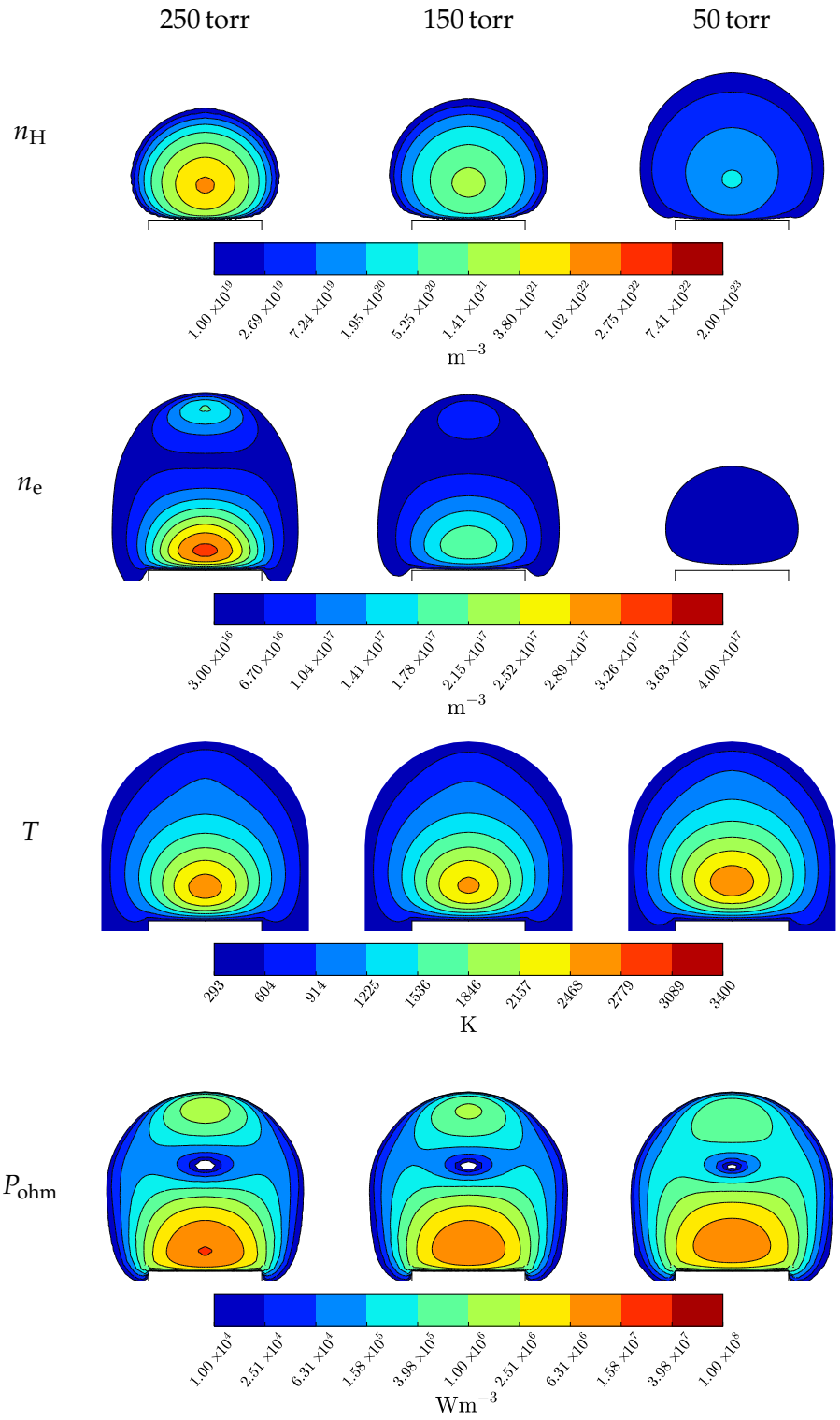


Figure 7.17: The first generation LIMHP inspired reactor plasma shape operating at 700 W.

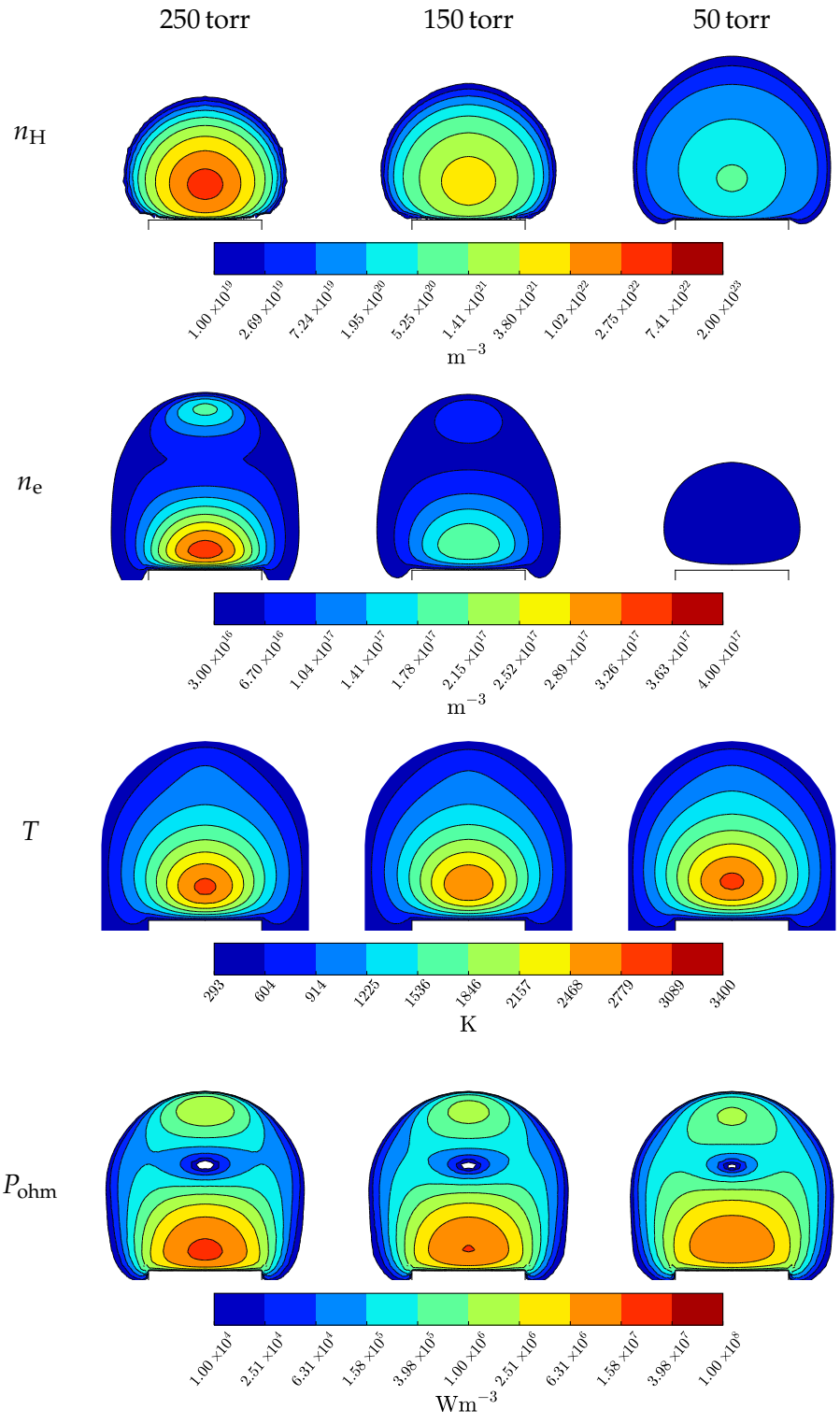


Figure 7.18: The first generation LIMHP inspired reactor plasma shape operating at 900 W.

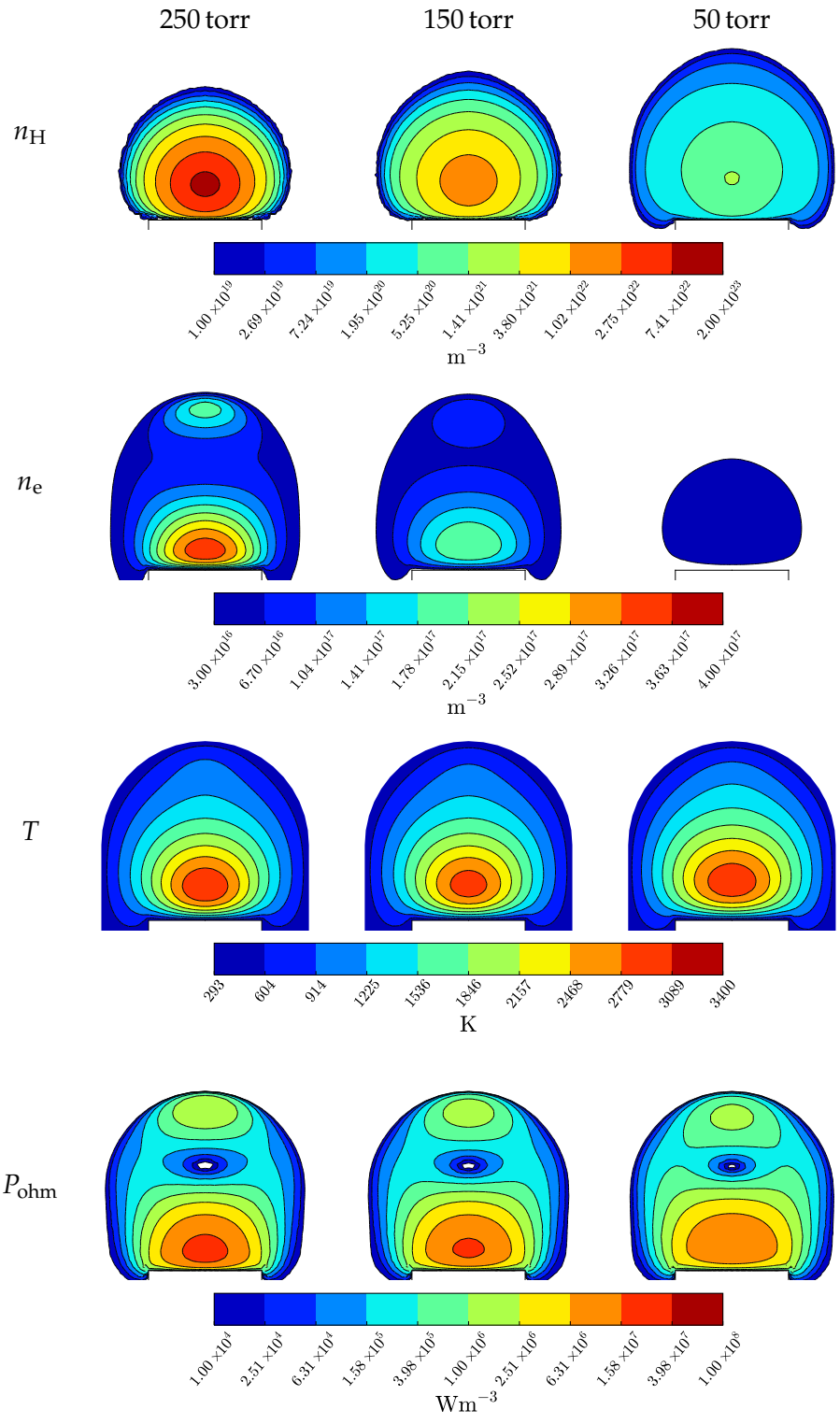


Figure 7.19: The first generation LIMHP inspired reactor plasma shape operating at 1000 W.

