University of Leicester

A Discontinuous Galerkin Finite Element Method with Turbulence Modelling for Incompressible Flows

Thesis submitted for the degree of

Doctor of Philosophy

at the University of Leicester

by

Luke Jolley

Department of Mathematics

University of Leicester

July 2019

Abstract

A Discontinuous Galerkin Finite Element Method with Turbulence Modelling for Incompressible Flows

Luke Jolley

This thesis explores the use of an innovative interior penalty Discontinuous Galerkin Finite Element Method (DGFEM) for the Reynolds averaged, incompressible Navier-Stokes equations, coupled with the $k-\omega$ turbulence model. The simulation of incompressible flows is relatively inexpensive computationally, and, with appropriate assumptions, provides a good approximation to compressible flows. This makes them useful for large simulations, such as those required by the steam turbine industry.

Current generation industrial CFD solvers require *ad hoc* user intervention with regards to solution refinement, in order to achieve numerical results with a sufficient degree of accuracy. Accurate simulations of curved blade geometries rely on a dense packing of straight edged elements in order to represent the geometry correctly. This results in extended simulation times and non-optimised numerical results.

Curved boundary elements allow highly curved geometries to be represented by fewer mesh elements, enabling effective mesh refinement perpendicular to the boundary, without increasing mesh density parallel to the boundary. To achieve this, we propose a novel approach using inverse estimates to derive a new discontinuity-penalisation function which stabilises the DGFEM for computations in both two and three dimensions, on meshes consisting of standard shaped elements with general polynomial faces. Automated solution refinement is achieved by considering the dual-weighted-residual approach, defining a suitable numerical approximation for the dual solution, along with a target functional to drive the refinement. A novel continuation and refinement algorithm, along with a prototype DGFEM solver is developed, producing a number of interesting numerical results for high Reynolds number flows.

These ideas are extended to incorporate the recent results in the literature for DGFEMs on general computational meshes consisting of polygonal elements. For high Reynolds number turbulent flows, we show that polygonal elements can be used to significantly reduce mesh density and the computational resources required for fluid simulations through several numerical experiments.

Acknowledgements

I would like to take this opportunity to acknowledge the endless support of my supervisors, who have helped me in so many ways throughout my study. Manolis Georgoulis, for always replying to my emails with advice and guidance. Whether it was the weekend, New Year's Day, or 1 o'clock in the morning, you always made yourself available. You have taught me so much, and pushed me further academically than I ever thought possible. Thank you to Aldo Rona, who tutored me in fluid dynamics and introduced me to the wonderful people at GE. My thanks to Edward Hall, who, without his help, I know I would still be looking for coding bugs to this day. Thank you for the countless supervisor meetings and unending chats, you made my time in Leicester so enjoyable.

I would also like to show my gratitude for the funding provided by the University of Leicester, the EPSRC and General Electric.

Special thanks to my parents, who were always at the end of the telephone to support me any way they could, especially through the stressful writing up period. You have afforded me so many opportunities in life, I would not be where I am today without you.

And to D'arcy, you kept me sane and made Leicester home. Thank you for always being there.

Contents

Nomenclature

Re Reynolds number.

L_C Characteristic linear dimension.

l_m Mixing length. (m)

c Chord length. (m)

α Angle-of-attack. (°)

s_s Stator pitch. (m)

μ Molecular viscosity. (kg/ms)

μ_t Turbulent viscosity. (m^2/s)

ν Kinematic viscosity. (m^2/s)

ρ Density. (kg/m^3)

$\mathbf{u}$ Solution vector.

u_i Solution velocity components. $i = 1, ..., d$. (m/s)

k Turbulent kenetic energy. (m^2/s^2)

ω Dissipation per unit turbulence kinetic energy. (1/s)

p Static pressure. (pa)

u_∞ Freestream velocity. (m/s)

p_∞ Freestream static pressure. (pa)

p_t Total pressure. (pa)

q Dynamic pressure. (pa)

Chapter 1

Introduction

The use of mathematics to model physical systems is one of the most important and interesting applications of the subject. Indeed, whilst being a matter of academic fascination for mathematicians, it is of special interest to those associated with the field of engineering, especially those in the industrial sector. The modelling of physical phenomena often leads to ordinary or partial differential equations (PDEs), equipped with the relevant boundary and initial conditions to mimic the phenomena of interest. However, finding solutions to these equations is usually rather complicated, since it is uncommon for an analytical solution to these systems to be known. Therefore, scientists are required to focus their efforts on numerical approximations and, whilst the advent of computing clusters and ever more powerful hardware has allowed them to tackle problems that would have been considered intractable even a few years ago. Improvements to the underlying methods should also not be neglected. Indeed, algorithmic improvements have the potential to provide the biggest leap in computational accuracy, whilst reducing overall computation time. The reduction of computational cost is a driving concern for the industrial sector, with cluster operating costs, in particular the price of electricity, directly impacting company profits. For instance, the power consumption of the ARCHER system operated by the EPSRC at the University of Edinburgh is 3306 kW [8, 132]. In one hour, the ARCHER cluster will use almost as much electricity as the average UK home consumed throughout 2017 [133].

General Electric (GE) [93], along with support and funding from the EPSRC, proposed new research into finding a more efficient replacement for current generation numerical solvers. In particular, they are concerned with fluid simulations for turbomachinery, aiding in the development of next generation steam turbines for use in power generation. Within this setting, a balance must be struck between computational accuracy and the cost of the calculation; noting that, from a business point of view, it is often more important to have

an approximation to the flow, rather than a set of PDEs that cannot be solved for a particular geometry within reasonable time constraints.

The current industry standard for fluid simulations are the long established Finite Volume Methods (FVM), due to their ability to resolve problems on geometrically complex domains whilst handling the types of non-self-adjoint PDEs associated with fluid mechanics. For an in-depth overview of FVMs, we refer the reader to [60, 105, 111, 134] and the references cited therein. Whilst the exact algorithms implemented within these industrial solvers are typically closely guarded secrets, they can be considered as implementations of the ideas presented in [111, 134] with added efficiency improvements, such as for specific equations [62], or particular boundary conditions [48]. However, it is known that these methods rely heavily on user-defined meshes in order to accurately capture flow features through often complex geometries, with engineers required to manually adjust the mesh density to improve the accuracy of the simulation. This is incredibly time-consuming and can often lead to areas of the mesh which have been unnecessarily, over-, or under-refined. Additionally, these FVM-based numerical solvers require a dense packing of straight-edge mesh elements around curves in order to preserve the underlying geometry [134], increasing the memory requirements of the computing cluster. This is made worse when pressure corrected methods are employed, requiring the storage of an offset mesh to handle the pressure calculations [105].

The inefficiencies mentioned above are some of the key areas that GE wishes to address through the development of a replacement numerical solver for their day-to-day fluid simulations. They have suggested that a DGFEM may prove to be a viable alternative to FVMs, with a special interest in automated solution refinement for both accuracy and efficiency improvements. Representatives from GE have stated that the majority of their fluid flow modelling only requires steady-state (time independent) simulations that utilise turbulence models coupled with the Reynolds-averaged Navier-Stokes equations (RANS). As such, we develop a numerical method suitable for these purposes, addressing the dense packing of straight edge elements through the implementation of curved boundaries, as well as algorithms for automated solution refinement suitable for turbomachinery flows.

1.1 Discontinuous Galerkin Finite Element Methods

Finite Element Methods (FEMs) are an attractive alternative to FVMs, offering more options for solution refinement and mesh design [126]. The construction of both FVMs and FEMs begins by discretising the domain into a collection of elements (volumes in the FVM) and then proceeds to find an approximate solution for the PDE across the mesh. FVMs associate a volume averaged value of the solution with each element; then numerical fluxes

must be introduced in order to ensure information is passed correctly between elements. On the other hand, FEMs typically represent the solution using piecewise continuous polynomials across each element, allowing for a potential increase in solution accuracy by altering the polynomial degree on an element by element basis, without the need to increase the mesh density.

Unfortunately, continuous FEMs have limited applicability to problems arising in fluid dynamics, suffering from difficulties in constructing stable discretisations of the advection terms for general nonlinear PDEs [86, 111, 147]. The lack of stability presents itself in the form of node to node oscillations when the advection terms become dominant. Attempts have been made to remedy this, through the addition of artificial diffusion terms [110, 58], or by the use of intractably fine meshes. As a remedy, we consider instead the class of FEMs known as discontinuous Galerkin FEMs (DGFEM).

The first DGFEM was introduced in 1973 by Reed and Hill [117] with the aim of computing solutions to the neutron transport problem, a first order hyperbolic equation. Analysis of this new method was carried out by LeSaint and Raviart [104] utilising Fourier techniques, and then later by Johnson and Pitkäranta [95]. Independently, around the same time Babuška [18] developed a penalty method to find numerical solutions to second-order elliptic problems; however it would not be until later that the name discontinuous Galerkin became associated with his work. These techniques were originally known as penalty methods, introducing terms to penalise the numerical solution for having discontinuities or jumps across element boundaries. Babuška's method was based on on the work of Nitsche [108], who suggested the weak imposition of boundary conditions for elliptic problems within the realms of the classical finite element method. However, Babuška's penalty method suffered consistency issues and it was not until the papers of Baker [20] and Arnold [10] that these problems were fully resolved. In recent years, there has been a whole host of DGFEMs developed, for a variety of problems. For a more in depth look at the developmental history of these methods, see Arnold, Brezzi, Cockburn and Marini [11, 12], and Cockburn, Karniadakis, and Shu [47]. In particular, this work focuses on the symmetric interior penalty DGFEM for the discretisation of the elliptic terms, which offers a good balance between implementation flexibility and conservation properties [82, 83].

In its simplest form, a DGFEM can be considered as a fusion of the classical FEM and the FVM. For DGFEMs, the global continuity requirement is relaxed such that cross-element continuity is only weakly imposed, opting instead for the use of numerical fluxes in order to pass information between elements. However, rather than using volume averaged values to represent the approximate solution, we proceed as in the continuous FEM, selecting (high order) polynomials on each element instead. The flexibility of DGFEMs to accommodate discontinuities in the numerical solution allows for the avoidance, or at

least severely dampens, the non-physical oscillations typically associated with continuous numerical approximations of boundary layer problems, without the need for a separate stabilisation method.

Furthermore, DGFEMs are able to exploit the usual mesh, or h-refinement, normally associated with FEMs easily, without the need to take any special steps to resolve hanging nodes. A hanging node is created when the vertex of one element lies on the edge or face of a neighbouring element, a particularly common occurrence in automatic mesh refinement. DGFEMs also offer the opportunity to adjust the polynomial degree of the numerical solution in the same way continuous FEMs do, but without requiring special techniques to enforce inter-element continuity. This is known as p-refinement, and it increases the accuracy of the numerical solution without altering the mesh size, resulting in a potentially exponential decrease in the error of the approximate solution as the number of degrees of freedom N, increases; i.e. $O\left(e^{-\beta N}\right)$, $\beta > 0$ a constant depending on the regularity of the solution. Whereas h-refinement achieves only an algebraic error reduction rate, i.e., $O(N^{-\alpha})$, $\alpha > 0$, at most.

1.2 Discontinuous Galerkin Finite Element Methods for Turbomachinery Applications

Typically, fluid flows around turbomachinery components are turbulent and of an extremely high Reynolds number, in the region of and above 1×10^6 [93]. When simulating these type of flows, the set of nonlinear equations that describe the motion of the fluid require a vast number of computational resources to compute their approximate numerical solutions. As such, we seek to reduce the hardware requirements of these simulations by considering steady-state, turbulent incompressible flows. To simulate the flow, we construct a nonlinear system consisting of two conservation equations, mass and momentum, coupled with the turbulence model equations. The alternative approach, is to consider the compressible Navier-Stokes equations, which require an additional equation of state, along with a further conservation of energy formulation, dramatically increasing the computational requirements of any proposed numerical solver. Normally, the incompressible Navier-Stokes equations are employed for hydrodynamic problems, but when coupled with a suitable turbulence model and under the appropriate assumptions, they provide a good approximation for compressible flows in industrial applications [43]. In general, turbomachinery flows tend to consist of large areas of almost laminar flow requiring very few mesh elements to accurately capture the average flow behaviour, as well as turbulent boundary layers and turbulent recirculation areas. The latter require a much denser packing of mesh

elements to provide usable numerical results. As such, the proposed DGFEM requires an algorithm to effectively control the mesh density and error values of the numerical solution.

In this work, we show that DGFEMs are well suited to problems arising in turbomachinery design, able to utilise existing meshing techniques such as blocking, as well as unstructured grids generated through algorithms like the Delaunay triangulation [113]. Traditionally, FVMs use highly optimised structured grids with refinement blocks in areas where the underlying geometry is complex, such as blade edges. These improve the overall computational efficiency of the calculation at the expense of automated solution refinement, unable to accommodate mesh refinement/coarsening. The hanging nodes that are generated by automated refinement strategies require special adjustments to the FVM data structures. These are organised for structured grids such that volume neighbour data is intrinsic to the order of the matrix. However, this approach is unsuitable for both unstructured grids and those that use automated refinement strategies. On the other hand, DGFEMs are able to handle these hanging nodes automatically, without any special treatment or considerations. The interior penalty DGFEM is selected as the underlying method for the proposed numerical solver, benefiting from a reduced stencil size compared to the Bassi-Rebay DGFEM [24, 66], with solutions on a specific element only dependent on the information from neighbouring elements, rather than the neighbours of neighbours. The reduced requirements of the interior penalty DGFEM complement the decision to focus on turbulent incompressible flows, ensuring that the proposed simulation tool is highly efficient. As we show, high Reynolds number, turbulent incompressible flow problems are extremely stiff, and, as such, require careful manipulations of the stiffness matrix to solve in a reasonable time frame. To combat this, DGFEMs can be easily parallelised, enabling the use of multiple processing cores to improve simulation time.

Interior penalty implementations of DGFEMs commonly rely on mapping equations from a defined reference element to the physical mesh elements. The solution on these elements is then penalised according to the transformation that maps the reference to the physical element. Manipulation of this penalisation term has been explored most notably for anisotropic elements [77, 67]. However, through a more considered approach, we demonstrate that this concept can be extended to also include curved elements in a highly efficient manner. By implementing this new penalty parameter in a prototype numerical solver for turbulent incompressible flows, we show that we are able to capture the underlying geometry of the problem, and in turn the numerical solution more accurately on coarse meshes.

Most recently, DGFEMs have been considered for general polytopic meshes (i.e. polygonal in two dimensions and polyhedral in three dimensions) [36, 37, 38]. This methodology is an alternative approach to the usual formulation of interior penalty methods, removing the need for a reference element, defining instead, all quantities on the physical elements.

Furthermore, this approach has demonstrated a reduction in computational cost compared to standard elements [38], whilst maintaining a level of precision acceptable for engineering applications. However, this approach is yet to be extended to high Reynolds number turbulent flows, and has the potential to provide a much more effective solution to the mesh coarsening requirement for almost laminar flow areas compared to using standard elements alone.

The current approach to solution refinement in industrial turbomachinery simulations is one that requires ad-hoc user intervention. In order to improve the accuracy of a particular numerical solution for a specified geometry, scientists typically manually adjust the mesh density after each progressive solve in a continuation type approach, until a desired density or numerical accuracy is met [93]. This process requires submitting multiple jobs to a high-performance computing (HPC) cluster for calculations to be performed, with potential down-time if jobs finish between working hours. This further extends the time necessary to complete a single simulation on a specified geometry. DGFEMs are able to utilise the automated solution refinement strategies, much like FEMs, by considering *a posteriori* error analysis. That is to say, we are able to approximate the solution error using the numerical solution alone, without knowing any information about the analytical solution. This is crucial in the case of the incompressible Navier-Stokes equations where the general solution is unknown. In particular, we manipulate the so called *a posteriori* error bound into its elementwise contributions. These are then used to drive an automated mesh modification algorithm by refining the elements that contribute the largest values to the error bound. In this work, we consider a specialised version of this known as the dual-weighted-residual (DWR) approach [30, 31, 59]. In DWR, a target quantity is chosen, and the numerical solution is refined in order to minimise the error associated with this quantity. The key difference here is that the numerical solution is not refined to the solution of the PDE system, meaning that we do not need to derive an *a posteriori* error bound for our equations. In the literature, DWR technique for DGFEMs has been considered, with high levels of success, for both laminar incompressible [46] and compressible flows [80, 84]. Furthermore, this approach has also been shown to be effective for meshes consisting of polytopic elements [38]. As such, in this work, we extend these ideas considerably to include high Reynolds number turbulent flows.

This work presents the following advancements to the ideas already discussed in the literature:

- a DGFEM for the incompressible Navier-Stokes equations with $k - \omega$ turbulence modelling;
- a novel implementation of curvilinear mesh elements for the interior penalty DGFEM;

- an adaptive DWR mesh refinement and continuation algorithm for high Reynolds number, turbulent, incompressible flows on curvilinear meshes;
- an adaptive DWR mesh refinement and continuation algorithm for high Reynolds number, turbulent, incompressible flows on polytopic meshes;
- the design and development of a prototype numerical solver for curvilinear and polytopic meshes to implement the ideas presented in this work, along with supporting numerical test cases.

1.3 Research Outline

This work begins by discussing interior penalty DGFEMs in finer detail. We consider problems of mixed typed, specifically the advection-diffusion equation, introducing and exploring key concepts such as Sobolev spaces, finite element spaces and trace operators. This introductory problem occupies Chapter 2, and serve to highlight many important considerations that must be accounted for when developing DGFEMs for the incompressible Navier-Stokes equations, especially when they are coupled with a turbulence model.

We then progress to introduce laminar flows in Chapter 3, providing a DGFEM discretisation of the incompressible Navier-Stokes equations. Laminar flows are the simplest type of flows, characterised by a series of fluid layers moving in parallel directions. Whilst typically only low Reynolds number flows are laminar, high Reynolds number flows feature almost-laminar sections in the free stream, so being able to calculate solutions to these problems is extremely important. Secondly, this prototype solver is used as a basis upon which turbulence models are incorporated, as well as capabilities for high Reynolds number flows. To compute numerical solutions to these problems, we need to explore the conditions for which the method is stable using the, so called, inf-sup condition, first proposed by Babuška [17] and Brezzi [35]. These equations are nonlinear and require the addition of a damped Newton Solver to calculate the numerical solution. As such, preconditioning methods for linear systems that arise are discussed in Section 3.3, as it is important that the proposed solvers can handle the stiffness of the final problem, especially when we introduce the turbulence model in Chapter 4. Finally, some simple two-dimensional test cases are considered in order to demonstrate the effectiveness of the prototype solver.

In Chapter 4 we extend the previous discussion to include turbulence models, focusing in particular on the $k-\omega$ model, due to its use in current generation industrial software. The $k-\omega$ model has the added benefit of being able to be applied throughout the turbulent boundary layer, without any special modifications, unlike those required by, e.g. the

$k-\epsilon$ model. Turbulence modelling provides an added layer of difficulty since it increases stiffness of the PDE system, reducing the rate at which both nonlinear and linear solvers converge. In order to overcome this challenge, we introduce continuation methods, solving a low Reynolds number flow on the desired geometry, before slowly increasing the Reynolds number for successive solves on the same mesh. We assume that if the increase in Reynolds number is sufficiently small, then the new solution should resemble the previous solution, with special consideration for the transition from laminar to fully turbulent flows. In particular, these lower Reynolds number solutions act as an initial estimate for the nonlinear solver, improving the likelihood that the solver converges to a solution. We conclude the chapter with some validation results for the new DGFEM, considering a two-dimensional turbulent flow through a channel domain, comparing the numerical results to those from the ODE solver presented in Wilcox [145]. This is significant, as this is the first interior penalty DGFEM solver for turbulent incompressible flows, and demonstrates the potential of DGFEM as a replacement for FVMs in industrial numerical solvers.

Chapter 5 is concerned with the development of a new interior penalty DGFEM capable of handling mesh elements with curved boundaries (curvilinear elements). In particular, we analyse inverse estimates of anisotropic elements to deduce a better approximation for the minimal value of the interior penalty parameter, capable of providing stability to a DGFEM that is stable for meshes consisting of both standard and curvilinear elements. It is important to note that both types of elements are required to accurately capture the intricate features of boundary layers in high Reynolds number turbulent flows for complicated geometries. Further numerical examples are considered in order to provide validation and analysis of the new parameter with respect to existing isotropic and anisotropic parameters. The choice of discontinuity-penalisation function is crucial for the behaviour of the method; too large and it will converge to the solution at a suboptimal rate, too small, the method will be unstable and fail to converge altogether. We conclude this chapter with a number of numerical experiments that demonstrate the effectiveness of the proposed penalty parameter for meshes containing standard elements with general polynomial faces.

Following the initial validation of the proposed DGFEM, we discuss the development of an automated mesh refinement strategy in Chapter 6. Then, following [80], we discuss the goal-oriented, dual-weighted-residual (DWR) a posteriori error estimation technique, applying it to high Reynolds number, turbulent incompressible flows. In particular, we derive a weighted a posteriori error bound, a numerical approximation to the dual solution, and discuss suitable target functionals for turbomachinery applications, such as profile drag. We also consider the use of mesh refinement coupled with continuation techniques, discussing the advantages of refining the solution at lower Reynolds numbers in order to improve the converge rate of the nonlinear solver for high Reynolds numbers. Then, we move to discuss different types of mesh refinement before validating the results with further

numerical experiments. In the particular scenario we consider, we show that adaptivity reduces significantly the error per numerical degree of freedom and that this efficiency can be extended to general polygonal meshes as well.