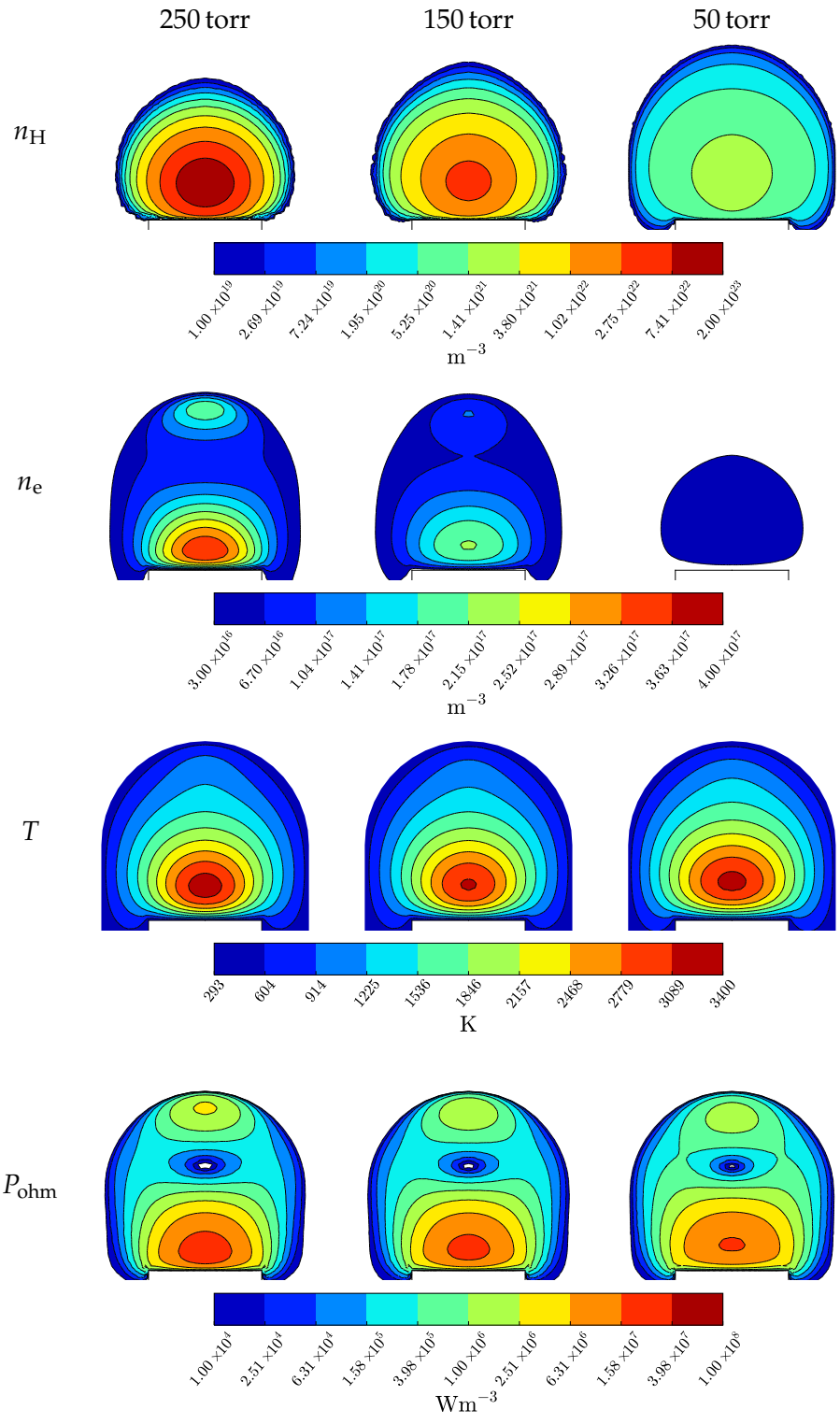


Figure 7.20: The first generation LIMHP inspired reactor plasma shape operating at 1100 W.

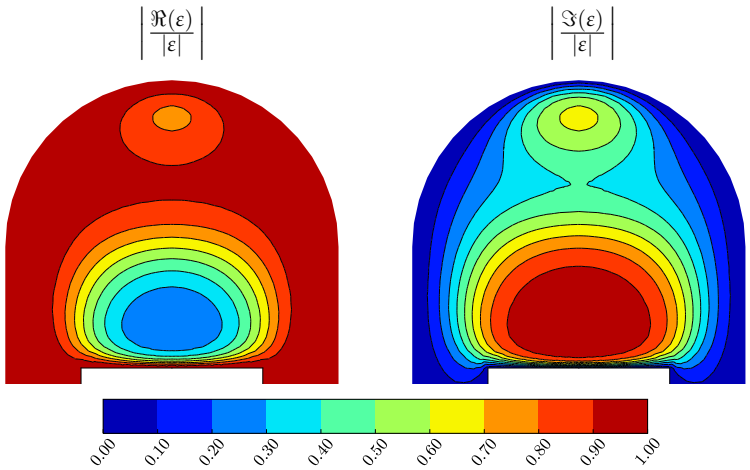


Figure 7.21: The real and imaginary parts of the dielectric constant in the first LIMHP generation reactor inspired design operating at a working pressure of 250 torr and power input of 1100 W. The shielding of the primary plasma is evident from the field attenuation in the secondary plasma ball.

7.4 Optimisation

In this section, we present an example of the optimisation procedure presented in Section 4.4 for reactor design improvement. As a case study, we examine again the LIMHP reactor shown in Section 7.3. The parameter to be optimised is the depth of the microwave antenna in the CVD reactor cavity. The microwave antenna will be displaced over a 40 mm distance between two reference positions. A plot of the starting and ultimate configuration of the microwave antenna position is shown in Figure 7.22.

The MPA-CVD reactor model was solved using meshes similar to that employed in Section 7.3 in terms of element density. We present results of the electric field magnitude DG approximation computed from the MPA-CVD model in Figure 7.23. Here we can see that even small millimetre scale changes in the reactor geometry have a profound effect on the electric field configuration.

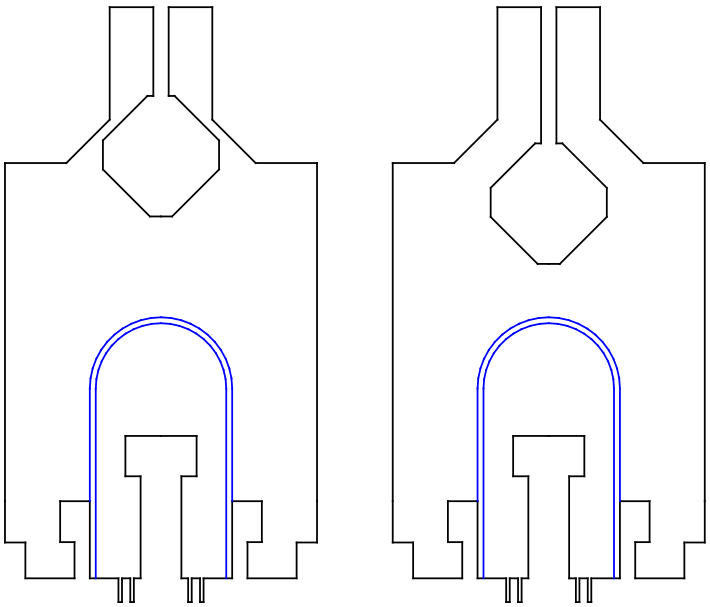
Referring back to the optimisation procedure discussed in Section 4.4, we present example target functions in Figure 7.24. Here we have plot the ratio of the electric field magnitude in the plasma with that in the remaining CVD reactor cavity, and the total power absorbed in the plasma. In Section 4.4 we referred to the work of Füner et al. [55] optimising reactor design based on a target function of the form

$$Q_f(\Phi) = \frac{\|\mathbf{E}\|_{L_2(\Omega_{\text{plasma}})}}{\|\mathbf{E}\|_{L_2(\Omega_{\text{E-field}})}}. \tag{4.4.1}$$

It is evident, however; that this quality factor does not give an indication of the power absorbed in the plasma. Only when large quantities of the microwave power are absorbed in the reactor plasma will the gas temperature rise and hence lead to the dissociation of hydrogen. With the ability to solve the self consistent MPA-CVD reactor model, employing a metric of the power absorbed by the plasma would appear to be the better control of quality factor; thereby, we would seek to optimise

$$Q_f(\Phi) = \int_{\Omega_{\text{plasma}}} P_{\text{ohm}} \, \mathrm{d}\mathbf{x}. \tag{7.4.1}$$

With the self consistent MPA-CVD reactor model we may also optimise the reactor design for the underlying model variables. To this end, we present plots of the maximum values of the temperature, molar mass fraction of atomic hydrogen and the plasma electron density in Figure 7.25. Here we see that the range of configurations which result in the highest temperatures, degree of hydrogen dissociation and plasma electron density is in the range of the greatest power absorbed by the plasma. It should



(a) High Position **(b)** Low Position

Figure 7.22: The start and end configurations of the LIMHP reactor waveguide antenna position range.

be noted that improvements on these metrics should take into consideration the shape and distribution of the plasma.

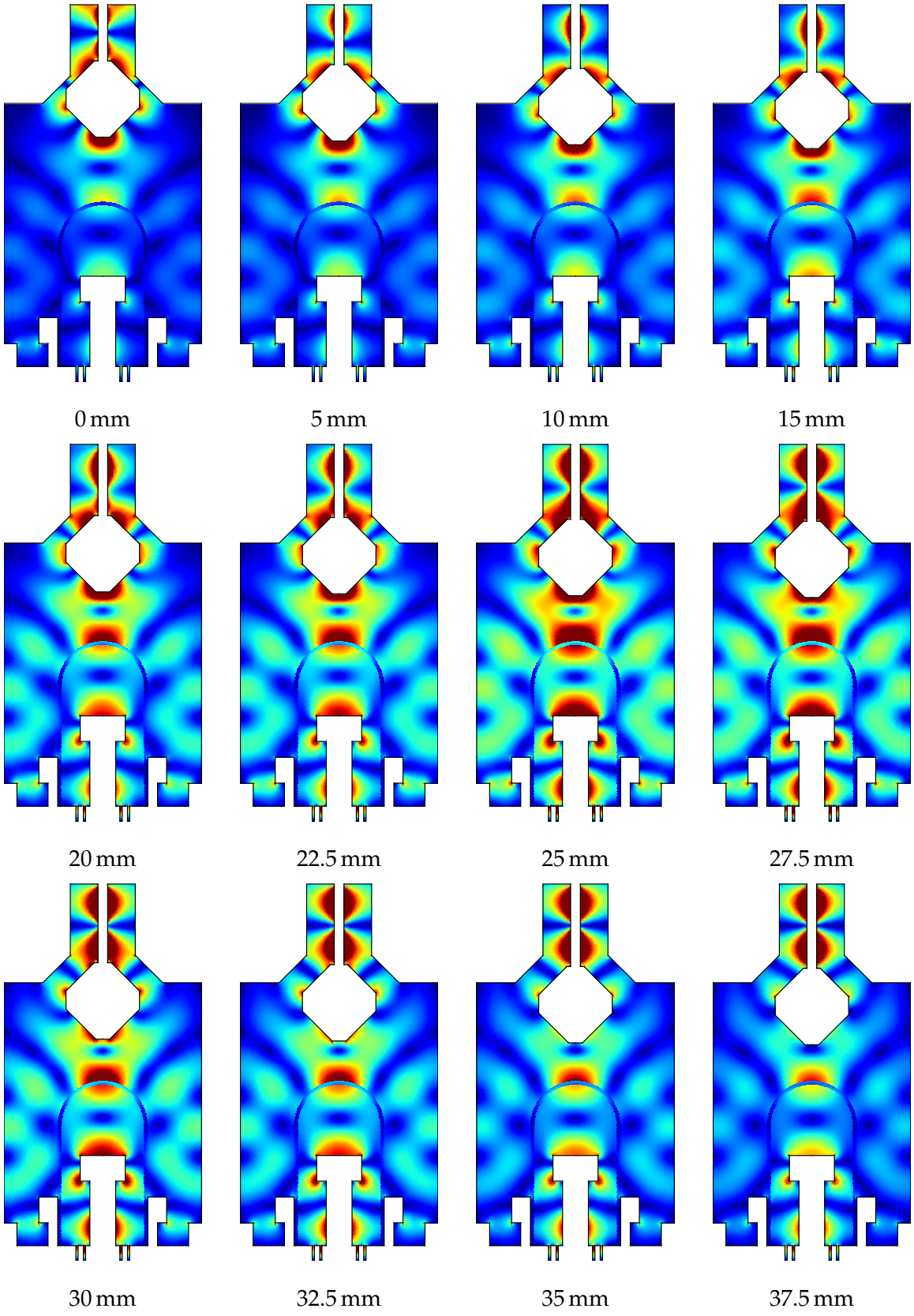
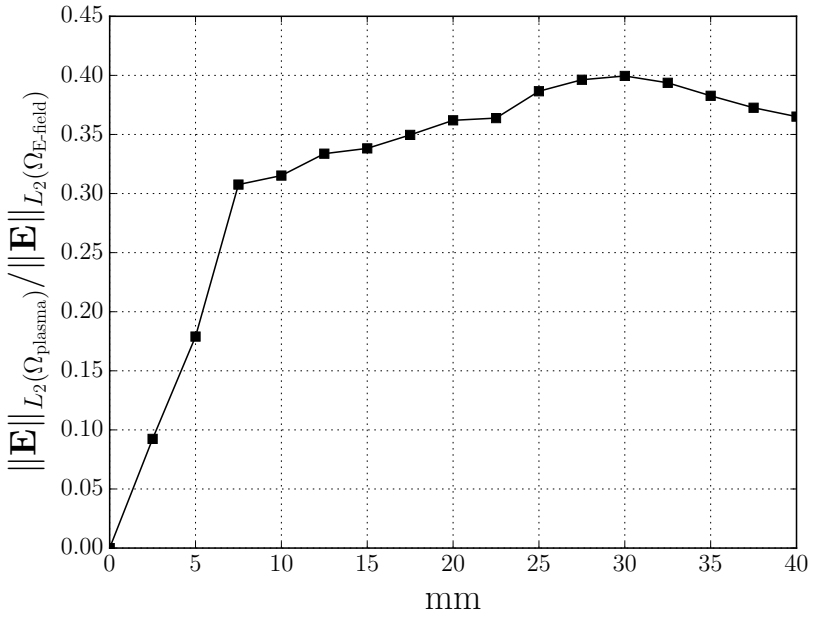
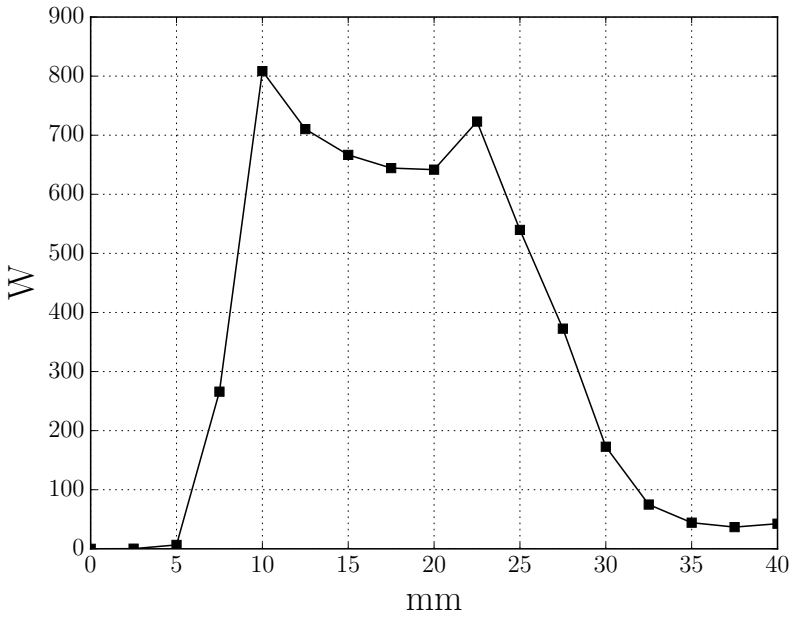


Figure 7.23: Variations in the electric field magnitude in the LIMHP inspired reactor as the position of the microwave antenna is lowered into the cavity.

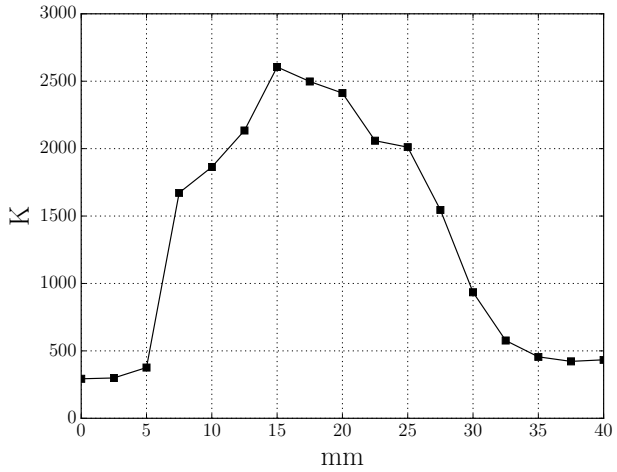


(a) Electric field magnitude ratio in the reactor.

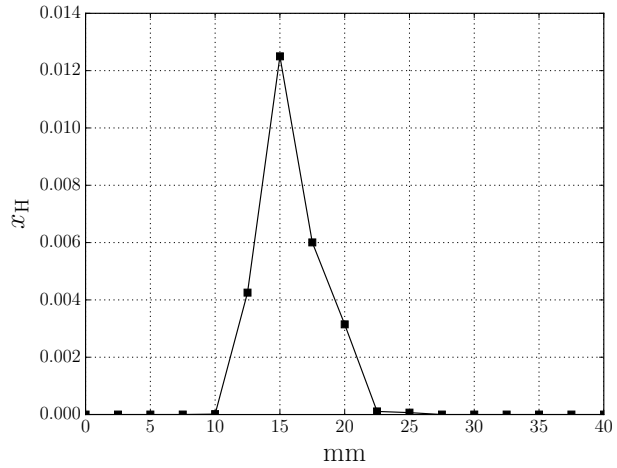


(b) Power absorbed in the plasma.

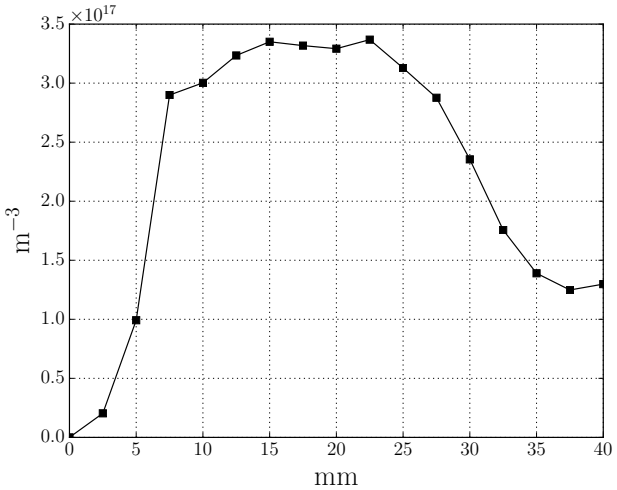
Figure 7.24: Examples of target functions for optimisation. Here the parameter varied is the depth of the microwave antenna from its reference position shown in Figure 7.22. The electric field magnitude ratio in and out of the plasma and the microwave power absorbed in the plasma, $\int_{\Omega_{\text{plasma}}} P_{\text{ohm}} \, \mathrm{d}\mathbf{x}$.



(a) $\max_\Omega T$



(b) $\max_\Omega x_\mathrm{H}$



(c) $\max_\Omega n_\mathrm{e}$

Figure 7.25: Varying the microwave antenna height from the reference position shown in Figure 7.22, here we plot the maximum value of the temperature, atomic hydrogen mass fraction and electron density measured in the numerical simulation.