Chapter 7 discusses the advantages associated with implementing polygonal meshes within the interior penalty DGFEM setting for high Reynolds number, turbulent, incompressible flows. We consider the different approaches that exist within the literature for the definition of polygonal elements, as well as methods to generate them. Furthermore, we also consider the relationship between standard triangular/quadrilateral elements and their exotic polygonal counterparts, deriving and comparing inverse estimates on these different element types in order to calculate the required discontinuity-penalisation parameter. The discussion then moves to consider the difficulties of implementing an interior penalty DGFEM on a general computational mesh, as well as the approaches in the literature to overcome these challenges. These difficulties include defining suitable basis functions for polygonal elements, along with suitably accurate quadrature rules for element integration. Chapter 7 concludes with some final numerical experiments, as well as an implementation of an interior penalty DGFEM for high Reynolds number, turbulent incompressible flows on a general computational mesh consisting of polygonal elements, with an adaptive mesh refinement strategy based upon the DWR a posteriori error indicator derived in Chapter 6.

Lastly, in Chapter 8 we draw the final conclusions from this work, discussing the real world applicability of the proposed DGFEM for current and future industrial applications. We discuss the effectiveness, as a prototype for industrial steam turbine applications, of the functioning numerical solver we have developed, along with the associated algorithms we have created. We consider some possible avenues of further research, proposing a number of natural extensions to the work presented in this volume, including the implementation of mixing planes for industrial meshes, as well as the applicability of time-dependent models such as Large Eddy Simulation for industrial type problems.

Chapter 2

Discontinuous Galerkin Methods for Problems of Mixed Type

We consider the construction of DGFEMs in detail, using the general second-order linear PDEs of mixed type as a model problem. For further information see [38, 130], as well as the references cited therein. We begin by introducing the general class of second-order PDEs with nonnegative characteristic form [91, 109]. The specific PDE considered here is more commonly referred to as the steady-state advection-diffusion-reaction equation, and serves to model natural fluid systems. In particular, it models how physical quantities are transferred within a physical system, using a combination of the two processes; advection and diffusion. These types of PDEs are referred to as mixed since they may exhibit characteristics of either hyperbolic, parabolic, or elliptic PDEs, depending on the PDE coefficients locally in different parts of the problem's domain. This formalism allows us to explore key concepts in the construction of DGFEMs, in particular, the construction of finite element spaces and operators, along with the existence and uniqueness of the approximate solution. It is instructive to consider the detail of these components before we study more complicated problems in subsequent chapters. Moreover, this class of equations and the respective DGFEMs form the basis of constructing the numerical approximation of the turbulent model equations, which is a particular focus of this work.

2.1 PDEs with Nonnegative Characteristic Form

PDEs of nonnegative characteristic form are a class of equation that encompasses a large variety of PDEs, including second-order elliptic and parabolic PDEs, ultra-parabolic equations, first-order hyperbolic problems, the Kolmogorov-Fokker-Planck equations of Brownian motion [23], the equations of boundary layer theory in hydrodynamics, and numerous

other degenerate equations. Of particular interest is the advection-diffusion-reaction equation; one of the simplest models of a natural fluid system. The advection term references the movement of substances or particles within a surrounding fluid such as air or water, whilst the diffusion terms describe the movement of particles from an area of high concentration to areas of lower concentration.

We define the problem on a bounded open Lipschitz domain in $\mathbb{R}^d$, $d = 2, 3, 4$, denoted by Ω. Let $c, s : \mathbb{R}^d \to \mathbb{R}$ be real valued functions, $\mathbf{b} \in \mathbb{R}^d$ be a vector function such that its entries are all Lipschitz continuous real-valued functions, and $\underline{\mathbf{a}}$ be a matrix such that $\underline{\mathbf{a}} = \{a_{ij}\}_{i,j=1}^{d}$, with a_{ij} Lipschitz continuous real-valued functions with $a_{ij} = a_{ji}$, $i, j = 1, ..., d$. Here $\underline{\mathbf{a}}$ is the diffusivity or diffusion coefficient, $\mathbf{b}$ is the velocity field (a function of space and time) of u the quantity of interest, and s is the 'sources' or 'sinks' of u. Consider the following PDE:

$$-\nabla \cdot (\underline{\mathbf{a}} \nabla u) + \nabla \cdot (\mathbf{b}u) + cu = s; \tag{2.1.1}$$

this is referred to as an equation with nonnegative characteristic form on the set $\Omega \subset \mathbb{R}^d$ if for each $\mathbf{x} \in \Omega$,

$$\sum_{i,j=1}^{d} a_{ij}(\mathbf{x}) \xi_i \xi_j \geq 0,$$

for any vector $\boldsymbol{\xi} = (\xi_i, ..., \xi_j)$ in $\mathbb{R}^d$. Let Γ represent the boundary of the domain Ω, and further define

$$\begin{aligned}
\Gamma_0 &= \left\{ \mathbf{x} \in \Gamma : \mathbf{n}(\mathbf{x})^\top \underline{\mathbf{a}}(\mathbf{x}) \mathbf{n}(\mathbf{x}) > 0 \right\}, \\
\Gamma_- &= \left\{ \mathbf{x} \in \Gamma \setminus \Gamma_0 : \mathbf{b}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) < 0 \right\}, \\
\Gamma_+ &= \left\{ \mathbf{x} \in \Gamma \setminus \Gamma_0 : \mathbf{b}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) \geq 0 \right\},
\end{aligned}$$

where $\mathbf{n}(\mathbf{x})$ is the unit outward normal vector to the boundary Γ at the point $\mathbf{x} \in \Gamma$. We refer to Γ_- as the inflow, and Γ_+ as the outflow boundary. Clearly the above sets are such that $\Gamma = \Gamma_0 \cup \Gamma_- \cup \Gamma_+$. If the set Γ_0 is non-empty, we may split it into two disjoint subsets Γ_{D} and Γ_{N}, such that Γ_{D} is non-empty. Respectively, Γ_{D} and Γ_{N} represent the Dirichlet and Neumann boundary conditions where we prescribe:

$$\begin{aligned}
\mathbf{u} &= g_{\mathrm{D}} \text{ on } \Gamma_{\mathrm{D}} \cup \Gamma_-, \\
\mathbf{n} \cdot (\underline{\mathbf{a}} \nabla u) &= g_{\mathrm{N}} \text{ on } \Gamma_{\mathrm{N}}.
\end{aligned} \tag{2.1.2}$$

We now provide some introductory examples relevant for CFD applications.

1. Selecting $a = \mathbf{I}_d$, $\mathbf{b} = \mathbf{0}$, and $c = 0$, with $\mathbf{I}_d$ denoting the $d \times d$ identity matrix and $\mathbf{0}$

the zero vector/matrix whose dimension will always be clear in a given context; we recover the Poisson equation $-\Delta u = s$ in Ω.

2. Let $a \in \mathbb{R}^{d\times d}$ and $\mathbf{b} \in \mathbb{R}^d$ such that

$$a = \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{d-1} \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 1 \\ \mathbf{0} \end{pmatrix},$$

and $c = 0$. This gives rise to the heat equation with the first coordinate direction representing the time direction.

3. Define $a = \mathbf{0}$, $\mathbf{b} \in \left[W^{1,\infty}(\Omega)\right]^d$, and $c \in L^\infty(\Omega)$; this choice gives rise to the first-order transport equation $\nabla \cdot (\mathbf{b}u) + cu = s$, with one variable possibly signifying the time direction.

4. For $d = 2$, let

$$a = \begin{pmatrix} x & 0 \\ 0 & 1 \end{pmatrix},$$

with $\mathbf{b} = \mathbf{0}$ and $c = 0$. Then, assuming the domain contains the line $x = 0$, the equation is hyperbolic when $x < 0$, elliptic when $x > 0$, and degenerates on the line $x = 0$. This is known as Keldyš equation [74, 97], which arises when modeling weak shock reflections off a wedge [41].

2.2 Sobolev Spaces

The solutions of PDEs are naturally found in Sobolev spaces, rather than in spaces of continuous functions with classical derivatives. The notion of a Sobolev space is based on that of a Lebesgue space $L^p(\Omega)$, where $\Omega \subset \mathbb{R}^d$, $d \geq 1$ and $p \in [1, \infty]$.

Definition 2.2.1. Let Ω be an open domain in $\mathbb{R}^d$, $d \geq 1$, then $L^p(\Omega)$, $p \in [1, \infty]$ is defined as the Lebesgue space of real-valued functions for which the norm defined by

$$\|u\|_{L^p(\Omega)} := \begin{cases} \left(\int_\Omega |u(\mathbf{x})|^p \, d\mathbf{x}\right)^{\frac{1}{p}}, & 1 \leq p < \infty, \\ \text{ess sup}\left\{|u(\mathbf{x})| : \mathbf{x} \in \Omega\right\}, & p = \infty, \end{cases}$$

is finite. For $k \in \mathbb{N} \cup \{0\}$, the Sobolev space $W_p^k(\Omega)$ is defined over the open domain $\Omega \subset \mathbb{R}^d$, $d \geq 1$ by

$$W_p^k(\Omega) := \left\{u \in L^p(\Omega) : D^\alpha u \in L^p(\Omega) \text{ for } |\alpha| \leq k\right\},$$

where D^α is the weak derivative of order $\alpha = (\alpha_1, ..., \alpha_d)$, with α being a standard multi-index notation; we associate the following norm $\|\cdot\|_{W_p^k(\Omega)}$, and semi-norm $|\cdot|_{W_p^k(\Omega)}$:

$$\|u\|_{W_p^k(\Omega)} := \left(\sum_{|\alpha|\leq k} \|D^\alpha u\|_{L^p(\Omega)}^p\right)^{\frac{1}{p}},$$
$$|u|_{W_p^k(\Omega)} := \left(\sum_{|\alpha|=k} \|D^\alpha u\|_{L^p(\Omega)}^p\right)^{\frac{1}{p}},$$

for $p = [1, \infty)$. For $p = \infty$, we define instead

$$\|u\|_{W_\infty^k(\Omega)} := \max_{|\alpha|\leq k} \|D^\alpha u\|_{L^\infty(\Omega)},$$
$$|u|_{W_\infty^k(\Omega)} := \max_{|\alpha|=k} \|D^\alpha u\|_{L^\infty(\Omega)}.$$

We refer to k as the Sobolev index. In the particular case when $p = 2$, the space $W_2^k(\Omega)$ with standard inner product is a Hilbert space, which we denote as $W_2^k(\Omega) =: H^k(\Omega)$. Finally, if we also have that $k = 0$, then we are left with the standard L^2 space where $H^0(\Omega) = L^2(\Omega)$.

2.3 Existence and Uniqueness of the Solution

Before we progress to solve (2.1.1), we must first be confident in the knowledge that a solution does indeed exist, and that it is unique. Existence and uniqueness in the weak setting was first considered by Oleĭnik and Radkevič [109] in various settings with certain regularity assumptions presented therein. Later Houston, Schwab and Süli were able to replicate this work with more relaxed regularity constraints [90]. It is these results we consider here. We simplify the analysis by first setting $g_\mathrm{N} = 0$. We define the space

$$\mathcal{V} := \left\{v \in H^1(\Omega) : v(\mathbf{x}) = 0 \; \forall \mathbf{x} \in \Gamma_\mathrm{D}\right\}.$$

Let $\mathcal{H}$ be the closure of $\mathcal{V}$ in $L^2(\Omega)$ with respect to the norm $\|\cdot\|_\mathcal{H} := \sqrt{(\cdot,\cdot)_\mathcal{H}}$, where $(\cdot,\cdot)_\mathcal{H}$ is an inner product defined by

$$(w, v)_\mathcal{H} := (\underline{\mathbf{a}}\nabla w, \nabla v) + (cw, v) + \langle w, v\rangle_{\Gamma_- \cup \Gamma_+ \cup \Gamma_\mathrm{N}}.$$

In this setting, $(\cdot,\cdot)$ and $\langle\cdot,\cdot\rangle_\gamma$ are inner products defined, respectively, by

$$(w, v) = \int_\Omega wv \, \mathrm{d}\mathbf{x},$$
$$\langle w, v\rangle_\gamma = \int_\gamma |\mathbf{b}\cdot\mathbf{n}| \, wv \, \mathrm{ds}.$$

It can be shown that $\mathcal{H}$ is a Hilbert space. We define the bilinear form $B(\cdot,\cdot) : \mathcal{H} \times \mathcal{V} \to \mathbb{R}$, as well as the linear functional $l : \mathcal{V} \to \mathbb{R}$ by

$$\begin{aligned} B(w,v) &= (\underline{\mathbf{a}}\nabla w, \nabla v) - (w, \mathbf{b}\cdot\nabla v) + (cw, v) + \langle w, v\rangle_{\Gamma_+\cup\Gamma_N}, \\ l(v) &= (s, v). \end{aligned}$$

We say that $u \in \mathcal{H}$ is a weak solution to the boundary value problem (2.1.1) with boundary conditions; $u = g_D$ on $\Gamma_D \cup \Gamma_-$ and $(\underline{\mathbf{a}}\nabla u)\cdot\mathbf{n} = g_N$ on Γ_N if

$$B(u,v) = l(v) \;\; \forall v \in \mathcal{V}.$$

Theorem 2.3.1. *[91] Let* (2.1.1) *together with the respective boundary conditions* (2.1.2) *be given. Furthermore, suppose that* $\mathbf{b}\cdot\mathbf{n} \geq 0$ *on* Γ_N*, and that there exists a positive constant* γ_0 *such that*

$$c(\mathbf{x}) + \frac{1}{2}\nabla\cdot\mathbf{b}(\mathbf{x}) \geq \gamma_0, \; \mathbf{x} \in \Omega.$$

Then, for every $s \in L^2(\Omega)$*, there exists* $u \in \mathcal{H}$ *such that* $B(u,v) = l(v)$*. Moreover, there exists a Hilbert subspace* $\mathcal{H}'$ *of* $\mathcal{H}$ *such that* $u \in \mathcal{H}'$ *and* u *is the unique element of* $\mathcal{H}'$ *such that* $B(u,v) = l(v)$*.*

2.4 Mesh Design

We partition the domain Ω into a subdivision $\mathcal{T}_h = \{\kappa\}$, where κ are disjoint open element domains and h_κ is the local mesh size, such that $\Omega = \cup_{\kappa\in\mathcal{T}_h}\kappa$. These are defined such that there exists an affine mapping $F_\kappa : \hat{\kappa} \to \kappa$ from the reference element $\hat{\kappa}$ to the physical element κ. For the purposes of this example, we assume that $\hat{\kappa}$ is the unit hypercube $(-1,1)^d$, see Figure 2.4.1, but this could also be the unit d-simplex. In Chapter 7, however, we show that we may do away with this mapping altogether, defining all quantities on a general polytope instead.

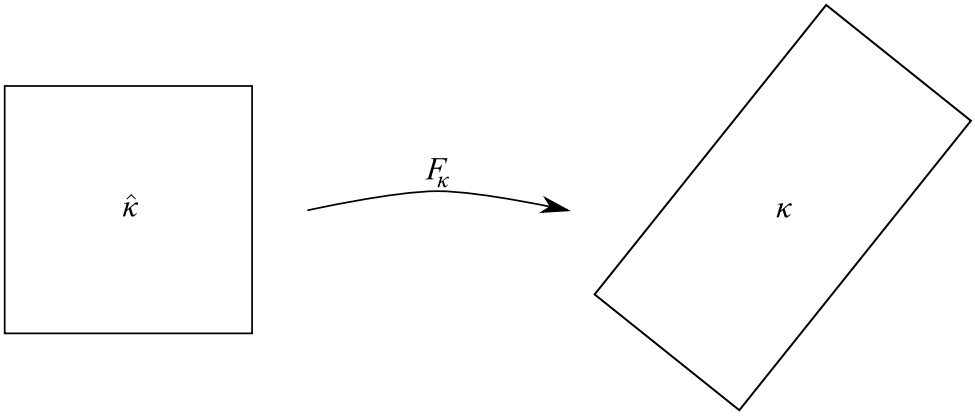

Figure 2.4.1: The mapping F_κ from the reference element $\hat{\kappa}$ to the physical element κ.

Consider two adjacent elements $\kappa_i, \kappa_j \in \mathcal{T}_h$ with boundaries $\partial\kappa_i$ and $\partial\kappa_j$ respectively. We define an interior face of $\mathcal{T}_h$ to be the non-empty set $\partial\kappa_i \cap \partial\kappa_j$ and define Γ_{int} to be the union of all interior faces of $\mathcal{T}_h$. One further concept we require before proceeding, is that of shape regularity.

Definition 2.4.1. (Shape Regularity) A family of subdivisions $\mathcal{T}_h$ is said to be shape-regular if there exists a positive constant C_r such that, for all $\kappa \in \mathcal{T}_h$, we have

$$\frac{h_\kappa}{\rho_\kappa} \leq C_r. \tag{2.4.1}$$

The constant C_r is independent of the varying mesh parameters. Hence, ρ_κ denotes the diameter of the largest ball contained in κ, and h_κ is the diameter of κ.

As we show throughout this work, DGFEMs can consider very general meshes making them perfectly suited for turbomachinery problems. Traditionally, FVMs for industrial applications make use of highly optimised structured grids with refinement blocks in areas where the underlying geometry is complex, i.e., around blade edges. DGFEMs are able to both utilise these pre-existing meshing methods, as well as the unstructured grids constructed through algorithms such as the Delaunay triangulation [113]. In Chapter 5, and subsequently Chapter 6, we also demonstrate that the proposed interior penalty method may do away with the need for block-structured grids altogether, as well as make use of differing polynomial orders to represent the numerical solution.

Definition 2.4.2. For an open set Ω with corresponding subdivision $\mathcal{T}_h$, the broken Sobolev space of composite order m is defined as

$$H^m(\Omega, \mathcal{T}_h) = \left\{ u \in L^2(\Omega) : u|_\kappa \in H^{m_\kappa}(\kappa) \ \forall \kappa \in \mathcal{T}_h \right\},$$

where $m := \{m_\kappa\}_{\kappa \in \mathcal{T}_h}$; the associated norm and seminorm are then defined, respectively, as

$$\|u\|_{m,\mathcal{T}_h} = \left(\sum_{\kappa \in \mathcal{T}_h} \|u\|^2_{H^{m_\kappa}(\kappa)} \right)^{\frac{1}{2}},$$
$$\|u\|_{m,\mathcal{T}_h} = \left(\sum_{\kappa \in \mathcal{T}_h} |u|^2_{H^{m_\kappa}(\kappa)} \right)^{\frac{1}{2}}.$$

Let $u \in H^1(\Omega, \mathcal{T}_h)$ and $\boldsymbol{\tau} \in \left[H^1(\Omega, \mathcal{T}_h)\right]^d$, then the broken gradient $\nabla_{\mathcal{T}_h} u$ of u and the broken divergence $\nabla_{\mathcal{T}_h} \cdot \boldsymbol{\tau}$ of $\boldsymbol{\tau}$ are defined element wise, as

$$\left(\nabla_{\mathcal{T}_h} u\right)\big|_\kappa = \nabla\left(u|_\kappa\right),\ \kappa \in \mathcal{T}_h,$$
$$\left(\nabla_{\mathcal{T}_h} \cdot \tau\right)\big|_\kappa = \nabla \cdot \left(\tau|_\kappa\right),\ \kappa \in \mathcal{T}_h.$$

2.5 Trace Operators

Consider a function $v \in H^1(\Omega, \mathcal{T}_h)$. We introduce a set of operators on the function v to define the behaviour on $v|_{\partial\kappa}$. For an element κ_i and its neighbour κ_j that share a face $f \in \Gamma_{\text{int}}$, the jump of v across f, and the mean value of v on f, are given respectively as

$$[v] = v|_{\partial\kappa_i \cap f} - v|_{\partial\kappa_j \cap f},$$
$$\langle v \rangle = \frac{1}{2}(v|_{\partial\kappa_i \cap f} + v|_{\partial\kappa_j \cap f}).$$

When $f \in \Gamma \setminus \Gamma_{\text{int}}$ then the following definitions are used instead:

$$[v] = v|_{\partial\kappa \cap f},$$
$$\langle v \rangle = v|_{\partial\kappa \cap f}.$$

Also, for every $\kappa \in \mathcal{T}_h$, let v_h^+ and v_h^- denote the interior and exterior traces, respectively, of the function v on $\partial\kappa$ and $\partial\kappa \cap \Gamma \neq \emptyset$.