CHAPTER 8

Conclusions and Future Work

8.1 Summary

In this thesis we have introduced a simplified model for the physical phenomena which occur in an MPA-CVD reactor and its contained hydrogen gas plasma discharge, cf. Chapter 2. The system of simultaneous PDEs underpinning this model provides a great challenge in determining an approximate solution in an efficient and reliable manner. Exploiting the DG numerical scheme, the discretisation of each of these equations follows a rigid procedure for each component of a standard conservation law, including convective and viscous terms. Furthermore, the DG discretisation of the *curl-curl* operator arising in Maxwell's equations allows for the use of the same space of functions for the entire system of equations. These discrete DG formulations have been collected, derived and summarised in Chapter 3.

With the MPA-CVD reactor model and a unified method of numerical discretisation, the full system of equations to be solved, their parameters and boundary conditions for a CVD reactor geometry were then summarised in Chapter 4. Even with this system of equations and discretisation scheme, there still exists the clear barrier of the difficulty of its implementation. We acknowledge the existence of many finite element libraries which facilitate the computation of such systems of finite element formulations, but even with these packages, there is a large volume of work to even code one of the equations of the DG MPA-CVD model. In Chapter 5 we introduce the idea of automatic computation of not only the underlying code of a DG finite element formulation, but indeed the DG finite element formulation itself. With the specification of the convective and viscous terms found in a conversation law, the entire process from DG FEM formulation to computation of its solution is automated. Reducing the effort

required to discretise and solve large systems of PDEs with AptoPy allows for rapid prototyping of modelling ideas, ideal for the underlying system of PDEs found in the MPA-CVD model.

Whilst having a tool such as AptoPy is useful, its utility is measured by its performance and the validity of its results. In Chapter 6 we addressed this issue, presenting a small number of the regression tests written for AptoPy which are most relevant to the MPA-CVD model. We further addressed the issue of the computation time required to parse the DG FEM formulation and automatically generate Fortran code, highlighting that a growing number of simultaneous equations and their interdependence negatively impacts performance. Overall, the one time computation cost of each DG finite element formulation and the required code to compute the DG finite element solution using AptoPy was found to be acceptable.

Equipped with AptoPy and the DG MPA-CVD model we were able to compute numerical solutions for several reactor geometries. In Chapter 7 we presented computations of MPA-CVD simulations for reactor designs inspired by the AIXTRON reactor, the LIMHP reactor, and a simple cylindrical geometry, serving to highlight the transport phenomena of the model. The results computed for the AIXTRON and LIMHP reactors were found to be in agreement with those presented in [127]. In the following section we discuss improvements which should be made in order to more accurately simulate the plasma physics occurring in the MPA-CVD reactor.

8.2 Further Work

8.2.1 Plasma Chemistry

In order to consider a more physically relevant plasma physics model regime, an approach similar to that summarised in [70] should be considered. This involves expanding the binary gas phase model discussed in this thesis to a full multiphase model which includes, at its simplest level, the species of H, H_2 and e. Considering the electrons in the plasma discharge as a separate chemical species allows for a more accurate description of their convective and diffusive mass and energy transport. The treatment of electrons as a component of the multiphase gas also requires a more precise representation of their generation and loss through ionisation, excitation and recombination.

A standard plasma physics model of the ionisation of the gas mixture in a CVD reactor models generation of electrons through ionisation by considering average ionisation collision cross sections between chemical species [58]. For chemical species s we denote its ionisation collision cross section $\sigma_{s,i}$ and electron velocity u_e, such that average collision frequency between electrons and the background gas is given by

$$\nu_{e-s} = n_s \left\langle \sigma_{s,i} u_e \right\rangle . \tag{8.2.1}$$

The ionisation rates of the neutral species in the CVD gas, n, and hence the source of the electrons in the plasma, is proportional to the electron density in the plasma and their collision frequencies

$$G = n_e n_n \left\langle \sigma_{s,i} u_e \right\rangle . \tag{8.2.2}$$

The ionisation frequency of the neutral species in a gas discharge is dependent on the thermal energy of the electrons. As such, the temperature of the electrons T_e which are heated by the deposited microwave power in the CVD reactor should be considered.

Accounting for the electron temperature provides a consistent description of the plasma energy density, a key factor in the plasma's mass and energy transport by convection. The ionisation frequency of each electron impact process with a neutral species can be approximated by rate coefficients

$$k_{s,i}\left(T_e\right) \sim A_{s,i} e^{-E_{s,i}/T_e}, \tag{8.2.3}$$

for ionisation energy $E_{s,i}$ and rate constant $A_{s,i}$. For example, in the case of hydrogen dissociation reaction

$$e + H_2 \longrightarrow e + H + H, \tag{8.2.4}$$

the rate constants can be approximated using $A_{H_2,i} \sim 10^{-14}$ and $E_{H_2,i} \sim 10\,\text{eV}$ [117].

The recombination of an electron with an ion is also a function of the electron-ion collision frequencies, and hence a function of the electron energy. As such, the sink term of the electron density conservation is of the form

$$L = n_e n_i \left\langle \sigma_{rec} u_e \right\rangle . \tag{8.2.5}$$

Developing this further is a key component in a fully self consistent model of the plasma physics in an MPA-CVD reactor. We refer again to the review paper [70] which discusses the multitude of ionisation, recombination and excitation reactions which take place in a Hydrogen and a Hydrogen-Methane plasma.

8.2.2 Automation of the CVD Design Optimisation Procedure

In Section 4.4 we briefly discussed an optimisation procedure for improving the efficiency and design of an MPA-CVD reactor. So far, this process has only been automated given a users' set of parameters Φ. Provided a given reference geometry and operating conditions of the underlying reactor, a simple optimisation algorithm could be considered for designing an optimum reactor targeting the quality factor function Q_f.

8.2.3 3D Numerical Simulation

Although taking advantage of the azimuthal symmetry of most CVD reactor designs in this thesis allowed for computationally less demanding numerical simulations, some properties of the gas flow and CVD geometries cannot be resolved. For example, in the reactor geometries presented in the results in Chapter 7, any gas inlet pipe which is not centred about $r = 0$ is indeed coaxial rather than cylindrical. As such, it is not possible to simulate gas inlet/outlet configurations such as a manifold. Furthermore, some reactor designs also include asymmetrical features such as view ports and measurement instruments [6, 98, 127, 130, 131]. This not only implies that the gas flow is not azimuthally symmetric, but also that the TM resonant mode could be perturbed.

The major challenge of moving to a 3D simulation is the very large system of nonlinear equations in the underlying DG formulation. The approach of using a direct solver implemented in this thesis would be insufficient. Investigating an iterative method and appropriate preconditioning would be necessary. As such, use of a library such as PETSc should be considered [12].

8.2.4 Error Control and Adaptive Refinement

When developing and optimising MPA-CVD reactor designs, there are functionals of interest such as heat flux at the diamond substrate surface. By implementing an *a posteriori* error estimation technique, the meshes used in the DG MPA-CVD simulation can be adaptively refined to reduce the error in these functionals of interest [64, 65, 77, 138]. The DG method is well suited to mesh refinement as it offers a distinct advantage in not requiring that a mesh be conforming, i.e., the function spaces incorporated in solving a DG FEM formulation permit hanging nodes. This could further prove valuable for 3D computationally expensive problems, refining the granularity of the mesh only where

required.

8.2.5 Development of AptoPy

Running a solution process using AptoPy involves running AptoPy and AptoFEM in distinct environments. Specifically, running an AptoPy script generates Fortran code files which are then compiled against the AptoFEM library to generate an executable. This executable then runs the required finite element computation separately from the AptoPy environment.

Connecting the use of the two libraries together in a more natural setting would provide a more seamless interface between the simple expression of a finite element formulation, and its computation. The intention would be for AptoPy to generate the required Fortran code to solve a finite element formulation, as it currently does, whilst also wrapping the functions and data structures of AptoFEM. This would allow seamless communication of data between the two libraries, and hence AptoPy combined with AptoFEM could constitute a numerical library rather than have a single purpose of solving finite element problems. This is an idea which has been exploited by the FEniCS project, whose underlying code is written in C++ and wrapped using the Simplified Wrapper and Interface Generator (SWIG) [103].

Unfortunately there are few libraries as mature as SWIG for wrapping Fortran data structures. The f2py library [113] available as a component of NumPy [111] could facilitate this with its feature set of wrapping Fortran primitives, arrays, function callbacks and modules, however, there is no support for Fortran derived types or class data structures. The modular nature of AptoFEM and its ability to mix element types and function spaces fundamentally relies on the polymorphism paradigm offered by object orientation. The result is that writing the appropriate code to wrap AptoFEM would require a further intermediate layer. This layer would somehow translate the derived type and class data structures of Fortran into primitives and arrays, such that f2py can then be used to wrap those for AptoPy.

APPENDIX A

MPA-CVD Model Equations and Parameters

In this section we summarise the set of simultaneous PDEs which arise from the MPA-CVD reactor model discussed in Chapter 2. We list the unknown system variables in Table A.1, the MPA-CVD model system parameters and constants in Table A.2, the system quantities in Table A.3 and finally the interdependency and nonlinearity of the system of equations in Table A.4.

Navier-Stokes

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) - \nabla\cdot\left(-p\underline{\mathbf{I}} + \eta\left(\nabla\mathbf{u} + \nabla\mathbf{u}^{\top} - \tfrac{2}{3}(\nabla\cdot\mathbf{u})\,\underline{\mathbf{I}}\right)\right) = \rho\mathbf{g} \tag{A.0.1}$$

Continuity

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0 \tag{A.0.2}$$

Conservation of molar mass fraction

$$\frac{\partial(cx_{\mathrm{H}})}{\partial t} + \nabla\cdot(cx_{\mathrm{H}}\mathbf{u}) - \nabla\cdot\left(\frac{M_{\mathrm{H}_2}}{M}c\mathcal{D}_{\mathrm{HH}_2}\nabla x_{\mathrm{H}} + \frac{D_{\mathrm{H}}^{T}}{M_{\mathrm{H}}}\nabla\ln T\right) = R_{\mathrm{H}} \tag{A.0.3}$$

Conservation of energy

$$\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho h\mathbf{u}) - \nabla\cdot(\kappa\nabla T) = P_{\mathrm{ohm}} \tag{A.0.4}$$

Conservation of electron density

$$\frac{\partial n_{\mathrm{e}}}{\partial t} + \nabla\cdot(n_{\mathrm{e}}\mathbf{u}) - \nabla\cdot(D_a\nabla n_{\mathrm{e}}) = G - L \tag{A.0.5}$$

Time harmonic Maxwell equation

$$\nabla\times\left(\mu^{-1}\nabla\times\mathbf{E}\right) + j\omega\left(\sigma_{\mathrm{dc}} + j\omega\varepsilon_p\right)\mathbf{E} = \mathbf{0} \tag{A.0.6}$$

Gauss's law

$$\nabla\cdot\left(\varepsilon_p\mathbf{E}\right) = 0 \tag{A.0.7}$$

Quanitity	Description
$\mathbf{u}$	gas velocity
p	gas relative pressure
x_{H}	molar mass fraction of H
T	gas mixture temperature
n_{e}	electron number density
$\mathbf{E}$	electric field

Table A.1: The unknown system variables.