2.6 Finite Element Spaces

Consider an element $\hat{\kappa} \in \mathcal{T}_h$, for polynomial degree p, we define two polynomial spaces by

$$\mathcal{Q}_p \coloneqq \operatorname{span}\left\{\prod_{i=1}^{d} \widehat{x}_i^{\alpha_i} : 0 \leq \alpha_i \leq p_i\right\},$$
$$\mathcal{P}_p \coloneqq \operatorname{span}\left\{\prod_{i=1}^{d} \widehat{x}_i^{\alpha_i} : 0 \leq \sum_{i=1}^{d} \alpha_i \leq p\right\}.$$

We associate a polynomial degree $p_\kappa \geq 1$ with every $\kappa \in \mathcal{T}_h$. Furthermore, for conciseness we introduce the notation

$$\mathcal{R}_p \coloneqq \begin{cases} \mathcal{Q}_p \text{ if } \widehat{\kappa} \text{ is the unit hypercube,} \\ \mathcal{P}_p \text{ if } \widehat{\kappa} \text{ is the unit simplex,} \end{cases}$$

thereby, if $\widehat{\kappa}$ is a simplex we use polynomials from $\mathcal{P}_p$, otherwise if $\widehat{\kappa}$ is a hypercube we use polynomials from $\mathcal{Q}_p$. Introducing $\mathbf{p} = \{p_\kappa : \kappa \in \mathcal{T}_h\}$ and $F_\kappa : \kappa \in \mathcal{T}_h$, we are able to concisely define the discontinuous finite element space

$$S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F}) \coloneqq \left\{u \in L^2(\Omega) : u\,|_\kappa \circ F_\kappa \in \mathcal{R}_{p_\kappa}\right\}. \tag{2.6.1}$$

2.7 A DGFEM for PDEs with Nonnegative Characteristic Form

We now return to the model problem defined previously, and are interested in deriving a suitable interior penalty DGFEM. On each element $\kappa \in \mathcal{T}_h$, we define an inflow and an outflow boundary denoted, respectively, by

$$\begin{aligned} \partial_- \kappa &= \{\mathbf{x} \in \partial\kappa : \mathbf{b}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) < 0\}, \\ \partial_+ \kappa &= \{\mathbf{x} \in \partial\kappa : \mathbf{b}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) \geq 0\}, \end{aligned} \tag{2.7.1}$$

where $\mathbf{n}(\mathbf{x})$ is the unit outward normal vector to κ at the point $\mathbf{x} \in \partial\kappa$.

The first stage in deriving a DGFEM is to convert the advection-diffusion-reaction equation (2.1.1) into a system of two first order equations

$$\begin{aligned} \Phi - \underline{\mathbf{a}} \nabla u &= 0, \\ -\nabla\Phi + \nabla \cdot (\mathbf{b}u) + cu &= s. \end{aligned} \tag{2.7.2}$$

We now introduce suitably smooth test functions, the vector $\boldsymbol{\tau}$ and scalar v, multiplying each equation of (2.7.2), respectively, and integrating by parts over $\kappa \in \mathcal{T}_h$, we obtain

$$\int_\kappa \Phi \cdot \boldsymbol{\tau} \,\mathrm{d}\mathbf{x} + \int_\kappa \nabla \cdot (\underline{\mathbf{a}}\boldsymbol{\tau}) u \,\mathrm{d}\mathbf{x} + \int_{\partial\kappa} (\underline{\mathbf{a}}\boldsymbol{\tau}) \cdot \mathbf{n} u \,\mathrm{ds} = 0,$$

$$\int_\kappa \Phi \cdot \nabla v \,\mathrm{d}\mathbf{x} - \int_{\partial\kappa \setminus \Gamma_\mathrm{N}} \Phi \cdot \mathbf{n} v \,\mathrm{ds} - \int_{\partial\kappa \cap \Gamma_\mathrm{N}} g_\mathrm{N} v \,\mathrm{ds}$$
$$- \int_\kappa u\mathbf{b} \cdot \nabla v \,\mathrm{d}\mathbf{x} + \int_{\partial\kappa} \mathbf{b} \cdot \mathbf{n} uv \,\mathrm{ds} + \int_\kappa cuv \,\mathrm{d}\mathbf{x} = \int_\kappa sv \,\mathrm{d}\mathbf{x}.$$

We aim to restrict the problem to a finite-dimensional one. Therefore, we proceed by restricting the choice of trial and test functions to subspaces based on $S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F})$. Then, summing over all elements κ of the domain, and introducing the following numerical fluxes $\widehat{u_h}$, $\mathcal{H}\left(u_h^+, u_h^-, n_\kappa\right)$and $\widehat{\Phi_h \cdot \mathbf{n}_f}$ we are able to define the flux formulation of the problem. Numerical fluxes are an important aspect for the construction of a DGFEM, since they are the method by which information is passed across element boundaries. Indeed, different numerical fluxes gives rise to different DGFEM formulations [12]. It is these fluxes that allow us to weakly enforce continuity between neighbouring elements. Here the fluxes $\widehat{u_h}$, $\mathcal{H}\left(u_h^+, u_h^-, n_\kappa\right)$ and $\widehat{\Phi_h \cdot \mathbf{n}_f}$ represent consistent approximations to u, $\mathbf{b} \cdot \mathbf{n} u$ and $\underline{\mathbf{a}} \nabla u \cdot \mathbf{n}$ on the faces of elements, respectively. In particular, we say the numerical fluxes are consistent if

$$\widehat{u_h}\left(v^+, v^-\right)|_f = v,$$
$$\mathcal{H}\left(v^+, v^-, n_\kappa\right)|_{\partial\kappa} = \mathbf{b} \cdot \mathbf{n} v,$$
$$\widehat{\Phi_h \cdot \mathbf{n}_f}\left(v^+ . \nabla v^+ . v^- . \nabla v^-\right)|_f = \underline{\mathbf{a}} \nabla v \cdot \mathbf{n}_f,$$

for any smooth function v satisfying the boundary conditions. Further, the numerical fluxes are conservative if they are single-valued on every face f in the mesh.

Consequently, summing over all the elements we have the following

$$\sum_{\kappa \in \mathcal{T}_\kappa} \left(\int_\kappa \Phi_h \cdot \tau \,\mathrm{d}\mathbf{x} + \int_\kappa \nabla \cdot (\underline{\mathbf{a}}\tau) u_h \,\mathrm{d}\mathbf{x} \right) - \int_{\Gamma_\mathrm{int} \cup \Gamma_0} \left[(\underline{\mathbf{a}}\tau) \cdot \mathbf{n}_f \right] \widehat{u_h} \,\mathrm{ds} = 0, \tag{2.7.3}$$

$$\sum_{\kappa \in \mathcal{T}_\kappa} \left(\int_\kappa \Phi \cdot \nabla v \,\mathrm{d}\mathbf{x} - \int_\kappa u_h \mathbf{b} \cdot \nabla v \,\mathrm{d}\mathbf{x} - \int_\kappa c u_h v \,\mathrm{d}\mathbf{x} + \int_{\partial\kappa} \mathcal{H}\left(u_h^+, u_h^-, n_\kappa\right) v^+ \,\mathrm{ds} \right)$$
$$- \int_{\Gamma_\mathrm{int} \cup \Gamma_\mathrm{D}} \widehat{\Phi_h \cdot \mathbf{n}_f} \left[v \right] \,\mathrm{ds} = \sum_{\kappa \in \mathcal{T}_\kappa} \left(\int_\kappa sv \,\mathrm{d}\mathbf{x} + \int_{\partial\kappa \cap \Gamma_\mathrm{N}} g_\mathrm{N} v \,\mathrm{ds} \right). \tag{2.7.4}$$

The numerical fluxes we shall use are defined as follows

$$\mathcal{H}\left(u_h^+, u_h^-, n_\kappa\right)|_{\partial\kappa} = \begin{cases} \mathbf{b}\cdot\mathbf{n}_\kappa g_{\mathrm{D}} & \mathbf{x}\in\partial\kappa\cap\left(\Gamma_{\mathrm{D}}\cup\Gamma_-\right), \\ \mathbf{b}\cdot\mathbf{n}_\kappa \lim_{m\to 0^+} u_h(\mathbf{x}-m\mathbf{b}) & \text{otherwise,} \end{cases}$$

$$\widehat{u_h} = \begin{cases} \langle u_h\rangle & f\subset\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{N}}, \\ g_{\mathrm{D}} & f\subset\Gamma_{\mathrm{D}}, \end{cases}$$

$$\widehat{\Phi_h\cdot\mathbf{n}_f} = \begin{cases} \langle(\underline{\mathbf{a}}\nabla u_h)\cdot\mathbf{n}_e\rangle - \sigma\left[u_h\right] & f\subset\Gamma_{\mathrm{int}}, \\ (\underline{\mathbf{a}}\nabla u_h|_f)\cdot\mathbf{n}_f - \sigma\left(u_h|_f - g_{\mathrm{D}}\right) & f\subset\Gamma_{\mathrm{D}}. \end{cases}$$

$\mathcal{H}\left(u_h^+, u_h^-, n_\kappa\right)$ represents the upwind flux and is chosen in an upwind fashion aiming to make the method stable. It takes on the value found upstream, or upwind of the current position $\mathbf{x}$, and a DGFEM discretisation based on this flux is both stable and consistent. σ is known as the discontinuity-penalisation function and it has the purpose of penalising any jump discontinuities across element boundaries, i.e., on $f\in\Gamma_{\mathrm{int}}$, and also the task of weakly imposing the Dirichlet boundary conditions on $f\in\Gamma_{\mathrm{D}}$. Later, we discuss how to select a more precise value for this parameter, but for the purposes of this initial study, we choose

$$\sigma = C_\sigma\frac{p^2}{h},$$

with C_σ large enough according to [70]. With the above choice of fluxes, we have the interior penalty DGFEM of Houston, Schwab and Süli [92].

Whilst it is acceptable to leave (2.7.3) and (2.7.4) in their current form, we proceed instead by eliminating the variable Φ_h through setting $\boldsymbol{\tau} = \nabla v$ in (2.7.3), and integrating by parts. Then, substituting for the term $\int_k \Phi_h\cdot\boldsymbol{\tau}\,d\mathbf{x}$, we combine (2.7.3) and (2.7.4) to arrive at the primal form: find $u_{DG}\in S^{\mathbf{p}}\left(\Omega,\mathcal{T}_h,\mathbf{F}\right)$ such that $B_{DG}(u_{DG},v) = l_{DG}(v)$, for all $v\in S^{\mathbf{p}}\left(\Omega,\mathcal{T}_h,\mathbf{F}\right)$, where

$$\begin{aligned} B_{DG}(w,v) &= B_A(w,v) + B_B(w,v) + B_C(w,v) + B_D(w,v) + B_E(w,v), \\ l_{DG}(v) &= l_A(v) + l_B(v), \end{aligned}$$

with

$$
\begin{aligned}
B_A(w,v) &= \sum_{\kappa\in\mathcal{T}_h}\left(\int_\kappa \underline{\mathbf{a}}\nabla w\cdot\nabla v\,\mathrm{d}\mathbf{x}\right),\\
B_B(w,v) &= \sum_{\kappa\in\mathcal{T}_h}\left(-\int_\kappa (w\mathbf{b}\cdot\nabla v - cwv)\;\mathrm{d}\mathbf{x}\right),\\
B_C(w,v) &= \sum_{\kappa\in\mathcal{T}_h}\left(\int_{\partial_+\kappa}(\mathbf{b}\cdot\mathbf{n}_\kappa)\,w^+v^+\,\mathrm{d}\mathrm{s}+\int_{\partial_-\kappa\setminus\Gamma}(\mathbf{b}\cdot\mathbf{n}_\kappa)\,w^-v^+\,\mathrm{d}\mathrm{s}\right),\\
B_D(w,v) &= \theta\int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}}\langle(\underline{\mathbf{a}}\nabla w)\cdot\mathbf{n}_f\rangle\,[v]\;\mathrm{d}\mathrm{s}-\int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}}\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\rangle\,[w]\;\mathrm{d}\mathrm{s},\\
B_E(w,v) &= \int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}}\sigma\,[w]\,[v]\;\mathrm{d}s,
\end{aligned}
$$

and

$$
\begin{aligned}
l_A(v) &= \sum_{\kappa\in\mathcal{T}_h}\left(\int_\kappa sv\,\mathrm{d}\mathbf{x}-\int_{\partial_-\kappa\cap(\Gamma_{\mathrm{D}}\cup\Gamma_-)}(\mathbf{b}\cdot\mathbf{n}_\kappa)\,g_{\mathrm{D}}v^+\,\mathrm{d}\mathrm{s}+\int_{\partial_-\kappa\cap\Gamma_{\mathrm{D}}}\theta g_{\mathrm{D}}\left(\left(\underline{\mathbf{a}}\nabla v^+\right)\cdot\mathbf{n}_\kappa\right)\;\mathrm{d}\mathrm{s}\right),\\
l_B(v) &= \sum_{\kappa\in\mathcal{T}_h}\left(\int_{\partial\kappa\cap\Gamma_{\mathrm{N}}}g_{\mathrm{N}}v^+\,\mathrm{d}\mathrm{s}+\int_{\partial\kappa\cap\Gamma_{\mathrm{D}}}\sigma g_D v^+\,\mathrm{d}\mathrm{s}\right).
\end{aligned}
$$

For the symmetric interior penalty method, we set the parameter θ to be equal to -1, resulting in a bilinear form which is symmetric in its diffusive terms, $B_A(w,v)+B_C(w,v)+B_D(w,v)+B_E(w,v)$. Other interior penalty methods can be produced through different choices of numerical fluxes and by setting $\theta=0$ for the incomplete penalty method, or $\theta=1$ for the non-symmetric interior penalty method.

2.8 Stability

We now explore the stability of the above DGFEM we have just presented. We wish to know if the DGFEM approximate solution to (2.1.1) exists and whether it is unique. This brings into question the coercivity of the bilinear form.

Definition 2.8.1. A bilinear form $B(\cdot,\cdot)$ on a normed linear space H, equipped with norm $\|\cdot\|_H$, is said to be continuous if there exists a constant $C<\infty$ such that

$$
|B(w,v)|\le C\,\|w\|_H\,\|v\|_H\;\;\forall w,v\in H. \tag{2.8.1}
$$

The bilinear form $B(\cdot,\cdot)$ is said to be coercive on $H\times H$ if there exists a constant $\alpha>0$ such that

$$
B(w,w)\ge\alpha\,\|w\|_H^2\;\;\forall w\in H. \tag{2.8.2}
$$

Theorem 2.8.2. *If $B(\cdot,\cdot)$ is a coercive bilinear form on a normed, linear, finite-dimensional space, H, then for any linear functional $l(\cdot)$ there exists a unique $u \in H$ such that*

$$B(u,v) = l(v)\ \forall w \in H. \tag{2.8.3}$$

Proof. Coercivity of the bilinear form $B(\cdot,\cdot)$, over $H \times H$, implies that if $B(w,w) = 0$, then $w \equiv 0$. This therefore implies the uniqueness of the solution. Formally, suppose we have two different solutions of (2.8.3), say u and u^*. Then

$$B(u,v) - B(u^*,v) = B(u-u^*,v) = l(v) - l(v) = 0\ \forall v \in H.$$

Choosing v such that $v = u - u^*$, yields $B(u-u^*, u-u^*) = 0$, thus $u \equiv u^*$. Since the space H is finite-dimensional and (2.8.3) is a linear problem, the existence of a solution to (2.8.3) is guaranteed by the fact that its homogeneous counterpart has the unique solution $u_{DG} \equiv 0$. □

Therefore, if we are able to prove coercivity of the bilinear form $B_{DG}(\cdot,\cdot)$ on the space $S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F})$, then existence and uniqueness of the solution follow. Note that we only consider the case of isotropic elements for the purposes of this introductory chapter, although anisotropic [67, 68, 77] and most recently polytopic elements [36, 38] have been considered in the literature. We make one further simplification before proceeding, and assume that the entries of the matrix $\underline{\mathbf{a}}$ are constant on each element $\kappa \in \mathcal{T}_h$, *i.e.*

$$\underline{\mathbf{a}} \in \left[S^{\mathbf{0}}(\Omega, \mathcal{T}_h, \mathbf{F})\right]^{d\times d}_{sym}$$

and let $\underline{\mathbf{a}}_\kappa \coloneqq \underline{\mathbf{a}}\ |_\kappa$. We also define $\bar{\underline{\mathbf{a}}} = \left|\sqrt{\underline{\mathbf{a}}}\right|_2^2$, where $|\cdot|_2$ denotes the matrix norm subordinate to the l_2-vector norm on $\mathbb{R}^d$:

$$|A|_2 \coloneqq \max_{v\in\mathbb{R}^d\setminus\{0\}} \frac{\|Av\|_2}{\|v\|_2} \quad A \in \mathbb{R}^{d\times d}.$$

We now seek to prove coercivity by first considering the following result.

Lemma 2.8.3. *Let $\hat{\kappa}$ be either the unit d-hypercube or the unit d-simplex, $d = 2, 3$, then for any function $\hat{v} \in \mathcal{R}_p(\hat{\kappa})$, $p \geq 1$, there exists a positive constant, C'_{inv} independent of $\hat{u}$ such that for any element face $\hat{f} \subset \partial\hat{\kappa}$, we have*

$$\|\hat{v}\|^2_{\hat{f}} \leq C'_{inv} p^2 \|\hat{v}\|^2_{L^2(\hat{\kappa})}. \tag{2.8.4}$$

Proof. A proof of (2.8.4) is omitted here; see [124] for details. □

We now need to scale (2.8.4) to the physical element κ.

Lemma 2.8.4. *Let κ be an element contained in the mesh $\mathcal{T}_h$ and let f denote one of its faces. Then, the following inverse inequality holds*

$$\|v\|^2_{L^2(f)} \leq C_{inv} \frac{p^2}{h} \|v\|^2_{L^2(\kappa)}, \tag{2.8.5}$$

for all v such that $v \circ F_\kappa \in \mathcal{R}_p(\hat{\kappa})$, where C_{inv} is a constant which depends only on the dimension d.

Proof. We employ (2.8.4) and re-scale both the left- and right-hand sides. In particular, for the left-hand side of (2.8.5), we make use of the affine mapping F_κ between the reference and physical element.

$$\|v\|^2_{L^2(f)} \leq C_1 \|\hat{v}\|^2_{L^2(\hat{f})} \tag{2.8.6}$$

For the right-hand side we have

$$\|\hat{v}\|^2_{L^2(\hat{\kappa})} = \det\left(F_\kappa^{-1}\right) \|v\|^2_{L^2(\kappa)} = \frac{1}{h} \|v\|^2_{L^2(\kappa)} \leq \frac{C_2}{h} \|v\|^2_{L^2(\kappa)} \tag{2.8.7}$$

where C_1 and C_2 are positive real constants. Finally, inserting (2.8.6) and (2.8.7) into (2.8.4) yields the desired result. □

We are now in a position to state and prove the following coercivity result for the bilinear form $B_{DG}(\cdot,\cdot)$ over $S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F}) \times S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F})$. Before proceeding however, it is required that we define a norm on the space $S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F})$. Following [78], we define the DG-norm $\|\cdot\|_{DG}$ by

$$\begin{aligned}\|w\|^2_{DG} = \sum_{\kappa \in \mathcal{T}_\kappa} \Bigg(& \|\sqrt{\underline{\mathbf{a}}} \nabla w\|^2_{L^2(\kappa)} + \|c_0 w\|^2_{L^2(\kappa)} + \frac{1}{2} \left\|w^+\right\|^2_{\partial_- \kappa \cap (\Gamma_{\mathrm{D}} \cup \Gamma_-)} \\ & + \frac{1}{2} \left\|w^+ - w^-\right\|^2_{\partial_- \kappa \setminus \Gamma} + \frac{1}{2} \left\|w^+\right\|^2_{\partial_+ \kappa \cap \Gamma} \Bigg) \\ & + \int_{\Gamma_{\mathrm{int}} \cup \Gamma_{\mathrm{D}}} \theta [w]^2 \, \mathrm{ds} + \int_{\Gamma_{\mathrm{int}} \cup \Gamma_{\mathrm{D}}} \frac{1}{\theta} \langle (\underline{\mathbf{a}} \nabla w) \cdot \mathbf{n}_f \rangle^2 \, \mathrm{ds},\end{aligned}$$

where $\|\cdot\|_\tau$, $\tau \subset \partial\kappa$ is the norm induced from the inner product

$$(v, w)_\tau = \int_\tau |\mathbf{b} \cdot \mathbf{n}_\kappa| \, vw \, \mathrm{ds},$$

and c_0 is defined as

$$(c_0(\mathbf{x}))^2 = c(\mathbf{x}) + \frac{1}{2}\nabla \cdot \mathbf{b}(\mathbf{x}),\ \forall \mathbf{x} \in \Omega.$$

Additionally, we define the function $a \in L_\infty(\Gamma_{\text{int}} \cup \Gamma_{\text{D}})$ by $a(x) = \max\left\{\bar{\underline{\mathbf{a}}}_{\kappa_1}, \bar{\underline{\mathbf{a}}}_{\kappa_2}\right\}$ if x is in the interior of $f = \partial\kappa_1 \cap \kappa_2$, and $a(x) = \bar{\underline{\mathbf{a}}}_{\kappa}$ if x is in the interior of $\partial\kappa \cap \Gamma_{\text{D}}$. Similarly, we define the function $p \in L_\infty(\Gamma_{\text{int}} \cup \Gamma_{\text{D}})$ by $p(x) = \max\{p_{\kappa_1}, p_{\kappa_2}\}$ if x is in the interior of $f = \partial\kappa_1 \cap \kappa_2$, and $p(x) = p_\kappa$ if x is in the interior of $\partial\kappa \cap \Gamma_{\text{D}}$.

Theorem 2.8.5. *If σ is defined as*

$$\sigma\,|_f = \sigma_f = C_\sigma \frac{ap^2}{h} \text{ for } f \subset \Gamma_D \cup \Gamma_{int}, \tag{2.8.8}$$

then there exists a positive constant C_s, which depends only on the dimension d and the shape regularity of $\mathcal{T}_h$, such that

$$B_{DG}(v,v) \geq C_s \|v\|_{DG}^2\ \forall v \in S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F}), \tag{2.8.9}$$

provided that the constant C_σ is chosen such that $C_\sigma > C'_\sigma > 0$, where C'_σ is a sufficiently large positive constant.