Constant	Name	Dependence
M_{H}	molar mass of H	
$M_{\mathrm{H_2}}$	molar mass of H_2	
R	gas constant	
P	mean vacuum pressure	
η_{H}	viscosity of H	T
$\eta_{\mathrm{H_2}}$	viscosity of H_2	T
$\mathbf{g}$	gravitational acceleration	
$D_{\mathrm{HH_2}}$	diffusivity of H in a mixture of H and H_2	T
D_{H}^{T}	thermal diffusivity of H	T
k_r	reverse rate coefficient	T
k_f	forward rate coefficient	T
ω	electric field frequency	
m_{e}	electron rest mass	
q_{e}	charge of the electron	
σ	electric conductivity	
μ	electric permeability	
ε_0	electric permittivity of free space	
$\nu_{m_{\mathrm{e}}}$	electron-neutral collision frequency	T
κ_{H}	thermal conductivity of H	T
$\kappa_{\mathrm{H_2}}$	thermal conductivity of H_2	T
h_{H}	specific enthalpy of H	T
$h_{\mathrm{H_2}}$	specific enthalpy of H_2	T
D_a	ambipolar diffusion coefficient	T

Table A.2: MPA-CVD reactor model system constants and parameters. Note that some system constants indicate temperature dependence. These are approximations of empirical data as power series expansions of the temperature variable (cf. [35, 43]).

Quantity	Name	Dependence
$M = (M_{\mathrm{H}} x_{\mathrm{H}} + M_{\mathrm{H}_2}(1 - x_{\mathrm{H}}))$	mean molar mass	x_{H}, T
$\eta =$ equation (2.2.43)	gas viscosity	x_{H}, T
$\rho = {}^{PM}/_{RT}$	gas density	x_{H}, T
$c = {}^{P}/_{RT}$	molar concentration	T
$R_{\mathrm{H}} = 2\left(k_f c^2 (1 - x_{\mathrm{H}}) - k_r c^3 x_{\mathrm{H}}^2\right)$	rate of production of H	x_{H}, T
$\sigma_{\mathrm{dc}} = \frac{{q_{\mathrm{e}}}^2 n_{\mathrm{e}}}{m_{\mathrm{e}} \nu_{m_{\mathrm{e}}}}$	cold plasma conductivity	T, n_{e}
$P_{\mathrm{ohm}} = \frac{1}{2}\lvert\mathbf{E}\rvert^2 \sigma_{\mathrm{dc}} \frac{\nu_{m_{\mathrm{e}}}^2}{\omega^2 + \nu_{m_{\mathrm{e}}}^2}$	power density	$\mathbf{E}$, T, n_{e}
$h = \frac{M_{\mathrm{H}}}{M} x_{\mathrm{H}} h_{\mathrm{H}} + \frac{M_{\mathrm{H}_2}}{M}(1 - x_{\mathrm{H}}) h_{\mathrm{H}_2}$	gas specific enthalpy	x_{H}, T
$\kappa = \frac{1}{2}\left(\sum_{i\in\{\mathrm{H},\mathrm{H}_2\}} x_i \kappa_i + \left(\sum_{i\in\{\mathrm{H},\mathrm{H}_2\}} \frac{x_i}{\kappa_i}\right)^{-1}\right)$	gas thermal conductivity	x_{H}, T
$G = n_{\mathrm{e}} A_0 \lvert\mathbf{E}\rvert^2$	gain of electrons	n_{e}, $\mathbf{E}$
$L = R_0 {n_{\mathrm{e}}}^2$	loss of electrons	n_{e}
$\omega_{\mathrm{pe}}^2 = \frac{{q_{\mathrm{e}}}^2 n_{\mathrm{e}}}{\varepsilon_0 m_{\mathrm{e}}}$	natural plasma frequency	n_{e}
$\varepsilon_p = \varepsilon_0\left(1 - \frac{\omega_{\mathrm{pe}}^2}{\omega(\omega - j\nu_{m_{\mathrm{e}}})}\right)$	cold plasma permittivity	T, n_{e}

Table A.3: MPA-CVD reactor model system quantities. We note the dependence of each quantity on the unknown system variables.

	$\mathbf{u}$	p	x_{H}	T	n_{e}	$\mathbf{E}$	$\mathfrak{p}$
Navier-Stokes (A.0.1)	✓	✓	ρ, η	ρ, η			
Continuity (A.0.2)	✓		ρ	ρ			
Mass fraction (A.0.3)	✓		✓	c, $\mathcal{D}_{\mathrm{HH}_2}$, D_{H}^T, R_{H}			
Energy (A.0.4)	✓		ρ, h, κ	✓	P_{ohm}	P_{ohm}	
Electron density (A.0.5)	✓			D_a	✓	G	
Maxwell (A.0.6)				$\nu_{m_{\mathrm{e}}}$	ε_p	✓	✓
Gauss' law (A.0.7)				$\nu_{m_{\mathrm{e}}}$	ε_p	✓	

Table A.4: Summary of CVD reactor equations and their nonlinear dependencies. We also note here the Lagrange multiplier term, $\mathfrak{p}$, which arises in the DG numerical scheme discussed in Section 3.7.

APPENDIX B

Element Boundary Identities

Theorem 1. *Given the definitions in Section 3.2.3, let* $q \in H^1\left(\mathcal{T}_\Omega^h\right)$ *and* $\mathbf{w} \in \left[H^1\left(\mathcal{T}_\Omega^h\right)\right]^d$, *then*

$$\sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} q\mathbf{w} \cdot \mathbf{n}\, ds = \int_{\Gamma_\mathcal{I} \cup \partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\}\, ds + \int_{\Gamma_\mathcal{I}} \{\!\{q\}\!\} [\![\mathbf{w}]\!]\, ds. \tag{B.0.1}$$

Proof. Initially we note that since $\mathbf{n}^+ = -\mathbf{n}^-$ we may write the identity

$$\frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^+ - \frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^+ = \frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^+ + \frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^- = 0. \tag{B.0.2}$$

We then rewrite the integration over element boundaries in terms of mesh faces

$$\begin{aligned}
\sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} q\mathbf{w} \cdot \mathbf{n}\, \mathrm{d}s &= \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \cap \partial\Omega} q\mathbf{w} \cdot \mathbf{n}\, \mathrm{d}s + \sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa \setminus \partial\Omega} q\mathbf{w} \cdot \mathbf{n}\, \mathrm{d}s \\
&= \int_{\partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\}\, \mathrm{d}s + \int_{\Gamma_\mathcal{I}} \left[q^+ \mathbf{w}^+ \cdot \mathbf{n}^+ + q^- \mathbf{w}^- \cdot \mathbf{n}^- \right]\, \mathrm{d}s \\
&= \int_{\partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\}\, \mathrm{d}s \\
&\quad + \int_{\Gamma_\mathcal{I}} \left[\frac{1}{2} q^+ \mathbf{w}^+ \cdot \mathbf{n}^+ + \frac{1}{2} q^+ \mathbf{w}^+ \cdot \mathbf{n}^+ \right. \\
&\qquad \left. + \frac{1}{2} q^- \mathbf{w}^- \cdot \mathbf{n}^- + \frac{1}{2} q^- \mathbf{w}^- \cdot \mathbf{n}^- \right]\, \mathrm{d}s
\end{aligned}$$

Employing (B.0.2) gives

$$\begin{aligned}
\sum_{\kappa \in \mathcal{T}_\Omega^h} \int_{\partial\kappa} q\mathbf{w} \cdot \mathbf{n}\, \mathrm{d}s &= \int_{\partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\}\, \mathrm{d}s \\
&\quad + \int_{\Gamma_\mathcal{I}} \left[\frac{1}{2} q^+ \mathbf{w}^+ \cdot \mathbf{n}^+ + \frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^+ \right. \\
&\qquad + \frac{1}{2} q^- \mathbf{w}^+ \cdot \mathbf{n}^- + \frac{1}{2} q^- \mathbf{w}^- \cdot \mathbf{n}^- \\
&\qquad + \frac{1}{2} q^+ \mathbf{w}^+ \cdot \mathbf{n}^+ + \frac{1}{2} q^+ \mathbf{w}^- \cdot \mathbf{n}^- \\
&\qquad \left. + \frac{1}{2} q^- \mathbf{w}^+ \cdot \mathbf{n}^+ + \frac{1}{2} q^- \mathbf{w}^- \cdot \mathbf{n}^- \right]\, \mathrm{d}s
\end{aligned}$$

$$
\begin{aligned}
&= \int_{\partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\} \, \mathrm{d}s \\
&\quad + \int_{\Gamma_{\mathcal{I}}} \left[\left(q^{+}\mathbf{n}^{+} + q^{-}\mathbf{n}^{-}\right) \cdot \frac{1}{2}\left(\mathbf{w}^{+} + \mathbf{w}^{-}\right) \right. \\
&\qquad \left. \frac{1}{2}\left(q^{+} + q^{-}\right)\left(\mathbf{w}^{+}\cdot\mathbf{n}^{+} + \mathbf{w}^{-}\cdot\mathbf{n}^{-}\right) \right] \, \mathrm{d}s \\
&= \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} [\![q]\!] \cdot \{\!\{\mathbf{w}\}\!\} \, \mathrm{d}s + \int_{\Gamma_{\mathcal{I}}} \{\!\{q\}\!\} [\![\mathbf{w}]\!] \, \mathrm{d}s. \qquad \text{(B.0.3)}
\end{aligned}
$$

□

Theorem 2. *Given the definitions in Section 3.2.3, let* $\mathbf{z} \in \left[H^1\left(\mathcal{T}_{\Omega}^{h}\right)\right]^{m}$ *and* $\tau \in \left[H^1\left(\mathcal{T}_{\Omega}^{h}\right)\right]^{m\times d}$, *then*

$$
\sum_{\kappa\in\mathcal{T}_{\Omega}^{h}} \int_{\partial\kappa} \tau : (\mathbf{z}\otimes\mathbf{n}_{\kappa}) \; ds = \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} \underline{[\![\mathbf{z}]\!]} : \{\!\{\tau\}\!\} \; ds + \int_{\Gamma_{\mathcal{I}}} \{\!\{\mathbf{z}\}\!\} \cdot [\![\tau]\!] \; ds. \qquad \text{(B.0.4)}
$$

Proof. Noting that

$$
\begin{aligned}
(\mathbf{z}\otimes\mathbf{n}) : \tau &= \sum_{i=1}^{m}\sum_{j=1}^{d} (\mathbf{z}\otimes\mathbf{n})_{ij}\,\tau_{ij} \\
&= \sum_{i=1}^{m}\sum_{j=1}^{d} \left(\mathbf{z}_i\mathbf{n}_j\right)\tau_{ij} \\
&= \sum_{i=1}^{m}\sum_{j=1}^{d} \mathbf{z}_i\left(\tau_{ij}\mathbf{n}_j\right) \\
&= \mathbf{z}\cdot(\tau\mathbf{n}), \qquad \text{(B.0.5)}
\end{aligned}
$$

the proof follows the same procedure as employed in Theorem 1. □

Theorem 3. *Given the definitions in Section 3.2.3, let* $\mathbf{w} \in \left[H^1\left(\mathcal{T}_{\Omega}^{h}\right)\right]^{d}$ *and* $\mathbf{v} \in \left[H^1\left(\mathcal{T}_{\Omega}^{h}\right)\right]^{d}$, *then*

$$
\sum_{\kappa\in\mathcal{T}_{\Omega}^{h}} \int_{\partial\kappa} \mathbf{v}\cdot(\mathbf{n}_{\kappa}\times\mathbf{w}) \; ds = \int_{\Gamma_{\mathcal{I}}\cup\partial\Omega} [\![\mathbf{w}]\!]_{T} \cdot \{\!\{\mathbf{v}\}\!\} \; ds - \int_{\Gamma_{\mathcal{I}}} \{\!\{\mathbf{w}\}\!\} \cdot [\![\mathbf{v}]\!]_{T} \; ds. \qquad \text{(B.0.6)}
$$

Proof. Noting that given the identity $\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}) = \mathbf{c}\cdot(\mathbf{a}\times\mathbf{b})$

$$
\begin{aligned}
\frac{1}{2}\left(\mathbf{n}^{+}\times\mathbf{w}^{+}\right)\cdot\mathbf{v}^{-} - \frac{1}{2}\mathbf{w}^{+}\cdot\left(\mathbf{n}^{-}\times\mathbf{v}^{-}\right) &= \frac{1}{2}\mathbf{v}^{-}\cdot\left(\mathbf{n}^{+}\times\mathbf{w}^{+}\right) + \frac{1}{2}\mathbf{w}^{+}\cdot\left(\mathbf{n}^{+}\times\mathbf{v}^{-}\right) \\
&= \frac{1}{2}\mathbf{v}^{-}\cdot\left(\mathbf{n}^{+}\times\mathbf{w}^{+}\right) - \frac{1}{2}\mathbf{w}^{+}\cdot\left(\mathbf{v}^{-}\times\mathbf{n}^{+}\right) \\
&= 0, \qquad \text{(B.0.7)}
\end{aligned}
$$

the proof follows the same procedure as employed in Theorem 1. □

References

[1] AptoFEM finite element analysis software, 2015. URL http://www.aptofem.com.

[2] M. Alnaes, A. Logg, and K. Ølgaard. Unified Form Language: A domain-specific language for weak formulations of partial differential equations. *ACM Transactions on Mathematical Software*, V(212):1–38, 2012.

[3] P. Amestoy, I. Duff, and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Computer Methods in Applied Mechanics and Engineering*, 184(2-4):501–520, 2000.

[4] P. Amestoy, I. Duff, J.-Y. L'Excellent, and J. Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[5] P. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156, 2006.

[6] K. An, S. W. Yu, X. J. Li, Y. Y. Shen, B. Zhou, G. J. Zhang, and X. P. Liu. Microwave plasma reactor with conical-reflector for diamond deposition. *Vacuum*, 117:112–120, 2015.

[7] D. N. Arnold. An Interior Penalty Finite Element Method with Discontinuous Elements. *SIAM Journal on Numerical Analysis*, 19(4):742–760, 1982.

[8] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

[9] J. Asmussen and D. Reinhard. *Diamond Films Handbook*. Taylor & Francis, 2002.

[10] I. Babuška and T. Strouboulis. *The Finite Element Method and Its Reliability*. Numerical mathematics and scientific computation. Clarendon Press, 2001.

[11] G. A. Baker. Finite element methods for elliptic equations using nonconforming elements. *Mathematics of Computation*, 31(137):45–45, 1977.

[12] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.6, Argonne National Laboratory, 2015.

[13] R. Balmer, J. Brandon, S. Clewes, H. Dhillon, J. Dodson, I. Friel, P. Inglis, T. Madgwick, M. Markham, T. Mollart, N. Perkins, G. Scarsbrook, D. Twitchen, A. Whitehead, J. Wilman, and S. Woollard. Chemical vapour deposition synthetic diamond: materials, technology and applications. *Journal of physics. Condensed matter.*, 21(36):364221, September 2009.

[14] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Transactions on Mathematical Software*, 33(4): 24/1–24/27, 2007.

[15] C. E. Baumann and J. T. Oden. A discontinuous finite element method for convection-diffusion. *Computer Methods in Applied Mechanics and Engineering*, 175: 311–341, 1999.

[16] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18(3):147 – 150, 1984.

[17] K. S. Bey and J. T. Oden. hp-Version discontinuous Galerkin methods for hyperbolic conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 133(3-4):259–286, 1996.

[18] R. B. Bird, W. E. Stewart, and E. N. Lightfoot. *Transport Phenomena*. John Wiley & Sons, 2007.

[19] C. H. Bischof, H. M. Bücker, P. Hovland, U. Naumann, and J. Utke. *Advances in Automatic Differentiation (Lecture Notes in Computational Science and Engineering)*. Springer, 2008.

[20] M. Braack and T. H. Richter. Stabilized finite elements for 3D reactive flows. *International Journal for Numerical Methods in Fluids*, pages 1–6, 2006.

[21] F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Element Methods*. Springer Series in Computational Mathematics, 1991.

[22] F. Brezzi, L. P. Franca, T. J. R. Hughes, and A. Russo. Stabilization technique and subgrid scales capturing. In *The State of the Art in Numerical Analysis, based on the proceedings of the Conference on the State of the Art in Numerical Analysis, York, England, April 1996*, pages 391–406. 1996.

[23] A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, (32):199–259, 1982.

[24] A. Buffa and I. Perugia. Discontinuous Galerkin approximation of the Maxwell eigenproblem. *SIAM Journal on Numerical Analysis*, 44(5):2198–2226, 2006.

[25] A. Buffa, P. Houston, and I. Perugia. Discontinuous Galerkin computation of the Maxwell eigenvalues on simplicial meshes. *Journal of Computational and Applied Mathematics*, 204(2 SPEC. ISS.):317–333, 2007.

[26] R. Bustinza and G. N. Gatica. A local discontinuous Galerkin method for nonlinear diffusion problems with mixed boundary conditions, 2004.

[27] A. Cangiani and E. Süli. Enhanced RFB method. *Numerische Mathematik*, 101(2): 273–308, 2005.

[28] A. Cangiani, E. H. Georgoulis, and M. Jensen. Continuous and Discontinuous Finite Element Methods for Convection-Diffusion Problems : A Comparison. In *Proceedings of the International Conference on Boundary and Interior Layers (BAIL) — Computational and Asymptotic Methods*, pages 1–8, 2006.

[29] M. Capitelli, G. Colonna, K. Hassouni, and A. Gicquel. Electron energy distribution functions in non-equilibrium {H2} discharges. the role of superelastic collisions from electronically excited states. *Chemical Physics Letters*, 228(6):687 – 694, 1994.

[30] M. Capitelli, G. Colonna, a. Gicquel, C. Gorse, K. Hassouni, and S. Longo. Maxwell and non-Maxwell behavior of electron energy distribution function under expanding plasma jet conditions: The role of electron-electron, electron-ion, and superelastic electronic collisions under stationary and time-dependent conditions. *Physical Review E*, 54(2):1843–1849, 1996.

[31] M. Capitelli, R. Celiberto, F. Esposito, A. Laricchiuta, K. Hassouni, and S. Longo. Elementary processes and kinetics of H2 plasmas for different technological. *Plasma Source Science and Technology*, 11:A7–A25, 2002.

[32] M. Capitelli, M. Cacciatore, R. Celiberto, O. Pascale, P. Diomede, F. Esposito, A. Gicquel, C. Gorse, K. Hassouni, A. Laricchiuta, S. Longo, D. Pagano, and M. Rutigliano. Plasmas for Negative Ion Production: Modelling Aspects. *Nuclear Fusion*, 46(6):S260–S274, 2006.

[33] R. Celiberto, U. Lamanna, and M. Capitelli. Dependence of electron-impact dissociative excitation cross sections on the initial vibrational quantum number in H2 and D2 molecules: $X^1\Sigma_g^+ \rightarrow B^1\Sigma_u^+$ and $X^1\Sigma_g^+ \rightarrow C^1\Pi_u$ transitions. *Physical Review A*, 50(6):4778–4785, 1994.

[34] R. Celiberto, M. Capitelli, M. Capitelli, and A. Laricchiuta. Towards a Cross Section Database of Excited Atomic and Molecular Hydrogen. *Physica Scripta*, T96: 32–44, 2002.

[35] M. W. Chase. *NIST-JANAF Thermochemical Tables*. Journal of Physical and Chemical Reference Data, fourth edition, 1998.

[36] B. Cherrington. *Gaseous Electronics and Gas Lasers*. International series in natural philosophy. Pergamon, New York, 1979.

[37] K. A. Cliffe and S. J. Tavener. Implementation of extended systems using symbolic algebra. *Notes on Numerical Fluid Mechanics*, 74:1–15, 2000.

[38] K. A. Cliffe, E. J. C. Hall, P. Houston, E. T. Phipps, and A. G. Salinger. Adaptivity and a Posteriori Error Control for Bifurcation Problems III: Incompressible Fluid Flow in Open Systems with O(2) Symmetry. *Journal of Scientific Computing*, 52(1): 153–179, October 2011.