Proof. The proof is well known, but we prefer to present it here as it is instructive on the importance of inverse estimates in the stability of DGFEMs of interior penalty type. Let C_s be an arbitrary real number and pick $v \in S^{\mathbf{p}}(\Omega, \mathcal{T}_h, \mathbf{F})$. Then,

$$\begin{aligned} B_{DG}(v,v) - C_s\|v\|_{DG}^2 = &(1-C_s)\left(B_A(v,v) + B_B(v,v) + B_C(v,v)\right) \\ &- 2\int_{\Gamma_{\text{int}}\cup\Gamma_{\text{D}}} \langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\rangle\,[v]\ \text{ds} \\ &- C_s\int_{\Gamma_{\text{int}}\cup\Gamma_{\text{D}}} \frac{1}{\sigma}\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\rangle^2\ \text{ds}. \end{aligned} \tag{2.8.10}$$

Restricted to the face $f \subset \Gamma_{\text{int}}$, the interface between the elements κ_i and κ_j the last term on the right-hand side of (2.8.10) can be bounded as follows:

$$\int_f \frac{1}{\sigma}\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\rangle^2\ \text{ds} \leq \int_f \frac{1}{2\sigma}\left[((a_{\kappa_i}\nabla v)\cdot\mathbf{n}_f)^2 + \left(\left(a_{\kappa_j}\nabla v\right)\cdot\mathbf{n}_f\right)^2\right]\ \text{ds} \tag{2.8.11}$$

As σ is constant on each face, and using (2.8.5) we are able to derive the following bound

$$\begin{aligned}\int_f \frac{1}{\sigma}\left((a_{\kappa_i}\nabla v)\cdot\mathbf{n}_f\right)^2 \,\mathrm{ds} &= \frac{1}{\sigma}\left\|a_{\kappa_i}\nabla v\right\|^2_{L^2(f)}\\ &\leq \frac{C_{\mathrm{inv}}}{\sigma}\frac{p^2_{\kappa_i}}{h_{\kappa_i}}\left\|a_{\kappa_i}\nabla v\right\|^2_{L^2(\kappa_i)}\\ &\leq \frac{C_{\mathrm{inv}}}{\sigma}\frac{p^2_{\kappa_i}}{h_{\kappa_i}}\bar{a}_{\kappa_i}\left\|\sqrt{a_{\kappa_i}}\nabla v\right\|^2_{L^2(\kappa_i)}.\end{aligned}$$

Therefore, setting $\sigma\,|_f = C_\sigma \frac{ap^2}{h}$ yields

$$\int_f \frac{1}{\sigma}\left((a_{\kappa_i}\nabla v)\cdot\mathbf{n}_f\right)^2 \,\mathrm{ds} \leq \frac{C_{\mathrm{inv}}}{C_\sigma}\left\|\sqrt{a_{\kappa_i}}\nabla v\right\|^2_{L^2(\kappa_i)}.$$

Hence

$$\int_f \frac{1}{\sigma}\left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle^2 \,\mathrm{ds} \leq \frac{C_{\mathrm{inv}}}{2C_\sigma}\left[\left\|\sqrt{\underline{\mathbf{a}}}\nabla v\right\|^2_{L^2(\kappa_i)} + \left\|\sqrt{\underline{\mathbf{a}}}\nabla v\right\|^2_{L^2(\kappa_j)}\right].$$

By employing (2.8.8) an analogous argument for $f\in\Gamma_{\mathrm{D}}$ yields

$$\int_f \frac{1}{\sigma}\left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle^2 \,\mathrm{ds} \leq \frac{C_{\mathrm{inv}}}{C_\sigma}\left\|\sqrt{a}\nabla v\right\|^2_{L^2(\kappa)}.$$

Hence

$$\int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}} \frac{1}{\sigma}\left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle^2 \,\mathrm{ds} \leq \frac{C_{\mathrm{inv}}C_f}{C_\sigma}B_A(v,v), \tag{2.8.12}$$

where C_f is dependent on the maximum number of element interactions on an element boundary, that is

$$C_f = \max_{\kappa\in\mathcal{T}_h}\mathrm{card}\left\{f\in\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}} : f\subset\partial\kappa\right\}.$$

For a face $f\in\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}$ the following bound holds:

$$\begin{aligned}2\int_f \left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle[v] \,\mathrm{ds} &\leq 2\sqrt{\int_f \frac{1}{\sigma}\left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle^2 \mathrm{ds}}\sqrt{\int_f \sigma\,[v]^2 \,\mathrm{ds}}\\ &\leq \epsilon\int_f \frac{1}{\sigma}\left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle^2 \mathrm{ds} + \frac{1}{\epsilon}\int_f \sigma\,[v]^2 \,\mathrm{ds},\end{aligned}$$

for any $\epsilon>0$. Summing over all faces $f\in\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}$ and utilising (2.8.12) we obtain

$$2\int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}} \left\langle(\underline{\mathbf{a}}\nabla v)\cdot\mathbf{n}_f\right\rangle[v] \,\mathrm{ds} \leq \epsilon\frac{C_{\mathrm{inv}}C_f}{C_\sigma}B_A(v,v) + \frac{1}{\epsilon}B_E(v,v),$$

and, therefore,

$$B_{DG}(v,v) - C_s \|v\|_{DG}^2 \geq \left(1 - C_s - (C_s + \epsilon)\frac{C_{\text{inv}}C_f}{C_\sigma}\right) B_A(v,v) + (1 - C_s)\left(B_B(v,v) + B_C(v,v)\right) + \left(1 - C_s - \frac{1}{\epsilon}\right) B_E(v,v).$$

Evidently $B_A(v,v)$ and $B_E(v,v)$ are non-negative, and similarly

$$B_B(v,v) + B_C(v,v) \geq 0,$$

due to Theorem 2.8.2. Hence, it follows that if

$$\left(1 - C_s - (C_s + \epsilon)\frac{C_{\text{inv}}C_f}{C_\sigma}\right) > 0,\ (1 - C_s) > 0,\ \left(1 - C_s - \frac{1}{\epsilon}\right) > 0, \tag{2.8.13}$$

then, coercivity holds. However, the last inequality only holds provided $\epsilon > 1$, while the first inequality implies that

$$0 < C_s < \frac{1 - \epsilon\frac{C_{\text{inv}}C_f}{C_\sigma}}{1 + \frac{C_{\text{inv}}C_f}{C_\sigma}} < \frac{1 - \frac{C_{\text{inv}}C_f}{C_\sigma}}{1 + \frac{C_{\text{inv}}C_f}{C_\sigma}} = \frac{C_\sigma - C_{\text{inv}}C_f}{C_\sigma + C_{\text{inv}}C_f}.$$

Therefore, in order for C_s to exist, C_σ must be taken sufficiently large, that is

$$C_\sigma > C_{\text{inv}}C_f, \tag{2.8.14}$$

in which case the second inequality in (2.8.13) automatically holds. As such, the method is coercive. □

2.9 Concluding Remarks

In this chapter we have demonstrated how the interior penalty DGFEM is derived for a second order linear scalar PDE. We also arrived at an analysis of the method to show existence and uniqueness (2.8.9) of the solution, with the main ideas and concepts that underpin the method. These ideas are extended to the case of the incompressible Navier-Stokes equations in Chapter 3, where the concept of coercivity is replaced by the weaker notion of inf-sup stability to accommodate the indefiniteness of the discrete linear system arising there.

The main tool in proving coercivity was the inverse estimate (2.8.5), which directly influences the choice of penalty parameter. In Chapter 5 we extend the discussion of the correct choice of penalty parameter, considering a elements with curved faces, as well as an improved estimate for the value that satisfies (2.8.14). It is through this discussion that a penalty parameter suitable for use with curved mesh elements is derived. We then provide an alternative approach to defining IP DGFEMs in Chapter 7, considering polytopic elements without the need for element transformations [38]. This requires the definition of new inverse inequalities suitable for general polytopic elements, thus altering the value of the discontinuity penalisation function.

Chapter 3

Discontinuous Galerkin Methods for Incompressible Laminar Flows

The Stokes system is traditionally used to model slow moving or highly viscous fluids [112], whilst the Navier-Stokes equations are used to model fast moving flows, such as those readily encountered in aerodynamic or aeroacoustic modelling. One key difference between the steady-state Navier-Stokes equations and the advection-diffusion problem studied in the previous chapter, is these are nonlinear equations. Discontinuous Galerkin methods have numerous advantages over conforming or classical finite element methods, especially in transport dominated problems due to their flexibility and stability. In this chapter, we introduce the steady-state incompressible Navier-Stokes equations for laminar flows, as well as deriving a discontinuous Galerkin discretisation for this system of PDEs.

A laminar, or streamline flow, is defined as one in which the fluid appears to move through the domain with no mixing between parallel flows or 'layers'. There are no cross-currents perpendicular to the flow direction, nor eddies, (a swirling of the fluid). In particular, laminar flows tend to be very orderly, with particles close to solid surfaces moving parallel to that surface. They tend to occur naturally at lower Reynolds numbers (see Definition 3.1.1 below), generally below a critical value of approximately $2,000$ for a "flow through a pipe" system. However, turbulent transition typically happens in the range of $1,800$ to $2,100$ [14]. Incompressible flows are those in which the material density is constant within an infinitesimal volume that moves with the flow velocity. An equivalent statement, is that the divergence of the flow velocity is zero. In general, an incompressible flow does not imply that the fluid itself is incompressible, just that the density remains constant within a defined volume. As such, under the correct conditions, incompressible flows can be used to give good approximations to the flow of compressible fluids.

3.1 The Incompressible Navier-Stokes Equations

Let Ω be defined as before, a bounded open Lipschitz domain in $\mathbb{R}^d$, $d = 2, 3$. We maintain the same notation in this chapter, letting Γ represent the boundary of the domain Ω.

Definition 3.1.1. The Reynolds number Re is the ratio of inertial forces to viscous forces within a fluid which is subjected to relative internal movement due to different fluid velocities:

$$Re = \frac{L_C u}{\nu},$$

where u is the velocity of the fluid with respect to the object/surface in contact with the flow, ν is the kinematic viscosity, and L_C is the characteristic linear dimension.

For two matrices, $\underline{\mathbf{A}}$, $c \times d$, and $\underline{\mathbf{B}}$, $s \times t$, the Kronecker product $\underline{\mathbf{A}} \otimes \underline{\mathbf{B}}$ is the $cs \times dt$ block matrix:

$$\underline{\mathbf{A}} \otimes \underline{\mathbf{B}} := \begin{bmatrix} a_{11}\underline{\mathbf{B}} & \cdots & a_{1d}\underline{\mathbf{B}} \\ \vdots & \ddots & \vdots \\ a_{c1}\underline{\mathbf{B}} & \cdots & a_{cd}\underline{\mathbf{B}} \end{bmatrix}.$$

The relative movement within the fluid generates fluid friction, which is a factor in developing a turbulent flow. The Reynolds number itself indicates at which fluid velocities a turbulent flow develops for a particular situation or domain. The non-dimensionalised steady-state incompressible Navier-Stokes equations read

$$\begin{aligned} -\frac{1}{Re}\nabla^2 \mathbf{u} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) + \nabla p &= \mathbf{0}, \\ \nabla \cdot \mathbf{u} &= 0, \end{aligned} \tag{3.1.1}$$

in Ω; here $\mathbf{u} \in \mathbb{R}^d$ is the flow velocity and p the pressure of the fluid. The first line in equation (3.1.1) expresses the conservation of momentum of the flow, whilst the second is the incompressibility condition which arises from the conservation of mass equations. In fluid dynamics, the divergence of the velocity vector measures the net flow out of a given point. For the velocity of a flow to have zero divergence, it means via the divergence theorem, that the amount of fluid entering a point or region is equal to the amount of fluid leaving that same region. That is to say, as before, the fluid does not compress. We note, that for the incompressible Navier-Stokes equations, the energy conservation equation, or as it is typically written, the temperature equation, is completely decoupled from (3.1.1) if the viscosity does not depend on the temperature. That is to say, we neglect the effect of heating due to viscous dissipation, and assume that the fluid is Newtonian. If information

about the temperature field is required, the energy equation may be solved separately using a different DGFEM discretisation. As this is not the focus of our work, and following the guidance of GE [93], we do not consider these results here. The notation $\mathbf{u}\otimes\mathbf{u}$ is used to represent the standard Kronecker product, with the flux defined such that $F := \mathbf{u}\otimes\mathbf{u}$.

We couple the Navier-Stokes system with the following boundary conditions [46], where $\mathbf{n}$ is the unit outward normal vector to the boundary

$$\mathbf{u} = g_{\mathrm{D}} \text{ on } \Gamma_{\mathrm{D}}, \tag{3.1.2}$$

$$\frac{1}{Re}\frac{\partial \mathbf{u}}{\partial \mathbf{n}} - p\mathbf{n} = 0 \text{ on } \Gamma_{\mathrm{N}}. \tag{3.1.3}$$

3.2 Discontinuous Galerkin Discretisation

We proceed similar to Section 2.6, considering a triangulation $\mathcal{T}_h$ of the domain Ω, defining the new discontinuous finite element spaces

$$\begin{aligned} \mathbf{V}_{h,m} &= \left\{ \mathbf{v} \in \left[L^2(\Omega)\right]^d : \mathbf{v}\,|_\kappa \in \left[\mathcal{R}^m(\kappa)\right]^d, \kappa \in \mathcal{T}_h \right\}, \\ Q_{h,m} &= \left\{ q \in L^2(\Omega) : q\,|_\kappa \in \mathcal{R}^{m-1}(\kappa), \kappa \in \mathcal{T}_h \right\}, \end{aligned} \tag{3.2.1}$$

$d = 2, 3$, $m \geq 1$, with $\mathbf{V}_{h,m}$ related to the velocity approximation and $Q_{h,m}$ related to the pressure. We also define the jumps and averages for both vector and tensor valued functions. Let q, $\mathbf{v}$, $\underline{\boldsymbol{\tau}}$ represent scalar, vector and matrix valued functions respectively, which are sufficiently smooth inside each element $\kappa^{\pm}$. Then let $(q^{\pm}, \mathbf{v}^{\pm}, \underline{\boldsymbol{\tau}}^{\pm})$ denote the traces of $(q, \mathbf{v}, \underline{\boldsymbol{\tau}})$ on an interior face $f = \partial\kappa^+ \cap \partial\kappa^- \subset \Gamma_{\mathrm{int}}$, taken from within the interior of $\kappa^{\pm}$. Then we define the following averages and jumps

$$\langle q \rangle = \frac{(q^+ + q^-)}{2}, \qquad \langle \mathbf{v} \rangle = \frac{(\mathbf{v}^+ + \mathbf{v}^-)}{2}, \qquad \langle \underline{\boldsymbol{\tau}} \rangle = \frac{(\underline{\boldsymbol{\tau}}^+ + \underline{\boldsymbol{\tau}}^-)}{2},$$

$$\begin{aligned} [q] &= q^+\mathbf{n}_{\kappa^+} + q^-\mathbf{n}_{\kappa^-}, & [\mathbf{v}] &= \mathbf{v}^+ \cdot \mathbf{n}_{\kappa^+} + \mathbf{v}^- \cdot \mathbf{n}_{\kappa^-}, \\ [\underline{\mathbf{v}}] &= \mathbf{v}^+ \otimes \mathbf{n}_{\kappa^+} + \mathbf{v}^- \otimes \mathbf{n}_{\kappa^-}, & [\underline{\boldsymbol{\tau}}] &= \underline{\boldsymbol{\tau}}^+\mathbf{n}_{\kappa^+} + \underline{\boldsymbol{\tau}}^-\mathbf{n}_{\kappa^-}. \end{aligned}$$

On boundary edges $f \in \Gamma$, we set $\langle q \rangle = q$, $\langle \mathbf{v} \rangle = \mathbf{v}$, $\langle \underline{\boldsymbol{\tau}} \rangle = \underline{\boldsymbol{\tau}}$, $[q] = q\mathbf{n}$, $[\mathbf{v}] = \mathbf{v} \cdot \mathbf{n}$, $[\underline{\mathbf{v}}] = \mathbf{v} \otimes \mathbf{n}$, and $[\underline{\boldsymbol{\tau}}] = \underline{\boldsymbol{\tau}}\mathbf{n}$, where $\mathbf{n}$ is the unit outward normal vector to the boundary Γ. Next, we introduce a suitably smooth test function $\mathbf{v}$, multiplying (3.1.1) through by $\mathbf{v}$ and integrating by parts. The derivation of the weak form here, follows in the same vein as the scalar mixed-type PDE considered in Chapter 2. Define the following bilinear forms

$A_h(\mathbf{u},\mathbf{v})$ and $B_h(\mathbf{u},\mathbf{v})$, along with nonlinear form $C_h(\mathbf{u},\mathbf{v})$ [46]:

$$\begin{aligned}
A_h(\mathbf{u},\mathbf{v}) &= \frac{1}{Re}\left(\int_\Omega \nabla_h\mathbf{u} : \nabla_h\mathbf{v}\,\mathrm{d}\mathbf{x} - \int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}} \left(\langle\nabla_h\mathbf{v}\rangle : [\mathbf{u}] + \langle\nabla_h\mathbf{u}\rangle : [\mathbf{v}]\right)\mathrm{d}s \right.\\
&\quad \left. + \int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}} \sigma\,[\mathbf{u}]\,[\mathbf{v}]\,\mathrm{d}s\right),\\
B_h(\mathbf{v},q) &= -\int_\Omega q\nabla_h\cdot\mathbf{v}\,\mathrm{d}\mathbf{x} + \int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}} \langle q\rangle\,[\mathbf{v}]\,\mathrm{d}s,\\
C_h(\mathbf{u},\mathbf{v}) &= -\int_\Omega F(\mathbf{u}) : \nabla_h\mathbf{v}\,\mathrm{d}\mathbf{x} + \int_{\Gamma_{\mathrm{int}}} \mathcal{H}\left(\mathbf{u}^+,\mathbf{u}^-,\mathbf{n}\right)[\mathbf{v}]\,\mathrm{d}s\\
&\quad + \int_\Gamma \mathcal{H}\left(\mathbf{u}^+,\mathbf{u}_{\boldsymbol{\Gamma}}\left(\mathbf{u}^+\right),\mathbf{n}\right)[v]\,\mathrm{d}s.
\end{aligned} \tag{3.2.2}$$

for $\mathbf{u},\mathbf{v} \in \mathbf{V}_{h,m}$ and $q \in Q_{h,m}$. In this setting, " : " denotes the usual colon product, such that for matrices $\underline{\mathbf{A}} = \sum_i \mathbf{a}_i\mathbf{b}_i$ and $\underline{\mathbf{B}} = \sum_j \mathbf{c}_j\mathbf{d}_j$, where $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{d}$ are vectors, $\underline{\mathbf{A}} : \underline{\mathbf{B}} := \sum_i\sum_j (\mathbf{a}_i\cdot\mathbf{d}_j)(\mathbf{b}_i\cdot\mathbf{c}_j)$. F is the flux defined in Section 3.1. Following [46], we make use of the Lax-Friedrichs flux to approximate the nonlinear convective terms:

$$\mathcal{H}(\mathbf{v},\mathbf{w},\mathbf{n}) := \frac{1}{2}\left(F(\mathbf{v})\cdot\mathbf{n} + F(\mathbf{w})\cdot\mathbf{n} - \alpha(\mathbf{w}-\mathbf{v})\right).$$

with $\alpha := \max(\phi^+,\phi^-)$, whereby ϕ^+ and ϕ^- are the largest eigenvalues in absolute magnitude of the Jacobi matrices $\frac{\partial}{\partial\mathbf{u}}(F(\cdot)\cdot\mathbf{n})$ computed at $\mathbf{v}$ and $\mathbf{w}$, respectively [46]. The boundary function in the Lax-Friedrichs flux is dependent upon the boundary it is computed on: $\mathbf{u}_\Gamma(\mathbf{u}) = g_{\mathrm{D}}$ on a Dirichlet boundary and $\mathbf{u}_\Gamma(\mathbf{u}) = \mathbf{u}^+$ on a Neumann boundary. In the case of flows which feature shocks or sharp changes in gradient, such as those that arise in the compressible setting, the Lax-Friedrichs flux can be considered to be too dissipative in certain regimes, and could be replaced with another, more suitable choice, e.g., the Vijayasundaram flux [61]. However, since we are considering the incompressible Navier-Stokes equations, this choice of numerical flux is sufficient. Nonetheless, any other stable flux can be used instead in what follows.

Next, we set $\sigma = C_\sigma m^2/h$, where h is the local mesh size and the constant $C_\sigma > 0$ is chosen independently of the polynomial degree m and the mesh size. The exact value of σ is considered in more detail in Chapter 5, where we discuss any potential flexibility in the choice of the discontinuity-penalisation function σ. For now, the choice of σ is selected inline with the literature for isotropic meshes [70].

Finally, we define

$$l_1(\mathbf{v}) = -\frac{1}{Re}\int_{\Gamma_D} ((g_D \otimes \mathbf{n}) : \nabla_h \mathbf{v} - \sigma g_D \cdot \mathbf{v})\, ds,$$
$$l_2(q) = \int_{\Gamma_D} q g_D \cdot \mathbf{n} ds.$$

The DGFEM then reads: find $(\mathbf{u}_h, p_h) \in \mathbf{V}_{h,m} \times Q_{h,m}$ such that for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_{h,m} \times Q_{h,m}$:

$$\begin{aligned} A_h(\mathbf{u}_h, \mathbf{v}_h) + B_h(\mathbf{v}_h, p_h) + C_h(\mathbf{u}_h, \mathbf{v}_h) &= l_1(\mathbf{v}_h), \\ B_h(\mathbf{u}_h, q_h) &= l_2(q_h). \end{aligned} \tag{3.2.3}$$

The only decision that remains is the choice of suitable polynomial degrees for our approximations to the velocity $\mathbf{u}$, and the pressure p. We wish to choose the degrees in such a way as to ensure that the proposed DGFEM is both stable and also converges to the correct solution. This brings around the notion of stable pairs, choices of the finite element spaces that represent the variables $\mathbf{u}$ and p such that the inf-sup condition is satisfied. Arbitrary choices for these spaces often yield instabilities, and indeed only special choices are suitable. It is possible to create a stable method whilst violating the inf-sup condition, however this requires special adjustments through the application of altered pressure projections [54].