[39] B. Cockburn, G. Kanschat, D. Schötzau, and C. Schwab. Local discontinuous Galerkin methods for the Stokes system. *SIAM Journal on Numerical Analysis*, 40 (1):319–343, 2002.

[40] B. Cockburn, G. Kanschat, and D. Schötzau. A locally conservative LDG method for the incompressible Navier-Stokes equations. *Mathematics of Computation*, 74 (251):1067–1095, 2005.

[41] B. Cockburn, G. Kanschat, and D. Schötzau. The local discontinuous Galerkin method for linearized incompressible fluid flow: a review. *Computers & Fluids*, 34(4-5):491–506, May 2005.

[42] B. Cockburn, G. Kanschat, and D. Schötzau. A note on discontinuous Galerkin divergence-free solutions of the Navier–Stokes equations. *Journal of Scientific Computing*, pages 1–12, 2007.

[43] J. D. Cox, D. D. Wagman, and V. A. Medvedev. *CODATA key values for thermodynamics*. CODATA series on thermodynamic properties. Hemisphere Pub. Corp., 1989.

[44] M. Crouzeix and P. A. Raviart. Conforming and nonconforming finite element methods for solving the stationary Stokes equations I. *R.A.I.R.O.*, 76(3):33–75, 1973.

[45] B. A. de Dios, F. Brezzi, O. Havle, and L. D. Marini. L2-estimates for the dg iipg-0 scheme. *Numerical Methods for Partial Differential Equations*, 28(5):1440–1465, 2012.

[46] L. Demkowicz and L. Vardapetyan. Modeling of electromagnetic absorption / scattering problems using adaptive finite elements. *Computer Methods in Applied Mechanics and Engineering*, 152(1-2):103–124, 1998.

[47] K. D. Devine, G. L. Hennigan, S. A. Hutchinson, A. G. Salinger, J. N. Shadid, and R. S. Tuminaro. High Performance MP Unstructured Finite Element Simulation of Chemically Reacting Flows. *ACMIEEE SC 1997 Conference SC97*, 1997.

[48] V. Dolejší and O. Havle. The L2-optimality of the IIPG method for odd degrees of polynomial approximation in 1D. *Journal of Scientific Computing*, 42(1):122–143, 2010.

[49] J. Douglas and T. Dupont. *Interior Penalty Procedures for Elliptic and Parabolic Galerkin Methods*, volume 58 of *Lecture Notes in Physics*. Springer Berlin Heidelberg, 1976.

[50] Nada Durić, Iztok Čadež, and Milan Kurepa. Electron impact total ionization cross-sections for methane, ethane and propane. *International Journal of Mass Spectrometry and Ion Processes*, 108(1):R1 – R10, 1991.

[51] M. J. J. Eerden, M. C. M. Van De Sanden, D. K. Otorbaev, and D. C. Schram. Cross section for the mutual neutralization reaction H_2^+ H^-, calculated in a multiple-crossing Landau-Zener approximation. *Physical Review A*, 51(4):3362–3365, 1995.

[52] G. Franz. *Low Pressure Plasmas and Microstructuring Technology*. Springer-Verlag Berlin Heidelberg, 2009.

[53] M. Füner, C. Wild, and P. Koidl. Numerical simulations of microwave plasma reactors for diamond CVD. *Surface and Coatings Technology*, 74-75:221–226, 1995.

[54] M. Füner, C. Wild, and P. Koidl. Novel microwave plasma reactor for diamond synthesis. *Applied Physics Letters*, 72(10):1149–1151, 1998.

[55] M. Füner, C. Wild, and P. Koidl. Simulation and development of optimized microwave plasma reactors for diamond deposition. *Surface and Coatings Technology*, 116-119:853–862, 1999.

[56] A. Gicquel, K. Hassouni, Y. Breton, M. Chenevier, and J. C. Cubertafon. Gas temperature measurements by laser spectroscopic techniques and by optical emission spectroscopy. *Diamond and Related Materials*, 5(3-5):366–372, 1996.

[57] A. Gicquel, M. Chenevier, K. Hassouni, A. Tserepi, and M. Dubus. Validation of actinometry for estimating relative hydrogen atom densities and electron energy evolution in plasma assisted diamond deposition reactors. *Journal of Applied Physics*, 83(12):7504–7521, 1998.

[58] R. J. Goldston and P. H. Rutherford. *Introduction to Plasma Physics*. CRC Press, 1995.

[59] A. M. Gorbachev, V. A. Koldanov, and A. L. Vikharev. Numerical modeling of a microwave plasma CVD reactor. *Diamond and Related Materials*, 10(3-7):342–346, 2001.

[60] A. Granier, C. Boisse-Laporte, P. Leprince, J. Marec, and P. Nghiem. Wave propagation and diagnostics in argon surface-wave discharges up to 100 Torr. *Journal of Physics D: Applied Physics*, 20(2):204–209, 1987.

[61] T. Grotjohn, W. Tan, V. Gopinath, A. Srivastava, and J. Asmussen. Modeling the electromagnetic excitation of a compact ecr ion/free radical source. *Review of Scientific Instruments*, 65(5), 1994.

[62] T. Gudi, N. Nataraj, and A. K. Pani. hp-discontinuous Galerkin methods for strongly nonlinear elliptic boundary value problems. *Numerische Mathematik*, 109 (2):233–268, 2008.

[63] G. Hagelaar, K. Hassouni, and A. Gicquel. Interaction between the electromagnetic fields and the plasma in a microwave plasma reactor. *Journal of Applied Physics*, 96(4):1819–1828, 2004.

[64] R. Hartmann and P. Houston. Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations. *Journal of Computational Physics*, 183(2):508–532, December 2002.

[65] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 24(3):979–1004, 2003.

[66] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible Navier-Stokes equations I: Method formulation. *International Journal of Numerical Analysis and Modeling*, 3(1):1–20, 2005.

[67] K. Hassouni, O. Leroy, S. Farhat, and A. Gicquel. Modeling of H 2 and H 2 / CH4 Moderate-Pressure Microwave Plasma Used for Diamond Deposition 1. *Plasma Chemistry and Plasma Processing*, 18(3):325–362, 1998.

[68] K. Hassouni, A. Gicquel, M. Capitelli, and J. Loureiro. Chemical kinetics and energy transfer in moderate pressure H2 plasmas used in diamond MPACVD processes. *Plasma Sources Science and Technology*, 8(3):494–512, 1999.

[69] K. Hassouni, A. Gicquel, and T. Grotjohn. Self-consistent microwave field and plasma discharge simulations for a moderate pressure hydrogen discharge reactor. *Journal of Applied Physics*, 86(1):134–151, 1999.

[70] K Hassouni, F Silva, and a Gicquel. Modelling of diamond deposition microwave cavity generated plasmas. *Journal of Physics D: Applied Physics*, 43(15):153001, April 2010.

[71] R. M. Hazen. *The Diamond Makers*. Cambridge University Press, 1999.

[72] F. Hecht. New development in freefem++. *Journal of Numerical Mathematics*, 20 (3-4):251–265, 2012.

[73] P. Houston and E. Süli. Stabilised hp-finite element approximation of partial differential equations with nonnegative characteristic form. *Computing*, 66(2): 99–119, 2001.

[74] P. Houston and E. Süli. Stabilised *hp*-finite element approximation of partial differential equations with nonnegative characteristic form. *Computing*, 66(2):99–119, 2001.

[75] P. Houston, C. Schwab, and E. Süli. Stabilized hp-Finite Element Methods for First-Order Hyperbolic Problems. *SIAM Journal on Numerical Analysis*, 37(5): 1618–1643, 1998.

[76] P. Houston, C. Schwab, and E. Süli. Stabilized *hp*-finite element methods for first-order hyperbolic problems. *SIAM Journal of Numerical Analysis*, 37(5):1618–1643, 2000.

[77] P. Houston, C. Schwab, and E. Süli. Discontinuous hp-finite element methods for advection-diffusion-reaction problems. *SIAM Journal on Numerical Analysis*, 39(6):2133–2163, 2002.

[78] P. Houston, Perugia I., and Schötzau D. *hp*-DGFEM for Maxwell's equations. In F. Brezzi, A. Buffa, S. Corsaro, and A. Murli, editors, *Numerical Mathematics and Advanced Applications: Proceedings of ENUMATH 2001 the 4th European Conference on Numerical Mathematics and Advanced Applications Ischia, July 2001*, pages 785–794. Springer, Berlin, 2003.

[79] P. Houston, I. Perugia, and D. Schötzau. Mixed discontinuous Galerkin approximation of the Maxwell operator. *SIAM Journal on Numerical Analysis*, 42(1):434–459, 2004.

[80] P. Houston, I. Perugia, and D. Schötzau. Mixed Discontinuous Galerkin Approximation of the Maxwell Operator: Non-Stabilized Formulation. *Journal of Scientific Computing*, 22-23(1-3):315–346, June 2005.

[81] P. Houston, E. Süli, and T. P. Wihler. A posteriori error analysis of hp-version discontinuous Galerkin finite element methods for second-order quasilinear elliptic problems. *IMA Journal of Numerical Analysis*, 28(2):245–273, 2008.

[82] P. Houston, D. Schötzau, and X. Wei. A Mixed DG Method for Linearized Incompressible Magnetohydrodynamics. *Journal of Scientific Computing*, 40(1-3): 281–314, January 2009.

[83] F. Ihlenburg and I. Babuska. Finite Element Solution of the Helmholtz Equation with High Wave Number Part I: The h-p Version of the FEM, 1995.

[84] F. Ihlenburg and I Babuška. Finite element solution of the Helmholtz equation with high wave number, Part II: The h-p version of the FEM. *SIAM Journal on Numerical Analysis*, 34:315–358, 1997.

[85] R. K. Janev and D. Reiter. Collision processes of CHy and CHy+ hydrocarbons with plasma electrons and protons. *Physics of Plasmas*, 9(9):4071–4081, 2002.

[86] J. Jin. *The Finite Element Method in Electromagnetics*. Wiley-IEEE Press, 3rd edition, 2014.

[87] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Dover Publications, 2009.

[88] M. Kamo, Y. Sato, S. Matsumoto, and N. Setaka. Diamond synthesis from gas phase in microwave plasma. *Journal of Crystal Growth*, 62(3):642 – 644, 1983.

[89] S. D. Kim, Y. H. Lee, and B. C. Shin. Newton's method for the Navier-Stokes equations with finite-element initial guess of Stokes equations. *Computers & Mathematics with Applications*, (51):805–816, 2006.