3.3 The Inf-Sup Condition

The inf-sup condition was proposed by Babuška [17] and Brezzi [35], as a generalisation of the conditions for the well-posedness of a finite element problem. For the finite element spaces (3.2.1), the inf-sup condition holds if there exists a constant $\beta > 0$, such that for every $\mathbf{v} \in \mathbf{V}_{h,m}$ and for every $p \in Q_{h,m}$, the following condition holds

$$\inf_{p \in Q_{h,m}} \sup_{\mathbf{v} \in \mathbf{V}_{h,m}} \frac{B_h(\mathbf{v}, p)}{\|\mathbf{v}\|_{\mathbf{V}_{h,m}} \|p\|_{Q_{h,m}}} \geq \beta; \tag{3.3.1}$$

β is independent of $h, m, \mathbf{v}, p$. This implies that the choice of approximation spaces for the velocity and pressure are dependent upon one another. In the remainder of this work, we choose the finite element spaces such that the velocity $\mathbf{u}$ is approximated by second order piece-wise continuous polynomials, whilst the pressure p by piecewise linear functions.

Before proceeding to solve (3.2.3), we first need to know that a solution exists, as well as whether the method converges to a unique solution. In order to do this, consider for the

moment the simplified version of (3.2.3), where $l_2(q_h)$ is set to zero, i.e. $g_D = 0$. Thus, we have

$$\begin{aligned} \overline{A_h}(\mathbf{u}_h, \mathbf{v}_h) + B_h(\mathbf{v}_h, p_h) &= l_1(\mathbf{v}_h), \\ B_h(\mathbf{u}_h, q_h) &= 0, \end{aligned} \tag{3.3.2}$$

where $\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h) = A_h(\mathbf{u}_h, \mathbf{v}_h) + C_h(\mathbf{u}_h, \mathbf{v}_h)$. $\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h)$ is nonlinear; however in practical computations, for each iteration of the Newton solver we consider a linear perturbation of $\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h)$ and then proceed with the standard analysis used for the Stokes problem. However, for the purposes of the theory, we proceed instead by preserving the nonlinearity of $\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h)$, following Girault and Raviart [72]. As before, given l_1, we seek $(\mathbf{u}_h, p_h) \in \mathbf{V}_{h,m} \times Q_{h,m}$ such that equation (3.3.2) is satisfied.

First we introduce the following linear operators $A(\mathbf{w}) \in L(\mathbf{V}; \mathbf{V}')$ for $\mathbf{w} \in \mathbf{V}$, and $B \in L(\mathbf{V}; Q')$, where $\mathbf{V}'$, Q' denote the corresponding dual spaces, such that $\forall \mathbf{u}_h, \mathbf{v}_h \in \mathbf{V}_{h,m}$ and $q_h \in Q_{h,m}$

$$\begin{aligned} \langle A(\mathbf{w})\mathbf{u}_h, \mathbf{v}_h \rangle &= \overline{A_h}(\mathbf{u}_h, \mathbf{v}_h), \\ \langle B\mathbf{v}_h, q_h \rangle &= B_h(\mathbf{v}_h, q_h). \end{aligned}$$

Then, we can rewrite problem (3.3.2) using the following notation: find $(\mathbf{u}_h, p_h) \in \mathbf{V}_{h,m} \times Q_{h,m}$ such that

$$\begin{aligned} A(\mathbf{u}_h)\mathbf{u}_h + B'p_h &= l_1, \\ B\mathbf{u}_h &= 0. \end{aligned} \tag{3.3.3}$$

Proceeding as we would if problem (3.3.2) was linear, we set $\mathbf{V}_{h,m} = \mathrm{Ker}_{\mathrm{df}}(B_h)$, the kernel of the divergence free vector field, and associate the following problem with (3.3.2). Find $\mathbf{u}_h \in \mathbf{V}_{h,m}$ such that $\forall \mathbf{v}_h \in \mathbf{V}_{h,m}$

$$\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h) = l_1(\mathbf{v}_h). \tag{3.3.4}$$

One final piece of notation is the L_2-orthogonal projection of the bilinear form. Consider the linear operator $\pi \in L(\mathbf{V}; \mathbf{V}')$ which is defined by $\langle \pi l_1, \mathbf{v}_h \rangle = \langle l_1, \mathbf{v}_h \rangle$, $\forall \mathbf{v}_h \in \mathbf{V}_{h,m}$. This allows us to rewrite problem (3.3.4) in the following equivalent form in $\mathbf{V}'$, where $\mathbf{V}'$ is the subspace of $\mathbf{V}$

$$\pi A(\mathbf{u}_h)\mathbf{u}_h = \pi l_1.$$

Clearly if $(\mathbf{u}_h, p_h)$ is a solution of (3.3.2), then $\mathbf{u}_h$ is a solution of (3.3.4). The converse of this can be proved as in the linear case, so long as the inf-sup condition holds [72].

Therefore, the only difficulty that remains here is finding a solution to the nonlinear problem (3.3.4). To begin to address this issue, we proceed by stating the classical fixed-point theorem of Brouwer.

Theorem 3.3.1. *Let Y denote a non-empty, convex and compact subset of a finite-dimensional space and let F be a continuous mapping from Y into Y. Then, F has at least one fixed point.*

Following on from this, we consider the corollary set out in [72].

Corollary 3.3.2. *Let $\mathcal{H}$ be a finite-dimensional Hilbert space whose scalar product is denoted by $(\cdot,\cdot)$ and the corresponding norm by $|\cdot|$. Let Φ be a continuous mapping from $\mathcal{H}$ into $\mathcal{H}$ with the following property:*

there exists $\mu > 0$ such that $(\Phi(f), f) \geq 0$ for all $f \in \mathcal{H}$ with $|f| = \mu$.

Then, there exists an element f in $\mathcal{H}$ such that $\Phi(f) = 0$, $|f| \leq \mu$.

We also need the following existence result for the solution to (3.3.3).

Theorem 3.3.3. *Assume that the following hypotheses hold:*

1) *there exists a constant $\alpha > 0$ such that $\overline{A_h}(\mathbf{v}_h, \mathbf{v}_h) \geq \alpha \|v\|_{\mathbf{V}}$, $\forall \mathbf{v}_h \in \mathbf{V}_{h,m}$.*

2) *the space V is separable and, for all $v \in \mathbf{V}_{h,m}$, the mapping $\mathbf{u}_h \to \overline{A_h}(\mathbf{u}_h, \mathbf{v}_h)$ is sequentially weakly continuous on $\mathbf{V}$, i.e.,*
   weak $\lim_{t\to\infty} u_t = u$ in $\mathbf{V}$ implies $\lim_{t\to\infty} \overline{A_h}(\mathbf{u}_t, \mathbf{v}_h) = \overline{A_h}(\mathbf{u}_h, \mathbf{v}_h)$, $\forall \mathbf{v}_h \in \mathbf{V}_{h,m}$.

Then, problem (3.3.4) *has at least one solution $\mathbf{u}_h \in \mathbf{V}_{h,m}$.*

Proof. Omitted here for conciseness, but available in [72]. □

Then, as $l_1 \in \mathbf{V}'$, there exists at least one element $\mathbf{u}_h \in \mathbf{V}_{h,m}$ such that $\overline{A_h}(\mathbf{u}_h, \mathbf{v}_h) = l_1(\mathbf{v}_h)$, for every $\mathbf{v}_h \in \mathbf{V}_{h,m}$. Therefore the above theorem shows us that a solution to (3.3.4) exists, so we now turn our attention to the uniqueness of this solution.

Theorem 3.3.4. *Assume that the following hold:*

1) *The bilinear form* $\overline{A_h}(\mathbf{w};\mathbf{u}_h,\mathbf{v}_h)$ *is uniformly* V*-elliptic with respect to* $\mathbf{w}$*, i.e., there exists a constant* $\alpha>0$ *such that* $\overline{A_h}(\mathbf{w};\mathbf{u}_h,\mathbf{v}_h)\geq\alpha\|\mathbf{v}_h\|^2_{\mathbf{V}_{h,m}}$, $\forall\mathbf{v}_h,\mathbf{w}\in\mathbf{V}_{h,m}$.

2) *The mapping* $w\to\pi A(\mathbf{w})$ *is locally Lipschitz-continuous in* $\mathbf{V}$*, i.e., there exists a continuous and monotonically increasing function* $L:\mathbb{R}_+\to\mathbb{R}_+$ *such that for all* $q_h>0$

$$\left|\overline{A_h}(\mathbf{w}_1;\mathbf{u}_h,\mathbf{v}_h)-\overline{A_h}(\mathbf{w}_2;\mathbf{u}_h,\mathbf{v}_h)\right|\leq L(q_h)\|\mathbf{u}_h\|_{\mathbf{V}_{h,m}}\|\mathbf{v}_h\|_{\mathbf{V}_{h,m}}\|\mathbf{w}_1-\mathbf{w}_2\|_{\mathbf{V}_{h,m}}$$

$\forall\mathbf{u}_h,\mathbf{v}_h\in\mathbf{V}_{h,m}$ *and* $\forall\mathbf{w}_1,\mathbf{w}_2\in\mathbf{S}_{q_h}=\left\{\mathbf{w}\in\mathbf{V}_{h,m};\|w\|_{\mathbf{V}_{h,m}}\leq q_h\right\}$.

Then, under the condition $\left[\|\pi l\|_{\mathbf{V}'}/\alpha^2\right]L\left(\|\pi l\|_{\mathbf{V}'}/\alpha\right)<1$*, problem* (3.3.3) *has a unique solution* $\mathbf{u}_h\in\mathbf{V}_{h,m}$.

Proof. For the proof see [72]. □

Then all that remains is to solve problem (3.3.2) in the same fashion as we would for the linear case.

Theorem 3.3.5. *Assume that the bilinear form* $B_h(\cdot,\cdot)$ *satisfies the inf-sup condition*

$$\inf_{p_h\in Q_{h,m}}\sup_{\mathbf{v}_h\in\mathbf{V}_{h,m}}\frac{B_h(\mathbf{v}_h,p_h)}{\|v\|_{\mathbf{V}_{h,m}}\|p\|_{Q_{h,m}}}\geq\beta>0$$

Then for each solution $\mathbf{u}_h$ *of* (3.3.3)*, there exists a unique* $p_h\in Q_{h,m}$ *such that the pair* $(\mathbf{u}_h,p_h)$ *is a solution of* (3.3.2).

Proof. Assume that $\mathbf{u}_h\in\mathbf{V}_{h,m}$ is a solution of (3.3.4). Then we must find $p_h\in Q_{h,m}$ which satisfies (3.3.3). However, $l_1-A(\mathbf{u}_h)\mathbf{u}_h$ belongs to the polar set V^0 of V. Additionally, the inf-sup condition implies that B' is an isomorphism from $Q_{h,m}$ onto V^0. Hence there exists a unique $p_h\in Q_{h,m}$ such that $(\mathbf{u}_h,p_h)$ is a solution of (3.3.3), and, in turn, of (3.3.2). □

3.4 Computing Numerical Solutions

It is necessary to employ specialised software libraries to construct the various data-structures associated with DGFEM discretisation of PDEs, as well as to compute the numerical solution in an efficient manner. In this section we provide details on the creation of computational meshes, the construction of the mass matrix and right-hand side vectors, along with a method of computing numerical solutions for systems of PDEs.

In this work, we employ the AptoFEM package created by Houston, Hall, Antonietti, Giani and Krahl [88]. It is a general purpose finite element software toolkit with a strong emphasis on applications arising in CFD. In particular, non-conforming DGFEM and standard conforming FEM can be used, along with *a posteriori* error estimation and *hp*-adaptivity. Also, a variety of structured and unstructured grids in several different file formats may be read in for mesh construction. Specifically for this work, routines have been written to compute the matrices and nonlinear functions associated with the DGFEM for incompressible laminar and turbulent flows. Further, we have developed the routines required for the DWR method detailed in Chapter 6, supplemented with pre-existing *a posteriori* error estimation routines, along with a new routine to compute the proposed penalty parameter in Chapter 5. In total, around $30,000$ lines of code have been written in order to produce the numerical results found within this work. A number of other supplementary routines were created within the MATLAB programing environment to perform tasks such as calculating cubic splines for the domain boundaries, and removing duplicate nodes from computational meshes. These tasks only needed to be performed once, so the ease of implementation associated with languages such as MATLAB was chosen over the increased speed of execution associated with compiled languages such as Fortran and C++. In total, around 30 different MATLAB routines were written to supplement the Fortran 90 implementation of AptoFEM.

3.4.1 Mesh Generation

Various meshing programs exist and are suitable for the creation of industrial type meshes. However, to ensure that the mesh density is acceptable across the entire domain, a manual blocking approach is undertaken, employing the ANSYS ICEM CFD software package. In this manner, we are able to finely control the particular details of the mesh, allowing or disallowing hanging-nodes as required. The meshes for the numerical examples found in Chapters 5, 6, 7 were created in this manner, whilst the numerical examples 3.5.1 and 3.5.3 were produced using mesh generation routines written within the AptoFEM framework.

All meshes read into the AptoFEM solver are stored in a tree structure. More precisely, two-dimensional meshes are stored in two separate trees, one containing the mesh elements,

and another containing the mesh faces. In the case of three-dimensional meshes, a third tree is created to store the mesh edges. Extraction of the computational mesh from this data structure is carried out by means of a "tree walk." The process begins by considering the first node in the first level of the tree structure. If no children nodes are present, then this node must be part of the computational mesh. In the case when there are children associated with a particular node, each child is considered sequentially and levels descended until there are no further children and the nodes are terminal. In this fashion, mesh refinement and coarsening is handled seamlessly, creating new branches in the tree for refined elements, and stepping back up levels to coarsen. However, it should be noted that this system does not allow for coarsening beyond the initial mesh, so extra considerations must be made during the design of the mesh.

We remark, that when we define the boundary conditions of a domain, we must consider the point where two different boundaries meet. The discretisation (3.2.2) is an integral form which may omit finitely mainly points from the computation. Therefore, in theory, we automatically avoid any concerns associated with the definition of such a point. However, in practice we must ensure that the numerical solver only computes one value for the solution at these points.

3.4.2 Quadrature

Numerical integration of functions is carried out by way of quadrature rules. As such, each type of mesh element, quadrilateral or simplex, requires its own quadrature defined upon it. More precisely, a set of points, x_q, and weights w_q such that for a function f,

$$\int_\kappa f \, \mathrm{d}\mathbf{x} \approx \sum_q f\left(x_q\right) w_q.$$

Therefore, quadrature routines for each element are all based on performing tensor product manipulations of the one-dimensional quadrature on the interval $[-1, 1]$. In particular, consider the set of one-dimensional quadrature points ρ_i, $i = 1, ..., n$, with a set of corresponding weights w_i. Then, for a quadrilateral element, the set of points and weights are given by

$$\{x, w\}_{ij} = \left\{\left(\rho_i, \rho_j\right), w_i, w_j\right\}.$$

For a simplex element, the quadrature points and weights are found by mapping the ref-

erence quadrilateral to the reference simplex via the mapping

$$\begin{aligned} x^S =& \frac{\left(1+x^Q\right)\left(1-y^Q\right)}{2} - 1, \\ y^S =& y^Q, \end{aligned}$$

where $\left(x^Q, y^Q\right)$ are the coordinates of the quadrature points in the quadrilateral reference frame, and $\left(x^S, y^S\right)$ are the coordinates in the simplex reference frame. The weights associated with the simplex reference frame are found by way of multiplication of the quadrilateral weights by the Jacobian of the above mapping at the corresponding coordinates. We note that this mapping collapses the line $y^S = 1$ to a single point, and is therefore singular. However, if the one-dimensional quadrature does not include the endpoints of the interval, quadrature on a triangle is well defined. Analogous arguments can be performed for three-dimensional elements to construct quadratures. The particular implementation we use in this work is Gauss quadrature, where the quadrature points are the roots of the Legendre polynomials.

3.4.3 Iterative Solvers

Unlike the advection-diffusion problem studied in the previous chapter, the incompressible Navier-Stokes equations contain nonlinear terms in their weak form, and as such require iterative techniques to find a solution. Specifically, we use a damped Newton method. Consider the following nonlinear problem

$$N\left(\mathbf{u}_h\right) = 0. \tag{3.4.1}$$

This is representative of the nonlinear problem in (3.2.3). Given an initial estimate $\mathbf{u}_h^0$ of the numerical solution $\mathbf{u}_h$, we are able to generate a sequence of progressively better approximations $\mathbf{u}_h^n$, $n = 0, 1, ...$, to the numerical solution. Following [82], for a fixed $\mathbf{v}_h \in \mathbf{V}_{h,m}$, we take the Fréchet derivative $\mathbf{d}_h^n \in \mathbf{V}_{h,m}$ of (3.4.1) at $\mathbf{u}_h^n$, such that

$$\mathbf{d}_h^n = -J_F^{-1}\left(N\left(\mathbf{u}_h^n\right)\right)$$

This, along with a corresponding damping parameter θ^n, provides the solution update values for the sequence. In particular

$$\mathbf{u}_h^{n+1} = \mathbf{u}_h^n + \theta^n \mathbf{d}_h^n.$$

θ^n is dynamically chosen to ensure that the discrete L^2-norm of the residual computed

with $\mathbf{u}_h^{n+1}$ is less than the same quantity computed with $\mathbf{u}_h^n$. The efficient implementation of this method requires solving a system of linear equations in order to find the update value $\mathbf{d}_h^n$. Whilst a direct solver software package such as MUMPS (Multifrontal Massively Parallel Sparse Direct Solver) [3] may be effective for solving systems which are not too large, we now discuss the structure of these matrices, as well as, consider an alternative approach suitable for full scale industrial problems.

3.4.4 Construction of DGFEM Data Structures

The matrices associated with DGFEM discretisations are often very large due to the numbers of degrees of freedom associated with each element. Fortunately, the matrices are also very sparse naturally and exhibit a block-type structure; characterised by non-zero blocks along the diagonals. In particular, the blocks on the central diagonal describe the interaction of element degrees of freedom with itself, and off-diagonal blocks describe interactions across mesh faces. In the case of nonlinear problems, especially those which involve turbulence modelling, the Jacobian of the mass-matrix becomes increasingly less sparse and more difficult to work with. As such, to avoid problems with computer memory, AptoFEM stores these main data structures in Compressed Sparse Row format.

3.4.5 Linear Solvers and Preconditioning

Consider the system of linear algebraic equations:

$$\underline{\mathbf{A}}\mathbf{u} = \mathbf{f}, \tag{3.4.2}$$

where $\mathbf{u}$ is the solution vector, $\underline{\mathbf{A}}$ is the matrix of the linear system, and $\mathbf{f}$ is the right-hand side vector.

Iterative solvers such as the generalised minimal residual method (GMRES) are employed to handle (3.4.2) when $\underline{\mathbf{A}}$ has a large number of degrees of freedom. To improve the rate of convergence of iterative solvers, we introduce the preconditioning matrix $\underline{\mathbf{P}}$, such that the matrix $\underline{\mathbf{P}}^{-1}\underline{\mathbf{A}}$ has a smaller condition number than that of $\underline{\mathbf{A}}$. Formally, the condition number is the maximum ratio of relative error in $\mathbf{u}$, divided by the relative error in $\mathbf{f}$. We now proceed to discuss the preconditioner that is most appropriate for our system. We note that other preconditioners may also be suitable; however, we omit there discussion and refer to [58].

3.4.5.1 Incomplete LU Factorisation

The Incomplete LU Factorisation (ILU) method is an approximation to traditional LU factorisation for matrices. The matrix $\underline{\mathbf{A}}$ must be sparse, and we seek the lower triangular matrix $\underline{\mathbf{L}}$, and upper triangular matrix $\underline{\mathbf{U}}$ such that $\underline{\mathbf{A}} \approx \underline{\mathbf{LU}}$. The solution to $\underline{\mathbf{LU}}\mathbf{u} = \mathbf{f}$ can be computed quickly using forward and backward substitution; however, this does not yield the exact solution to (3.4.2). This is because we wish to preserve the sparsity pattern of matrix $\underline{\mathbf{A}}$ in the triangular matrices $\underline{\mathbf{L}}$ and $\underline{\mathbf{U}}$, rather than adding extra entries to them to ensure $\underline{\mathbf{A}} = \underline{\mathbf{LU}}$. This is known as the ILU(0) preconditioner. However, we may wish to reflect the underlying structure of higher order matrices, thus other preconditioners known generally as ILU(k) exist, which share the structure of the matrix $\underline{\mathbf{A}}^{k+1}$. The factorisations ILU(k), become steadily more accurate and less sparse as k increases in value, but the trade off between computation time and number of iterations for the iterative solver to converge becomes less and less favourable. Hence a balance must be struck when choosing the value of k, and this is quite often chosen based upon the linear system that needs to be solved. In all circumstances however, we let $\underline{\mathbf{P}} = \underline{\mathbf{LU}}$, and use $\underline{\mathbf{P}}$ as a preconditioner for other iterative methods such as the GMRES. The method for producing such a preconditioner is the same as the usual LU factorisation of matrix $\underline{\mathbf{A}}$, but every time a new nonzero entry appears in either $\underline{\mathbf{L}}$ or $\underline{\mathbf{U}}$ which is zero in the matrix of interest $\underline{\mathbf{A}}^{k+1}$, we disregard its value and write a zero in its place instead. This preserves the underlying structure of the matrix $\underline{\mathbf{A}}^{k+1}$.

3.5 Numerical Experiments

We present a number of numerical experiments to validate the effectiveness of the DGFEM. Initially, we simulate a low Reynolds number, laminar, two-dimensional channel flow in Section 3.5.1 to provide an early proof of concept for the proposed numerical solver.Then, in Section 3.5.2, we consider a backwards-facing step test case, comparing experimental data [9], with the results calculated using the DGFEM. Further, in Section 3.5.3, we produce simulations of the experimental results presented in [44, 56]. These initial simulations are rather simplistic, however, they do provide a good indication of the accuracy and effectiveness of the solver. Later, we progress to tackle more advanced test cases with suitable turbulence models. In particular, a high Reynolds number blade cascade in Chapter 5, as well as the flow around an aerofoil in Chapters 6 and 7.

The simulations presented in Sections 3.5.1, 3.5.2 and 3.5.3 use the usual setup of boundary conditions. The standard no-slip condition (3.1.2) in the form of the Dirichlet boundary is used to model the domain walls, as well as to prescribe the flow inlet conditions. Whereas

for the domain outlet, the stress-free outflow condition (3.1.3) is applied, as the final flow velocity and pressure are unknown prior to the calculation of the numerical solution. We note that the stress-free condition is only valid once the flow has fully developed, since it assumes a zero normal gradient for all flow variables except pressure. Therefore it is important that any constructed mesh has adequate space after any obstructions to ensure that the flow has time to properly recover before passing through the outlet boundary. This is of particular importance in Chapter 4 and beyond, since turbulent flows take longer to fully develop after passing an obstruction.

The following calculations are carried out as detailed in Section 3.4, using the AptoFEM software package, supplemented with MUMPS to solve the linear systems, and ParMETIS to partition the mesh and reorder the matrices.

3.5.1 Channel

All quantities in this numerical experiment are non-dimensional, as this is a purely theoretical test case, used primarily to identify any problems with the numerical solver. We consider comparisons to experimental data in Section 3.5.2. A such, we define a two-dimensional rectangular channel domain $\Omega \subset \mathbb{R}^2$ on the usual Cartesian plane, with a horizontal x-axis and a vertical y-axis. The channel is orientated parallel to the horizontal axis, channel length $L = 10$, and half width $R = 1$. We define the domain boundaries Γ, such that $\Gamma = \Gamma_D \cup \Gamma_N$. The inflow boundary, located at $x = 0$, is a Dirichlet boundary modelling Poiseuille flow with a peak velocity equal to 1. No-slip conditions are used for the channel walls positioned at $y = -1$ and $y = 1$, and the out-flow boundary, $x = 10$, is equipped with the stress-free Neumann condition. For reference, see Figure 3.5.1.

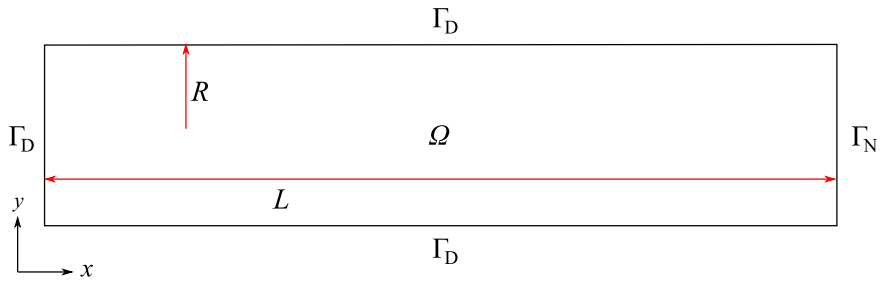

Figure 3.5.1: Channel geometry. $R = 1$, $L = 10$.

The domain is partitioned into a mesh, with 11 equally spaced nodes spanning the width, and 21 equally spaced nodes covering the length of the channel. The mesh $\mathcal{T}_h$, consisting of 200 uniform quadrilateral elements is shown in Figure 3.5.2.

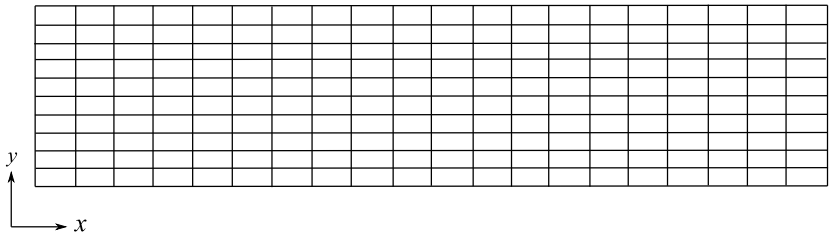

Figure 3.5.2: Uniform channel mesh, 21×11 nodes.

All variables in these initial experiments are fixed, except for the kinematic viscosity ν, which is varied in order to adjust the Reynolds number of the simulated flow. The Reynolds number is calculated using a characteristic length scale $L_C := R$, the channel half width. For low Reynolds number laminar flows, we wish to observe a constant flow speed down the length of the channel. The velocity is scaled according to the shear velocity with $u^+ = \frac{u}{u_\tau}$, where $u_\tau = \sqrt{\frac{\tau_w}{\rho}}$ is the shear velocity, and τ_w is the local wall shear stress.

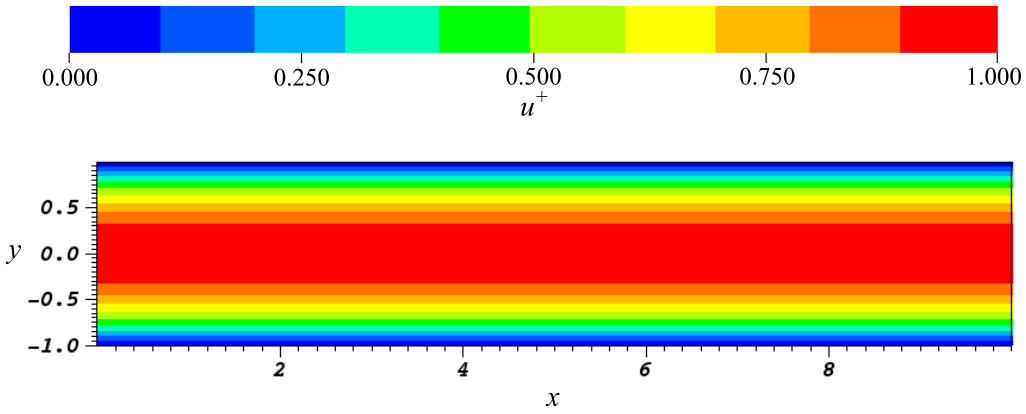

Figure 3.5.3: Colour plot of the scaled axial velocity u^+, $Re = 100$.

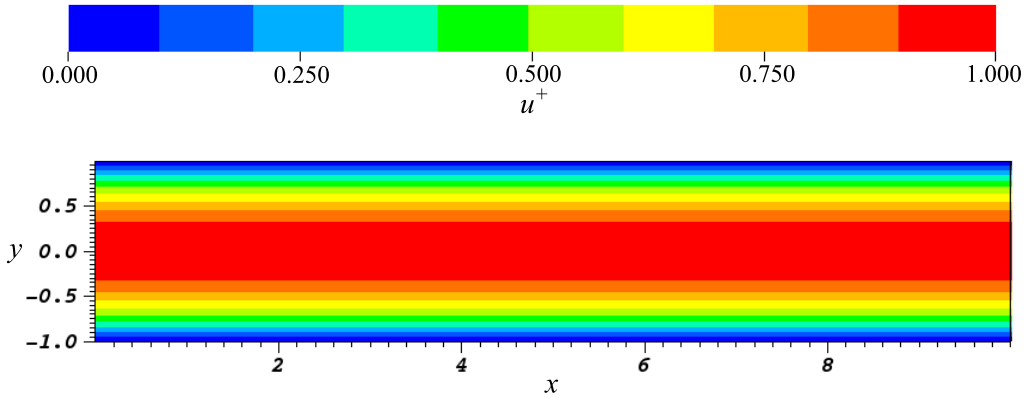

Figure 3.5.4: Colour plot of the scaled axial velocity u^+, $Re = 300$.

The preliminary results presented in Figures 3.5.3 and 3.5.4, show the scaled axial velocity of the numerical solution along the length of channel. As expected, the flow speed remains constant in the x-direction, denoted by coloured bands of constant width. This is shown in greater detail in Figure 3.5.5, where the solution in Figure 3.5.4 is sampled by way of a cross section of the flow at $x = 8$.

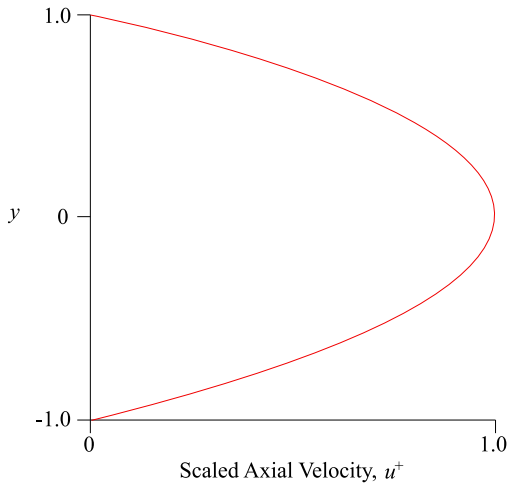

Figure 3.5.5: Cross section of the flow at $x = 8$, showing the scaled axial velocity profile u^+, $Re = 300$.

Since the flow is constant along the length of the channel, and Figure 3.5.5 mirrors the Poiseuille flow inlet conditions, we can be certain that the DGFEM solver is performing as expected for laminar flows in this simple geometry. We remark, that since we use quadratic basis functions to represent the velocity solution, it is indeed possible to compute Poiseuille

flow through a rectangular channel, by using a single element to cover the domain. In Section 3.5.2, we consider experimental results to more carefully compare the accuracy of DGFEM solver.

3.5.2 Backwards-Facing Step

The backwards facing step experiment is commonly used in the literature for the development and validation of numerical methods, due to the high level of understanding surrounding the flow features, as well as, the numerous accurate experimental readings that have been collected. We follow [73, 141], considering the experimental results collected in [9]. In particular, we consider a two dimensional test case for flows with $Re = 100$ and $Re = 389$. The data for these Reynolds numbers has been collected using the three dimensional apparatus detailed in [9]. We note, that the two-dimensional numerical solution should agree closely with the three-dimensional experiment, as long as, $Re \leq 400$. Beyond this value, it is postulated that a longitudinal wave develops in the three-dimensional case, causing the flow to deviate from the expected two-dimensional result [141].

We consider the domain $\Omega \subset \mathbb{R}^2$, shown in Figure 3.5.6, on the usual Cartesian plane, with a horizontal x-axis and a vertical y-axis. We introduce the inlet height $h = 5.2\,\text{mm}$, the inlet length $l = 5\,\text{mm}$, the step height $S = 4.9\,\text{mm}$, channel length $L = 300\,\text{mm}$ and the channel height $H = 10.1\,\text{mm}$.

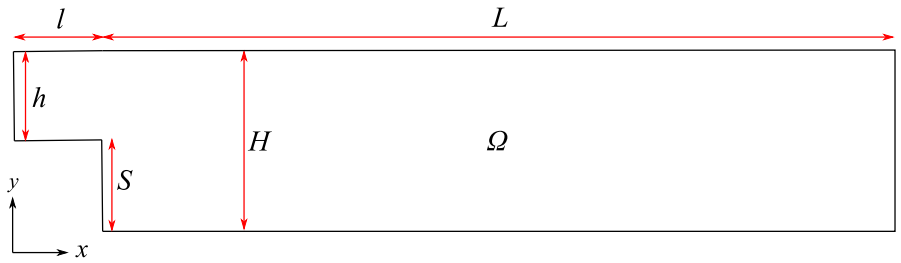

Figure 3.5.6: Backward-facing step geometry [141], $l = 5\,\text{mm}$, $h = 5.2\,\text{mm}$, $S = 4.9\,\text{mm}$, $L = 300\,\text{mm}$ and $H = 10.1\,\text{mm}$.

The boundaries of the domain are such that the inlet and walls use Dirichlet boundary conditions, with a Neumann boundary condition for the outlet. In particular, the domain inlet located on the left most edge of Figure 3.5.6, is simple Poiseuille flow. The domain outlet, the far right edge of Figure 3.5.6, is the stress free Neumann condition. Finally, all other edges are domain boundaries are walls, described using the no-slip Dirichlet boundary condition. Next, the domain is partitioned into the non-uniform mesh $\mathcal{T}_h$ shown in Figure

3.5.7, consisting of 6100 elements, with grading in the x and y directions about the step corner.

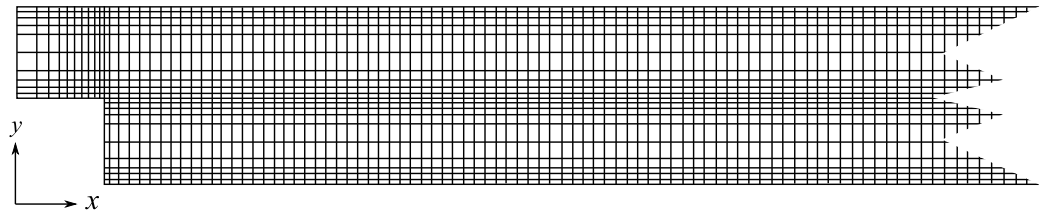

Figure 3.5.7: Two-dimensional non-uniform mesh $\mathcal{T}_h$, 6100 elements.

Following [9], the test cases for $Re = 100$ and $Re = 389$, use peak inlet velocities u_{peak}, of 21.1 cm/s and 88.0 cm/s, respectively. The Reynolds number is calculated using Definition 3.1.1, with $u = \frac{2}{3}u_{\text{peak}}$, and $L_C = 2h$, the hydraulic diameter of the inlet. The following numerical results were obtained using the proposed DGFEM solver.

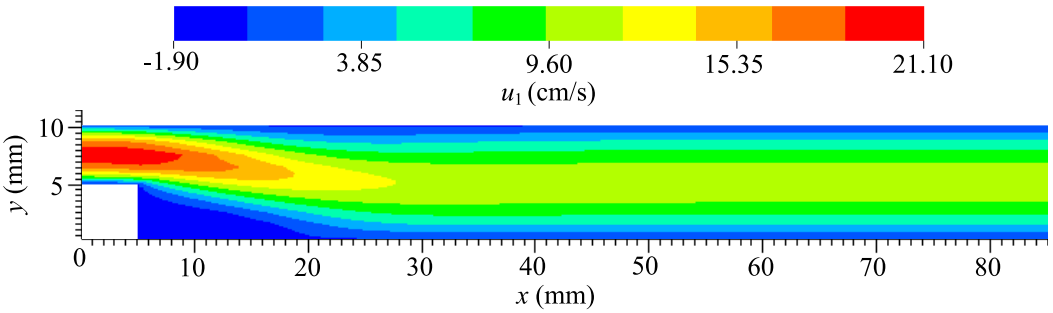

Figure 3.5.8: Colour plot of axial velocity u_1, $Re = 100$.

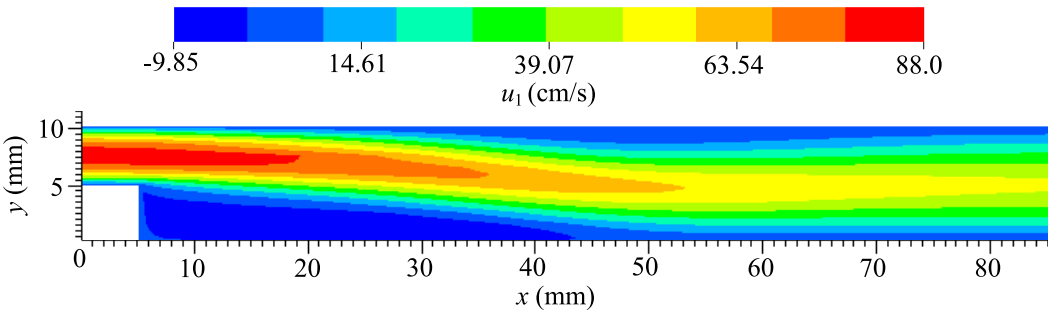

Figure 3.5.9: Colour plot of axial velocity u_1, $Re = 389$.

As we expected, the area of recirculation after the step on the lower wall becomes larger as we increase the Reynolds number; as shown in Figures 3.5.8 and 3.5.9. We limit the numerical experiments to $Re \leq 400$ in order to suppress the three-dimensional effects of the flow, but, as such, are unable to ascertain whether the DGFEM is able to predict the secondary recirculation area on the upper wall for larger Reynolds numbers. This area appears to be developing in Figure 3.5.9 at $x = 50\,\text{mm}$, which would agree with the findings in [73], which suggests that it is possible for laminar RANS simulations to capture this flow region. On the other hand, it is difficult for simulations that utilise RANS with turbulence modelling at these Reynolds numbers to do so. Therefore, in the interest of accuracy, so long as the Reynolds number is below the turbulent transitional value for the geometry, ($Re < 1200$ in this case,) then it is more accurate to use laminar simulations. However, this is a side note for our work, as we target Reynolds numbers in the region of $Re = 1 \times 10^6$ for industrial flow problems.

Further, we present a comparison between the axial velocity calculated by the DGFEM solver and the experimental results in [9]. In particular, the velocity is recorded at a series of stations x/S, once the flow has passed over the step edge. These stations are represented by a numerical value, equal to the distance from the step wall, divided by the step height S.

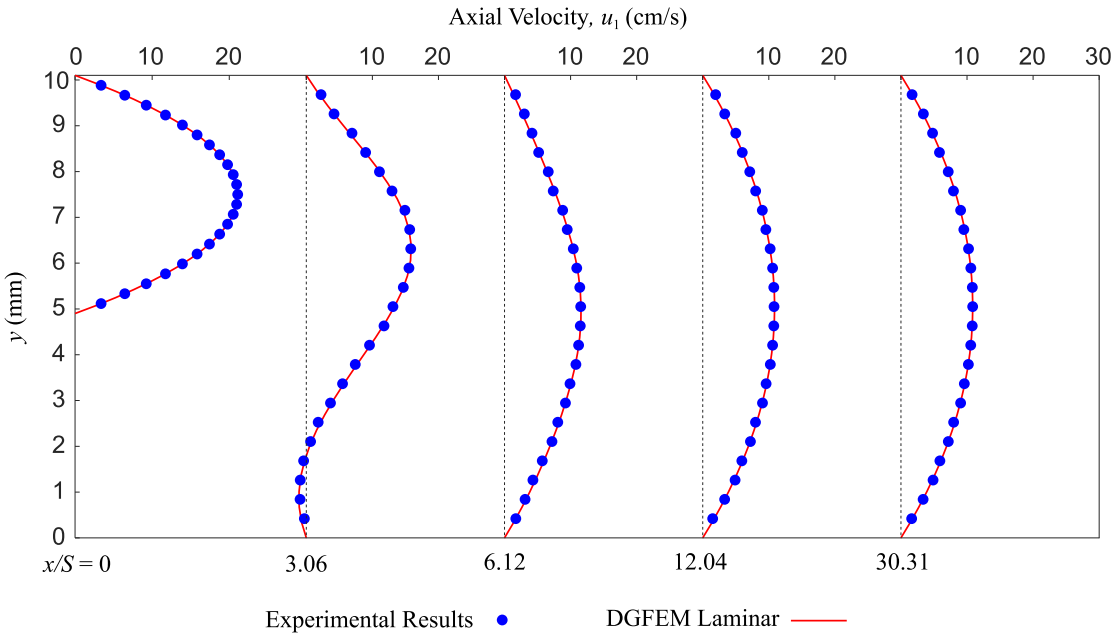

Figure 3.5.10: Comparison of experimental [9] and numerical, axial velocity profiles, $Re = 100$.

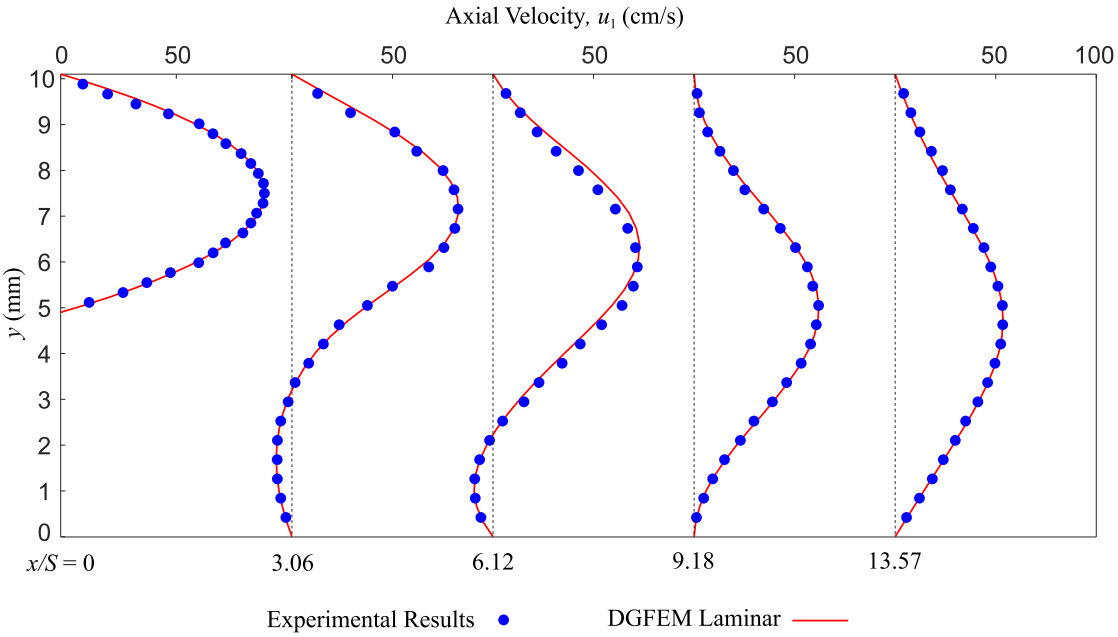

Figure 3.5.11: Comparison of experimental [9] and numerical, axial velocity profiles, $Re = 389$.