[90] R. L. Kinder and M. J. Kushner. Consequences of mode structure on plasma properties in electron cyclotron resonance sources. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 17(5):2421–2430, 1999.

[91] R. Koemtzopoulos, D. Economou, and R. Pollard. Hydrogen dissociation in a microwave discharge for diamond deposition. *Diamond and Related Materials*, 2 (1):25–35, 1993.

[92] V. A. Koldanov, A. M. Gorbachev, A. L. Vikharev, and D. B. Radishchev. Self-consistent simulation of pulsed and continuous microwave discharges in hydrogen. *Plasma Physics Reports*, 31(11):965–977, 2005.

[93] N. A. Krall and A. W. Trivelpiece. *Principles of Plasma Physics*. International series in pure and applied physics. McGraw-Hill, 1973.

[94] D. Kröner. *Numerical schemes for conservation laws*. Wiley Teubner, Chichester New York Stuttgart, 1997.

[95] R. Lehoucq, D. Sorensen, and C. Yang. *ARPACK Users' Guide*. Society for Industrial and Applied Mathematics, January 1998.

[96] R. J. LeVeque and H. C. Yee. A Study of Numerical Methods for Hyperbolic Conservation Laws with Stiff Source Terms. *Journal of computational physics*, 86 (1):187 – 210, 1990.

[97] Y. F. Li, J. J. Su, Y. Q. Liu, M. H. Ding, X. L. Li, G. Wang, P. L. Yao, and W. Z. Tang. Design of a new TM021 mode cavity type MPCVD reactor for diamond film deposition. *Diamond and Related Materials*, 44:88–94, 2014.

[98] Y. F. Li, J. J. Su, Y. Q. Liu, M. H. Ding, G. Wang, and W. Z. Tang. A circumferential antenna ellipsoidal cavity type MPCVD reactor developed for diamond film deposition. *Diamond and Related Materials*, 51:24–29, 2015.

[99] S. Y. Liao. *Microwave Devices and Circuits*. Prentice Hall, 1990.

[100] M. A. Lieberman and A. J. Lichtenberg. *Principles of Plasma Discharges and Materials Processing*. Wiley, 2005.

[101] C. Liu, J. Wang, and R. Janev. Mutual neutralization in slow $H_2^+ - H^-$ collisions. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 39(5):1223–1229, 2006.

[102] A. Logg and G. Wells. DOLFIN: Automated finite element computing. *ACM Transactions on Mathematical Software (TOMS)*, V:1–27, 2010.

[103] A. Logg, K. A. Mardal, and G. Wells. *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book (Lecture Notes in Computational Science and Engineering)*. Springer, 2012.

[104] J. Loureiro and C. Ferreira. Electron and vibrational kinetics in the hydrogen positive column. *Journal of Physics D: Applied Physics*, 22(11):1680–1691, 1989.

[105] A. Matveyev and V. Silakov. Kinetic processes in a highly-ionized non-equilibrium hydrogen plasma. *Plasma Sources Science and Technology*, (4):606–617, 1995.

[106] P. Monk. *Finite Element Methods for Maxwell's Equations*. Numerical Mathematics and Scientific Computation. Clarendon Press, 2003.

[107] J. C. Nedelec. Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 35(3):315–341, 1980.

[108] J. C. Nedelec. A new family of mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 50(1):57–81, 1986.

[109] J. Nitsche. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 36(1):9–15, 1971.

[110] K. Ølgaard, A. Logg, and G. Wells. Automated code generation for discontinuous Galerkin methods. *SIAM Journal on Scientific Computing*, 31(2):1–16, 2008.

[111] Oliphant, T. E. Python for Scientific Computing. *Computing in Science & Engineering*, 9(3):10–20, 2007.

[112] D. W. Olson. Diamond, industrial. *U.S. Geological Survey Minerals Yearbook*, 2012.

[113] Pearu, P. F2PY: a tool for connecting Fortran and Python programs. *International Journal of Computational Science and Engineering*, 4(4):296–305, 2009.

[114] I. Perugia and D. Schötzau. The hp-local discontinuous Galerkin method for low-frequency time-harmonic Maxwell's equaitons. *Mathematics of Computation*, 72: 1179–1214, 2003.

[115] I. Perugia, D. Schötzau, and P. Monk. Stabilized interior penalty methods for the time-harmonic Maxwell equations. *Computer Methods in Applied Mechanics and Engineering*, 191(41-42):4675–4697, 2002.

[116] A. V. Phelps. Collisions of H^+, H_2^+, H_3^+, ArH^+, H^-, H, and H_2 with Ar and of Ar^+ and ArH^+ with H_2 for Energies from 0.1 eV to 10 keV. *Journal of Physical and Chemical Reference Data*, 21(4):883–897, 1992.

[117] M. A. Prelas, G. Popovici, and L. K. Bigelow. *Handbook of Industrial Diamonds and Diamond Films*. Taylor & Francis, 1997.

[118] F. Read. Extensions of the Wannier theory for near-threshold excitation and ionisation of atoms by electron impact. *Journal of Physics B: Atomic and Molecular Physics*, 17(19):3965–3986, 1984.

[119] D. Reiter and R. K. Janev. Hydrocarbon Collision Cross Sections for Magnetic Fusion: The Methane, Ethane and Propane Families. *Contributions to Plasma Physics*, 50(10):986–1013, 2010.

[120] B. Rivière, M. F. Wheeler, and V. Girault. Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems. Part I. *Computational Geosciences*, 3:337–360, 1999.

[121] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.

[122] A. B. Sá, C. M. Ferreira, S. Pasquiers, C. Boisse-Laporte, P. Leprince, and J. Marec. Self-consistent modeling of surface wave produced discharges at low pressures. *Journal of Applied Physics*, 70(8):4147–4158, 1991.

[123] A. G. Salinger, R. P. Pawlowski, J. N. Shadid, and B. G. van Bloemen Waanders. Computational Analysis and Optimization of a Chemical Vapor Deposition Reactor with Large-Scale Computing. *Industrial & Engineering Chemistry Research*, 43(16):4612–4623, August 2004.

[124] J. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[125] I. Shinmamura and T. Fujimoto. State densities and ionization equilibrium of atoms in dense plasmas. *Physical Review A*, 42(4):2346–2353, 1990.

[126] F. Silva, a. Michau, X. Bonnin, a. Gicquel, N. Boutroy, N. Chomel, and L. Desoutter. Electromagnetic modelling of a microwave cavity used for the deposit of

amorphous carbon films on the inner wall of PET bottles. *Diamond and Related Materials*, 16(4-7 SPEC. ISS.):1278–1281, 2007.

[127] F. Silva, K. Hassouni, X. Bonnin, and A. Gicquel. Microwave engineering of plasma-assisted CVD reactors for diamond deposition. *Journal of physics. Condensed matter*, 21(36):364202, 2009.

[128] H. Smyth, H. Skogen, and W. Harsell. Thermal capacity of vitreous silica. *Journal of the American Ceramic Society*, 36(10):327–328, 1953.

[129] K. Spear and J. Dismukes, editors. *Synthetic Diamond: Emerging CVD Science and Technology*. Wiley-Interscience, 1994.

[130] J. J. Su, Y. F. Li, M. H. Ding, X. L. Li, Y. Q. Liu, G. Wang, and W. Z. Tang. A dome-shaped cavity type microwave plasma chemical vapor deposition reactor for diamond films deposition. *Vacuum*, 107:51–55, 2014.

[131] J. J. Su, Y. F. Li, X. L. Li, P. L. Yao, Y. Q. Liu, M. H. Ding, and W. Z. Tang. A novel microwave plasma reactor with a unique structure for chemical vapor deposition of diamond films. *Diamond and Related Materials*, 42:28–32, 2014.

[132] S. Sun and M. Wheeler. Discontinuous Gaerlkin methods for coupled for an reactive transport problems. *Applied Numerical Mathematics*, 52:273–298, 2005.

[133] SymPy Development Team. *SymPy: Python library for symbolic mathematics*, 2014. URL http://www.sympy.org.

[134] T. Tabata and T. Shirai. Analytic Cross Sections for Collisions of H^+, H_2^+, H_3^+, H, H_2, and H^- With Hydrogen Molecules. *Atomic Data and Nuclear Data Tables*, 76 (1):1–25, 2000.

[135] W. Tan and T. Grotjohn. Modeling the electromagnetic excitation of a microwave cavity plasma reactor. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 12(4):1216, 1994.

[136] W. Tan and T. Grotjohn. Modelling the electromagnetic field and plasma discharge in a microwave plasma diamond deposition reactor. *Diamond and Related Materials*, 4(9):1145–1154, 1995.

[137] C. Taylor and P. Hood. A numerical solution of the Navier-Stokes equations using the finite element technique. *Computers and Fluids*, (1):73–100, 1973.

[138] L. Vardapetyan and L. Demkowicz. Hp-Adaptive Finite Elements in Electromagnetics. *Computer Methods in Applied Mechanics and Engineering*, 169(3-4):331–344, 1999.

[139] G. Vijayasundaram. Transonic flow simulations using an upstream centered scheme of Godunov in finite elements. *Journal of Computational Physics*, 63:416–433, 1986.

[140] A. L. Vikharev, A. M. Gorbachev, V. A. Koldanov, R. A. Akhmedzhanov, D. B. Radishchev, T. A. Grotjohn, S. Zuo, and J. Asmussen. Comparison of pulsed and CW regimes of MPACVD reactor operation. *Diamond and Related Materials*, 12 (3-7):272–276, 2003.

[141] L. Vriens and A. Smeets. Cross-section and rate formulas for electron-impact ionization, excitation, deexcitation, and total depopulation of excited atoms. *Physical Review A*, 22(3):940–951, 1980.

[142] H. G. Weller and G. Tabor. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, 12(6):620–631, 1998.

[143] M. F. Wheeler. An Elliptic Collocation-Finite Element Method with Interior Penalties. *SIAM Journal on Numerical Analysis*, 15(1):152–161, 1978.

[144] C. Wilke. A viscosity equation for gas mixtures. *The Journal of Chemical Physics*, 18(4):517–519, 1950.

[145] H. Yamada, a. Chayahara, Y. Mokuno, Y. Soda, Y. Horino, and N. Fujimori. Modeling and numerical analyses of microwave plasmas for optimizations of a reactor design and its operating conditions. *Diamond and Related Materials*, 14(11-12): 1776–1779, 2005.

[146] K. S. Yee and J. S. Chen. The finite-difference time-domain (FDTD) and the finite-volume time-domain (FVTD) methods in solving Maxwell's equations. *IEEE Transactions on Antennas and Propagation*, 45(3):354–363, 1997.