The numerical solution in Figure 3.5.10 agrees remarkably well with the experimental data for $Re = 100$, capturing the recirculation velocity at $x/S = 3.06$ accurately. There was a concern regarding the Lax-Friedrichs numerical flux being too dissipative to capture these flow features well, but this does not seem to be the case at these low Reynolds numbers.

Figure 3.5.11 shows the second test case with $Re = 389$. Again, the DGFEM solution approximates the experimental data well, accurately capturing the recirculation region on the lower wall. However, there is some disagreement with the data for $x/S = 6.12$, with the peak velocity predicted to be higher up in the outlet channel following the step. This also appears to be the case in the literature [73], with laminar RANS simulations exhibiting the same problem. This could be due to the three-dimensional effects of the flow beginning to take hold as the Reynolds number approaches the agreed upon value, $Re = 400$ [141]. However, since the DGFEM predicts the axial velocities for $x/S = 9.18$ and $x/S = 13.57$ well following this discrepancy, we can be sure that the DGFEM is performing as expected for laminar flows.

3.5.3 Channel with a Sudden Expansion

Following the success of the DGFEM solver in Section 3.5.2, we now explore one final laminar test case, with the aim of adjusting the solver parameters to find different physical solutions. Typically, in experiments involving symmetric domains, a symmetric flow solution is stable only for specific Reynolds numbers. Whereas, numerical solvers tend to consistently find these unstable symmetric solutions for a wide range of Reynolds numbers. As such, in this test case, we show that using the proposed DGFEM, we are able to find the non-symmetric solution if it is required. Following [44, 56], we have identified that a non-symmetric flow through a channel with a sudden expansion is stable, i.e. physical, for $Re = 350$.

Consider the two-dimensional channel with a sudden expansion part way along its length, shown in Figure 3.5.12. In the same vein as [46], and continuing the notation developed in previous test cases, we define no-slip Dirichlet boundaries for the domain walls, a Poiseuille flow inlet condition, and the stress free Neumann boundary condition for the outlet. In particular, $\Gamma = \Gamma_{\mathrm{D}} \cup \Gamma_{\mathrm{N}}$, with a peak inlet velocity $u_{\mathrm{peak}} = 0.25$. Again, this numerical experiment is non-dimensional, as we are only interested in the behaviour of the axial flow velocity for a chosen Reynolds number. We define the inlet length $l = 1$, the inlet radius $r = 0.5$, the channel length $L = 10$, and the channel radius $R = 1.5$.

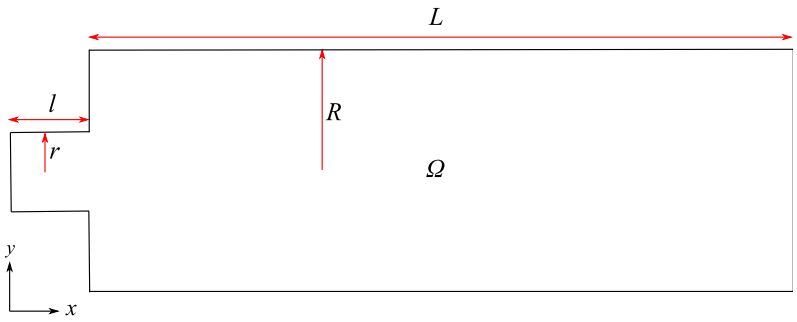

Figure 3.5.12: Two-dimensional channel domain with a sudden expansion. $l = 1$, $r = 0.5$, $L = 10$ and $R = 1.5$.

The domain $\Omega \subset \mathbb{R}^2$ is partitioned into a mesh $\mathcal{T}_h$ consisting of 496 quadrilateral elements. The inlet channel is subdivided by 5 uniformly spaced nodes in each coordinate direction, whilst the outlet channel is partitioned by 41 equally spaced x-direction nodes, and 13 equally spaced y-direction nodes. The mesh is shown in Figure 3.5.13.

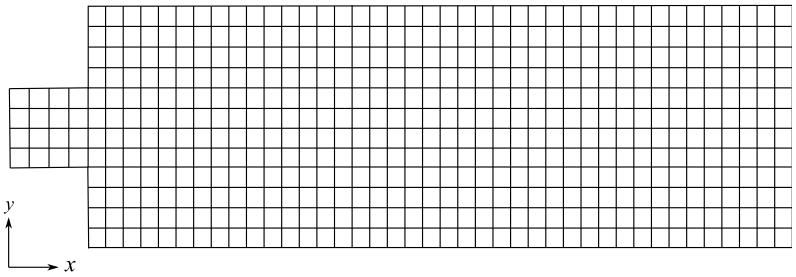

Figure 3.5.13: Two-dimensional mesh $\mathcal{T}_h$, 496 elements.

We present two numerical results, $Re = 300$, where the expected solution is symmetric, and $Re = 350$, where the expected solution is non-symmetric. To achieve the variation in Reynolds number, the kinematic viscosity ν is altered according to Definition 3.1.1.

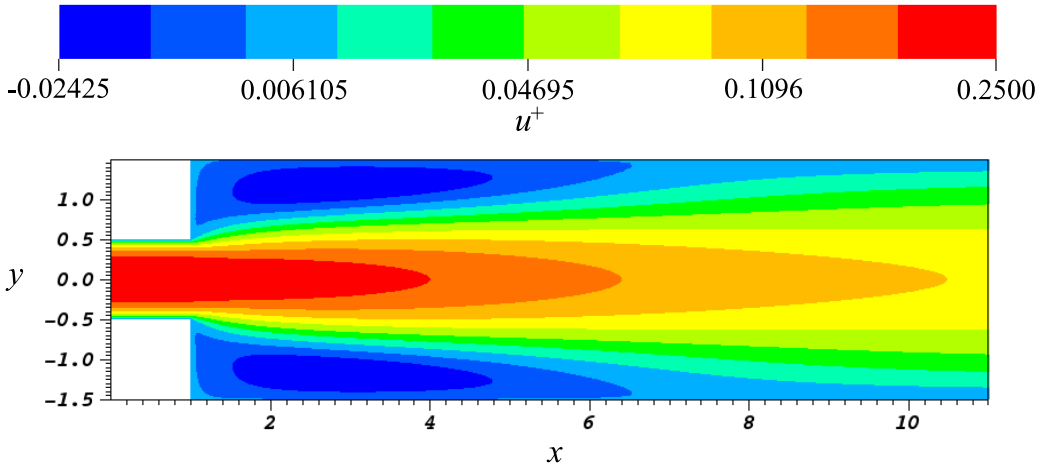

Figure 3.5.14: Colour plot of the scaled axial velocities u^+, $Re = 300$.

Figure 3.5.13 shows the axial velocity of the fluid flow for a Reynolds number $Re = 300$. It is calculated using a symmetric initial estimate for the solution in the Newton method detailed in Section 3.4.4. However, when a non-symmetric initial estimate is chosen for the Newton solver, we still observe the same symmetric numerical solution shown in Figure 3.5.14. This is expected, as the fluid flow is physically stable for this geometry, and is therefore, what is observed in real world experiments.

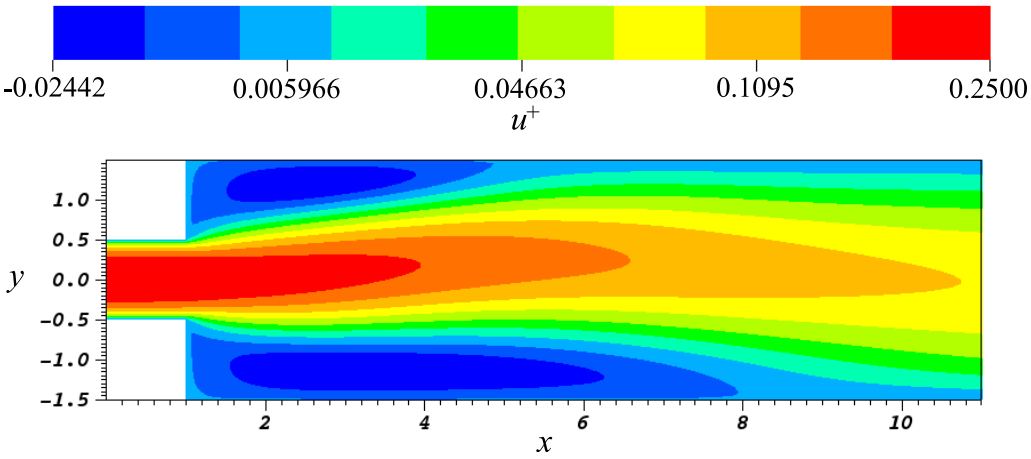

Figure 3.5.15: Colour plot of the scaled axial velocities u^+, $Re = 350$.

Similarly, a non-symmetric fluid flow is observed in experiments with a Reynolds number $Re = 350$, such as Durst, Cherdron, Melling and Whitelaw [44, 56]. Therefore, by selecting a non-symmetric initial estimate for the Newton solver, we generate the results shown in Figure 3.5.15. However, when a symmetric initial estimate is chosen, the physically unstable symmetric solution is produced, comparable to that of Figure 3.5.14. This symmetric numerical solution is not necessarily incorrect, but its physical instability means that it is unlikely to be observed in experiments. Therefore, in such cases where the physical stability of the solution is not clear, we suggest carrying out an initial simulation with a symmetric initial estimate, before perturbing the initial estimate and resolving. If both final solutions are in agreement then we can be sure that the solution is physically stable. Alternatively, a stability analysis can be found in Cliffe, Hall, and Houston [46].

3.5.4 Sudden Expansion Validation Experiment

In this numerical experiment we consider the results presented in [55], of the flow over a backward facing step in three dimensions. We seek to validate the DGFEM solver for three-dimensional domains. This is important as it enables us to consider problems which contain three-dimensional flow patterns. That is, flows which cannot be approximated correctly by only considering two-dimensional cross sections of a three-dimensional problem. Consider $\Omega \subset \mathbb{R}^3$ shown in Figure 3.5.16, with a horizontal x-axis, a vertical y-axis and a perpendicular z-axis. We introduce the inlet height $h = 9.6\,\text{cm}$, the inlet length $l = 46\,\text{cm}$, the step height $S = 0.96\,\text{cm}$, channel length $L = 135\,\text{cm}$ and the channel height $H = 11.5\,\text{cm}$. The rectangular duct is $30.5\,\text{cm}$ wide.

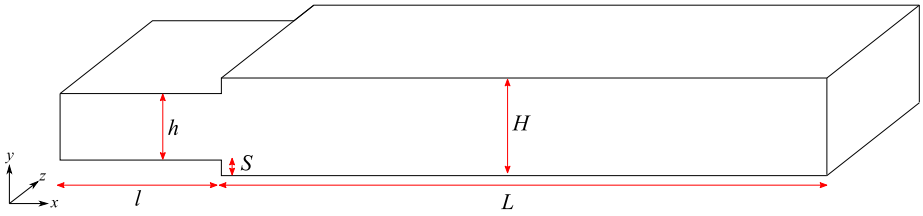

Figure 3.5.16: Schematic of the step geometry.

The boundaries of the domain are such that the inlet and walls use Dirichlet boundary conditions, with a Neumann boundary condition for the outlet. In particular, the domain inlet located on the left most edge of Figure 3.5.16, is Poiseuille flow with peak inlet velocity $u_{\text{peak}} = 11.58\,\text{m}/s$, giving the required average velocity $u_0 = 7.72\,\text{m}/s$. The domain outlet, the far right edge of Figure 3.5.16, is the stress free Neumann condition (3.1.3). Finally, all other edges are domain boundaries are walls, described using the no-slip Dirichlet boundary condition (3.1.2). Next, the domain is partitioned into the graded mesh $\mathcal{T}_h$ shown in Figure 3.5.17 and 3.5.18, consisting of 1896 elements.

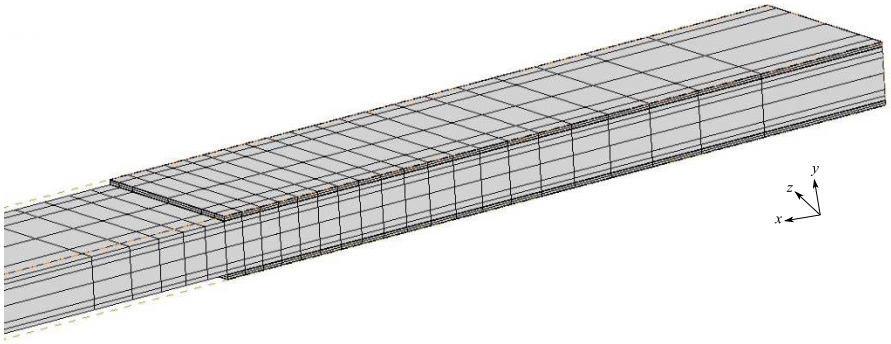

Figure 3.5.17: Three-dimensional graded mesh consisting of 1896 hexahedral elements.

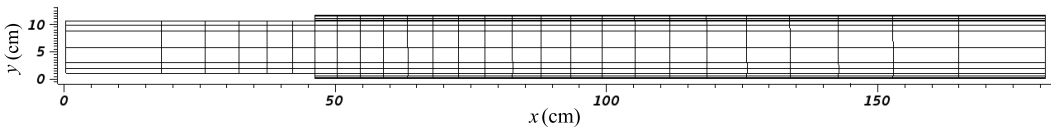

Figure 3.5.18: Cross-section of the mesh, $z = 15.25\,\text{cm}$.

According to [55], the Reynolds number based on the step height is calculated to be $Re = 5000$. The streamwise velocity is sampled in the centre of the duct $z = 15.25\,\text{cm}$ at a number

of stations denoted by $\frac{x}{S}$ along the bottom of the duct, and compared to the results in [55]. The experimental measurements in [55] are time averaged streamwise velocities, making them a good point of comparison for the RANS simulation. These are presented in Figure 3.5.19.

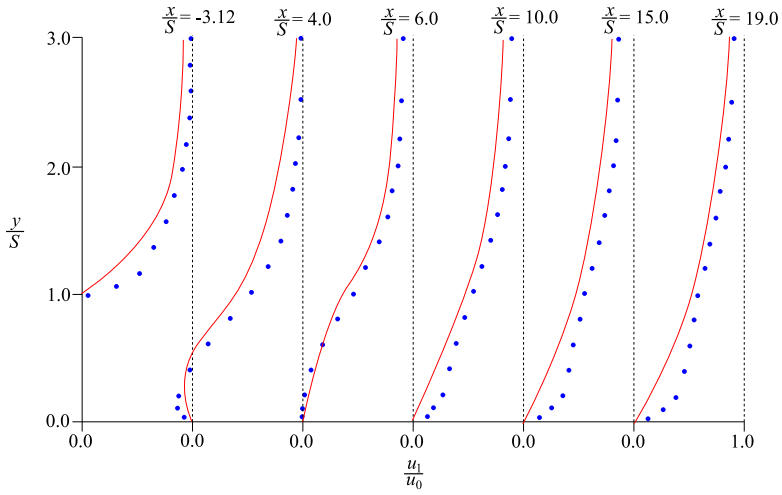

Figure 3.5.19: Comparison of streamwise velocity profiles at $z = 15.25\,\text{cm}$ between [55] (blue circles) and the DGFEM solver (red lines).

For reference, we include colour plots of the axial velocity in Figure 3.5.20 and the static pressure in 3.5.21.

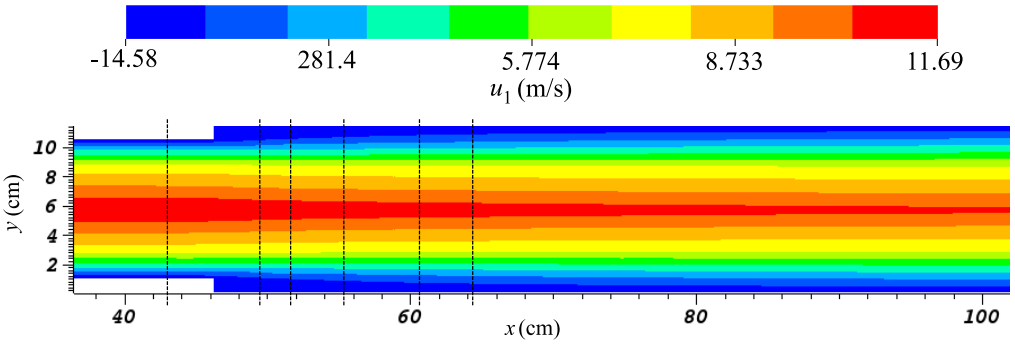

Figure 3.5.20: Colour plot of the axial velocity u_1 at $z = 15.25\,\text{cm}$. $Re = 5000$. The dashed lines denote the locations of the measurement stations $\frac{x}{S}$.

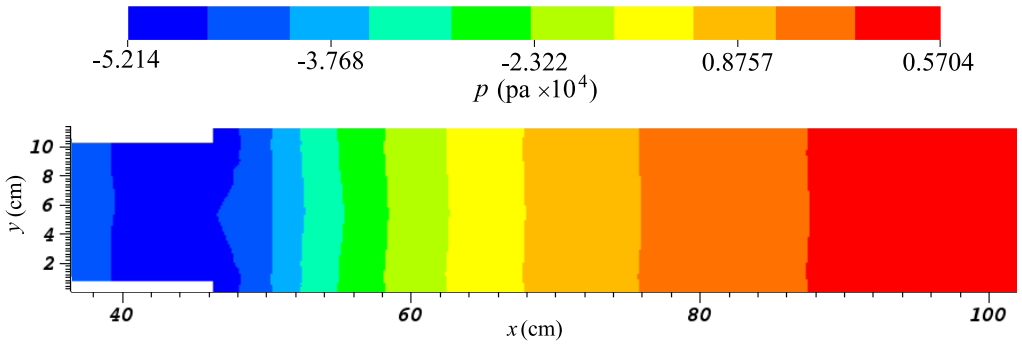

Figure 3.5.21: Colour plot of the static pressure p at $z = 15.25\,\text{cm}$. $Re = 5000$.

This numerical experiment demonstrates the ability of the proposed DGFEM solver to handle three-dimensional flows. This is an important attribute of the solver, as turbomachinery simulations typically contain three-dimensional flow patterns. These flows cannot be accurately represented by considering two-dimensional cross sections of the domain, and instead require a solver capable of resolving in three dimensions to simulate them correctly. Figure 3.5.19 shows some agreement between the experimental data [55] and the DGFEM solution. However, the coarseness of the mesh, combined with the experimental turbulent flow, means that we are unable to properly capture the boundary layer using the laminar DGFEM solver. The relatively coarse mesh, particularly in the recirculation region, is the cause of some of solution inaccuracy at station $\frac{x}{S} = 4.0$, predicting a much more gradual change in velocity close to the wall than is observed in the experimental data, see Figure 3.5.19. In particular, we note that the recirculation velocity is simulated to be smaller than what is observed at station $\frac{x}{S} = 4.0$, as well as making the transition from negative to positive velocity more gradual as we move towards the centre of the duct.

However, in this example, the lack of a turbulence model has a far greater affect on the overall accuracy of the numerical solution. Due to this, we see a reduced rate at which the velocity changes close to the wall, as well as a much earlier flow recover than is observed in the experimental data [55]. It is clear from Figure 3.5.19, that as the flow moves down stream, we recover Poiseuille flow much sooner than the experimental flows recovers. In particular, the laminar solution recovers by station $\frac{x}{S} = 10.0$, whereas the experimental flow does not until station $\frac{x}{S} = 19.0$ is reached. Therefore, before we are able to consider more complicated geometries where the flow undergoes greater physical changes, we must first consider a method to model turbulent flows. As such, in Chapter 4 we explore the development of a suitable turbulence model in order to more accurately capture the boundary layer, before considering automated solution refinement techniques in Chapter 6 to improve mesh density, and, in turn, solution accuracy.

3.6 Concluding Remarks

In this chapter we introduced the DGFEM we use throughout this work to model incompressible flows. In particular, we derived the DGFEM discretisation in Section 3.2, as well as, choosing the numerical flux. We followed [46], using the Lax-Friedrichs flux to approximate the nonlinear convective terms. It is suggested that for compressible flows, the Lax-Friedrichs flux is too dissipative, and there is no mention of its performance for turbulent incompressible flows. As such, when we introduce the turbulence model in Chapter 4, we reassess this choice.

The numerical experiments carried out in Section 3.5 have served to validate the DGFEM for laminar flow test cases. In particular, Figures 3.5.10 and 3.5.11 demonstrated a good agreement between the axial velocities recorded in experiments [9] and the numerical simulation. Figure 3.5.11 showed some discrepancies between the data and the simulation. However, since the simulated flow resolved to the experimental data further along the channel once Poiseuille flow had been recovered, these differences are likely due to three-dimensional flows beginning to develop in the experimental apparatus. The accuracy of the numerical solution suggests that the choice of numerical flux is indeed appropriate for laminar incompressible flows. This finding is reaffirmed in Section 3.5.3, which shows the ability of the DGFEM to capture both the symmetric and non-symmetric solutions for the chosen geometry. Section 3.5.4 demonstrates the ability of the DGFEM to handle three-dimensional flow problems, providing reasonably accurate flow predictions without the use of a turbulence model.

We found that as the Reynolds number of the flow increased, the nonlinear solver converged to the numerical solution in a greater number of Newton iterations. While this is a minor concern for laminar flows, it does present a problem for high Reynolds number turbulent flows, as the nonlinear solver may not converge if the initial solution estimate is too far from the final numerical solution. As such, we devise an algorithm to address these difficulties in Section 4.4.3, ensuring that industrially relevant, high Reynolds number solutions can be found reliably. Therefore, we now progress to study turbulent flows in Chapter 4, with the aim of validating the DGFEM for turbulent simulations.

Chapter 4

Discontinuous Galerkin Methods for Incompressible Turbulent Flows

Turbulence is a naturally occurring physical phenomenon characterised by the creation and dissipation of unsteady vortices or eddies during fluid flow. These eddies occur on all scales, with energy passed from the larger down to the smaller until it is completely dissipated. Turbulent flows are chaotic, possessing an inherently complicated nature with a large amount of information being passed on all time and length scales. This makes them almost impossible to model using direct numerical simulation (DNS), where mesh and time scales are required to be too fine for modern computers to handle effectively. We seek to understand and model only the average flow behaviour using the Reynolds-averaged Navier-Stokes (RANS) equations [2, 25]. Other approaches include Large Eddy Simulation (LES) and model only the largest of the eddies, those which characterise the overall flow and behaviour of the fluid; LES was first proposed in 1963 by Smagorinsky [127], before later being explored in greater detail by Deardorff [49]. These methods are used for simulating time-dependent problems, and as such, require more computational resources compared to the steady-state solutions produced by RANS. Before moving to such transient simulations in the future, we follow the recommendations and requirements of G.E. representatives, by considering steady-state simulations using the RANS equations coupled with a turbulence model.

The incompressible, steady state RANS equations can be written in the following divergence form on the domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$:

$$\begin{aligned} \nabla \cdot (\underline{\boldsymbol{\tau}}) + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) &= \mathbf{0}, \\ \nabla \cdot \mathbf{u} &= 0. \end{aligned} \tag{4.0.1}$$

Here, the matrix $\underline{\boldsymbol{\tau}}$ represents the stress tensor whose value vary depending on the particular turbulence model that is being implemented. As such, we begin our discussion by analysing the various merits of competing turbulence models.

4.1 The Mixing Length Model

In 1925, Prandtl [115] proposed the mixing length hypothesis in order to describe turbulent flows in a simple manner. He suggested that the particles that make up the fluid, group together into larger bodies and then move as uniform masses. Further to this, in a shear flow, the bodies maintain their initial, say x_1 direction momenta for a specified distance in the x_2 direction. This is known as the mixing length and denoted as l_m. In the mixing length model, the only significant velocity gradient is $\partial u_1/\partial x_2$ (maintaining the previous notation). In the laminar, two-dimensional case, the stress tensor $\underline{\boldsymbol{\tau}}$ is a matrix of the form:

$$\underline{\boldsymbol{\tau}} = \begin{bmatrix} -\nu\frac{\partial u_1}{\partial x_1} + p & -\nu\frac{\partial u_1}{\partial x_2} \\ -\nu\frac{\partial u_2}{\partial x_1} & -\nu\frac{\partial u_2}{\partial x_2} + p \end{bmatrix}, \tag{4.1.1}$$

where ν is the kinematic viscosity, and p is the pressure scaled by $\frac{1}{\rho}$. In order to incorporate the mixing length turbulence model into the laminar equations, we can simply introduce a turbulent viscosity term μ_t, writing:

$$\tau_{ij} = -\left(\nu + \mu_t\right)\frac{\partial u_i}{\partial x_j} + \delta_{ij}p,$$

with δ being the Kronecker delta. As such, in the two-dimensional case we have

$$\underline{\boldsymbol{\tau}} = \begin{bmatrix} -\left(\nu + \mu_t\right)\frac{\partial u_1}{\partial x_1} + p & -\left(\nu + \mu_t\right)\frac{\partial u_1}{\partial x_2} \\ -\left(\nu + \mu_t\right)\frac{\partial u_2}{\partial x_1} & -\left(\nu + \mu_t\right)\frac{\partial u_2}{\partial x_2} + p \end{bmatrix}. \tag{4.1.2}$$

We define the turbulent viscosity for the mixing length model as

$$\mu_t = \rho l_m^2 \left|\frac{\partial u_1}{\partial x_2}\right|.$$

The value of l_m for a particular flow through a specified geometry is chosen based upon established experimental results [145]; these values are summarised in Table 4.1, where y represents the distance from the wall. y^+ is the scaled distance from the wall, detailed in Section 4.4.1.

Flow	Mixing Length l_m	L_m
Mixing Layer	$0.07L_m$	Layer width
Jet	$0.09L_m$	Jet half width
Wake	$0.16L_m$	Wake half width
Axisymmetric Jet	$0.075L_m$	Jet half width
Boundary Layer (Viscous sublayer and Log law layer)	$Ky\left(1-\exp\left(-y^+/26\right)\right)$	Boundary layer thickness (K is the eddy diffusion coefficient)
Boundary Layer (Outer layer)	$0.09L_m$	Boundary layer thickness
Pipes and Channels (Fully developed flow)	$L_m\left(0.14-0.08\left(1-y/L_m\right)^2-0.06\left(1-y/L_m\right)^4\right)$	Pipe radius or Channel half width

Table 4.1: Mixing Length l_m values [145].

The Mixing Length Model brings with it a certain simplicity, as we are not introducing any extra PDEs to describe the creation and dissipation of turbulent eddies, and are just adding an algebraic expression into the viscosity term. We are, though, making some rather abrupt assumptions about the nature of the flow in question. For instance, Prandtl's model assumes that the local intensity of the turbulence depends only on the local generation and dissipation rates. Whereas, in reality, turbulent eddies may be created elsewhere and carried to areas of the flow which were otherwise laminar, i.e., turbulence is not always generated locally. Secondly, for geometries with non-planar walls, it becomes particularly difficult to estimate the distribution of mixing-length magnitudes to a sufficient degree of accuracy. As such, this makes the mixing length model inappropriate for our uses in turbine geometries, which are typically curved.

4.2 The $k-\omega$ Turbulence Model

The $k-\omega$ model, first proposed by Kolmogorov [99] in 1942, is a two-equation turbulence model. This means that it adds an extra two PDEs to the original Navier-Stokes system, making it far more computationally expensive. The first equation describes k, the kinetic energy of the turbulence, whereas the second equation describes the dissipation per unit turbulence kinetic energy, denoted as ω. In 1970, Saffman [123] proposed an improved version of Kolmogorov's model, which he developed without prior knowledge of Kolmogorov's work. Saffman's model overcame many of the shortcomings of the original model, and indeed proved to be superior. Further developments by Spalding [102], Wilcox [144], Traci [140] and Rubesin [143] have helped to refine the model and improve its accuracy, particularly on complex geometries.

The particular model [144] implemented here can be seen detailed below. Once again, we consider (4.0.1), but with Reynolds stresses given by

$$\tau_{ij} = -\left(\mu+\mu_t\right)\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\frac{2}{3}k\delta_{ij}\rho.$$

In the two-dimensional case, the stress tensor may be written as

$$\underline{\boldsymbol{\tau}} = \begin{bmatrix} -\left(\mu+\mu_t\right)\left(\frac{\partial u_1}{\partial x_1}+\frac{\partial u_1}{\partial x_1}\right)-\frac{2}{3}k\rho & -\left(\mu+\mu_t\right)\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right) \\ -\left(\mu+\mu_t\right)\left(\frac{\partial u_2}{\partial x_1}+\frac{\partial u_1}{\partial x_2}\right) & -\left(\mu+\mu_t\right)\left(\frac{\partial u_2}{\partial x_2}+\frac{\partial u_2}{\partial x_2}\right)-\frac{2}{3}k\rho \end{bmatrix}.$$

Here, μ is the molecular viscosity, μ_t represents the turbulent viscosity, and ρ the density. Then, making use of the standard Einstein summation notation and the Kronecker delta δ, the transport equations for k and ω for high Reynolds number flows are given as

$$\begin{aligned} \nabla\cdot(\rho k\mathbf{u}) =& \nabla\cdot\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\nabla k\right]+P_k-\beta^*\rho k\omega, \\ \nabla\cdot(\rho\omega\mathbf{u}) =& \nabla\cdot\left[\left(\mu+\frac{\mu_t}{\sigma_\omega}\right)\nabla\omega\right]+\gamma_1\left(2\rho S_{ij}\cdot S_{ij}-\frac{2}{3}\rho\omega\frac{\partial u_i}{\partial x_j}\delta_{ij}\right)-\beta_1\rho\omega^2, \end{aligned} \tag{4.2.1}$$

where P_k, S_{ij} and μ_t are given by:

$$P_k = \left(2\mu_t S_{ij}\cdot S_{ij}-\frac{2}{3}\rho k\frac{\partial \mathbf{u}_i}{\partial x_j}\delta_{ij}\right),\ S_{ij}=\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij},\ i,j=1,2,$$

with

$$\mu_t = C_\mu\rho\frac{k}{\omega}.$$

The model constants are set according to [145], with $C_\mu = 1.0$, $\sigma_k = 2.0$, $\sigma_\omega = 2.0$, $\gamma_1 = 0.553$, $\beta_1 = 0.075$ and $\beta^* = 0.09$. We remark that (4.2.1) are advection-diffusion equations of the form (2.1.1). These transport equations become increasingly convection dominated at high Reynolds numbers, rendering them unsuitable for solving using traditional continuous FEM techniques. On the other hand, DGFEMs handle the lack of solution "smoothness" associated with convection dominated problems without the need for any special considerations.

Following, as per [25, 145], our model uses $\tilde{\omega} = \ln(\omega)$ rather than simply ω. This is in order to ensure a smoother near-wall distribution, as well as to guarantee the positivity of the variable ω. Furthermore, we also introduce realizability conditions on the turbulent stresses, constraining $\overline{k} = \max(k, 0)$ and $\tilde{\omega}_r = \max(\tilde{\omega}, \tilde{\omega}_{r_0})$. As proposed in [25, 26, 118], let $\tilde{\omega}_{r_0}$ represent the lower bound on $\tilde{\omega}$ that ensures the positivity of the normal turbulent stresses, as well as the satisfaction of the following inequalities:

$$e^{\tilde{\omega}_{r_0}} - \frac{3}{2} C_\mu S_{ii} \geq 0, \quad i = 1, ..., d,$$

$$\left(e^{\tilde{\omega}_{r_0}}\right)^2 - \frac{3}{2} C_\mu \left(S_{ii} + S_{jj}\right) e^{\tilde{\omega}_{r_0}} + \frac{9}{4} C_\mu^2 \left(S_{ii} S_{jj} - S_{ij}^2\right) \geq 0, \quad i, j = 1, ..., d, \; i \neq j.$$

Therefore, the limited turbulent viscosity is given by

$$\bar{\mu}_t = C_\mu \rho k e^{-\tilde{\omega}_r}.$$

Since we are considering here the incompressible Navier-Stokes equations, we are able to make use of the incompressibility condition and eliminate the constant density ρ. One major advantage over other turbulence models, especially for the purpose of a DGFEM implementation, is that it allows for the standard Dirichlet boundary conditions to be used to model flow inlets and boundary walls. It is important to note that other turbulence models were considered in this work before settling upon the $k - \omega$ model; in particular the $k - \varepsilon$ model [103]. We prefer to work with the $k - \omega$ model due to the straight-forward nature of the model's implementation, being able to apply it throughout the boundary layer without near-wall modifications, as well as its high performance in industrial turbomachinery applications [145].

We apply boundary conditions similar to those used in Chapter 3, with

$$\mathbf{u} = g_{\mathrm{D}} \text{ on } \Gamma_{\mathrm{D}},$$

$$(\mu + \mu_t) \frac{\partial \mathbf{u}}{\partial \mathbf{n}} - p\mathbf{n} = 0 \text{ on } \Gamma_{\mathrm{N}}.$$

The specific values of k and $\tilde{\omega}$ on these boundaries are discussed in Section 4.4.

4.3 DGFEM Discretisation

We derive an interior penalty DGFEM discretisation of the incompressible Navier-Stokes equations with the $k-\omega$ model turbulence model. As such, we first write the system in a compact form as follows on a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$:

$$\nabla \cdot \mathbf{F_C} - \nabla \cdot \mathbf{F_V} - \mathbf{S} = \mathbf{0}, \tag{4.3.1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{4.3.2}$$

where

$$\mathbf{u} = \begin{bmatrix} u_i \\ k \\ \tilde{\omega} \end{bmatrix}, \quad \mathbf{F}_j^C(\mathbf{u}) = \begin{bmatrix} u_i u_j \\ k u_j \\ \tilde{\omega} u_j \end{bmatrix}, \quad \mathbf{F}_j^V(\mathbf{u}, \nabla_h \mathbf{u}) = \begin{bmatrix} -\tau_{ij} \\ (\mu + \sigma_k \mu_t) k_{x_j} \\ (\mu + \sigma_\omega \mu_t) \tilde{\omega}_{x_j} \end{bmatrix},$$

$$\mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) = \begin{bmatrix} 0 \\ -\tau_{ij}^R - \beta_k k e^{\tilde{\omega}_r} \\ -\alpha_\omega \frac{\omega}{k} \tau_{ij}^R u_{i,x_j} - \beta_\omega e^{\tilde{\omega}_r} + (\mu + \sigma_\omega \mu_t) \tilde{\omega}_{x_k} \tilde{\omega}_{x_k} \end{bmatrix}, \quad i, j = 1, ..., d,$$

and

$$\tau_{ij}^R = -\mu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} k \delta_{ij}.$$

The weak form may be derived upon multiplication of (4.3.1) by a suitable smooth test function $\mathbf{v}$ and integrating by parts over the domain Ω,

$$-\int_\Omega \nabla \mathbf{v}^{\mathrm{T}} \cdot \mathbf{F}(\mathbf{u}, \nabla \mathbf{u}) \, \mathrm{d}\mathbf{x} + \int_{\partial\Omega} \mathbf{v}^{\mathrm{T}} \mathbf{F}(\mathbf{u}, \nabla \mathbf{u}) \cdot \mathbf{n} \, \mathrm{d}s - \int_\Omega \mathbf{v}^{\mathrm{T}} \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) \, \mathrm{d}\mathbf{x} = \mathbf{0},$$

for all $\mathbf{v} \in \left(H^1(\Omega)\right)^{d+2}$. Similarly, we multiply (4.3.2) by a suitable smooth test function q and integrate by parts,

$$-\int_\Omega \nabla q \cdot \mathbf{u} \, \mathrm{d}\mathbf{x} + \int_{\partial\Omega} q (\mathbf{u} \cdot \mathbf{n}) \, \mathrm{d}s = 0,$$

for all $q \in H^1(\Omega)$.

For readability, the symbol $\mathbf{F}$ represents the algebraic sum of the fluxes, $\mathbf{F} = \mathbf{F}_C - \mathbf{F}_V$. Then, the test function and the solution space are defined such that

$$\begin{aligned} \mathbf{V}_{h,m} &= \left\{ (\mathbf{v}, k, \tilde{\omega}) \in \left[L^2(\Omega)\right]^{d+2} : \ (\mathbf{v}, k, \tilde{\omega}) \mid_\kappa \in [\mathcal{R}^m(\kappa)]^d \times \left[\mathcal{R}^{m-1}(\kappa)\right]^2, \kappa \in \mathcal{T}_h \right\}, \\ Q_{h,m} &= \left\{ q \in L^2(\Omega) : \ q \mid_\kappa \in \mathcal{R}^{m-1}(\kappa), \kappa \in \mathcal{T}_h \right\}. \end{aligned} \tag{4.3.3}$$

Once again, we make the choice of stable pairs, employing second order piece-wise continuous polynomials to approximate the velocity $\mathbf{u}$, and piece-wise linear functions for the pressure p. We remark here that the choice of approximation space of the turbulent variables k and $\tilde{\omega}$ is important to the computational efficiency, but not to the stability of the proposed DGFEM as $\mathbf{u}$ and p are. Therefore, since we are not directly concerned with the specific values of k and $\tilde{\omega}$, only the effect they have on the physical variables $\mathbf{u}$ and p, we use piece-wise linear functions to represent them also. In general, the choice of polynomial degree is arbitrary and should be at most m [118].

The choice of numerical flux remains the Lax-Friedrichs flux denoted by

$$\mathcal{H}\left(\mathbf{v},\mathbf{w},\mathbf{n}\right) := \frac{1}{2}\left(F\left(\mathbf{v}\right)\cdot\mathbf{n} + F\left(\mathbf{w}\right)\cdot\mathbf{n} - \gamma\left(\mathbf{w}-\mathbf{v}\right)\right),$$

with $\gamma := \max\left(\phi^{+},\phi^{-}\right)$, whereby ϕ^{+} and ϕ^{-} are the largest eigenvalues in absolute magnitude of the Jacobi matrices $\frac{\partial}{\partial\mathbf{u}}\left(F\left(\cdot\right)\cdot\mathbf{n}\right)$ computed at $\mathbf{v}$ and $\mathbf{w}$ respectively [46]. The boundary function is dependent upon the boundary it is computed on: $\mathbf{u}_{\Gamma}\left(\mathbf{u}\right) = g_{\mathrm{D}}$ on a Dirichlet boundary and $\mathbf{u}_{\Gamma}\left(\mathbf{u}\right) = \mathbf{u}^{+}$ on a Neumann boundary.

Hence the discrete problem becomes: find $\left(\mathbf{u}_h, p_h\right) \in \mathbf{V}_{h,m} \times Q_{h,m}$ such that for all $\left(\mathbf{v}_h, q_h\right) \in \mathbf{V}_{h,m} \times Q_{h,m}$:

$$\begin{aligned}
A_h\left(\mathbf{u},\mathbf{v}\right) = &\left(\int_{\Omega}\left(-\mathbf{F}^{C}\left(\mathbf{u}\right) + \mathbf{F}^{V}\left(\mathbf{u},\nabla_h\mathbf{u}\right)\right) : \nabla_h\mathbf{v} - \mathbf{S}\left(\mathbf{u},\nabla_h\mathbf{u}\right)\cdot\mathbf{v}\,\mathrm{d}\mathbf{x}\right.\\
&+\int_{\Gamma_{\mathrm{int}}}\mathcal{H}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)\left[\mathbf{v}\right]\,\mathrm{d}s + \int_{\Gamma_{\mathrm{int}}}\left\langle\mathbf{F}^{V}\left(\mathbf{u},\nabla_h\mathbf{u}\right)\right\rangle : \left[\mathbf{v}\right]\,\mathrm{d}s\\
&\left.-\int_{\Gamma_{\mathrm{int}}}\left\langle\mathbf{G}^{\mathrm{T}}\left(\mathbf{u}\right)\nabla_h\mathbf{v}\right\rangle : \left[\mathbf{u}\right]\,\mathrm{d}s + \int_{\Gamma_{\mathrm{int}}}\sigma\left[\mathbf{u}\right]\left[\mathbf{v}\right]\,\mathrm{d}s + N_{\Gamma}\left(\mathbf{u},\mathbf{v}\right)\right) = 0,\\
B_h\left(\mathbf{v},q\right) = &-\int_{\Omega} q\nabla_h\cdot\mathbf{v}\,\mathrm{d}\mathbf{x} + \int_{\Gamma_{\mathrm{int}}\cup\Gamma_{\mathrm{D}}}\left\langle q\right\rangle\left[\mathbf{v}\right]\,\mathrm{d}s = 0,
\end{aligned} \tag{4.3.4}$$

where

$$\begin{aligned}
N_{\Gamma}\left(\mathbf{u},\mathbf{v}\right) = &\int_{\Gamma}\mathcal{H}\left(\mathbf{u}^{+},\mathbf{u}_{\mathbf{\Gamma}}\left(\mathbf{u}^{+}\right),\mathbf{n}\right)\cdot\mathbf{v}^{+}\,\mathrm{d}s - \int_{\Gamma}\mathbf{n}\cdot\mathbf{F}_{\Gamma}^{V}\left(\mathbf{u}^{+},\nabla_h\mathbf{u}^{+}\right)\mathbf{v}^{+}\,\mathrm{d}s\\
&+\int_{\Gamma}\sigma\mathbf{u}^{+}\mathbf{v}^{+}\,\mathrm{d}s - \int_{\Gamma}\left(\mathbf{G}_{\Gamma}^{\mathrm{T}}\left(\mathbf{u}^{+}\right)\nabla_h\mathbf{v}^{+}\right) : \left(\mathbf{u}^{+} - \mathbf{u}_{\Gamma}\left(\mathbf{u}^{+}\right)\right)\otimes\mathbf{n}\,\mathrm{d}s,
\end{aligned}$$

with

$$\mathbf{G}^{\mathrm{T}}\left(\mathbf{u}\right)\nabla_h\mathbf{v} : \mathbf{u}\otimes\mathbf{n} = \left(\frac{\partial\mathbf{F}_i^{V}\left(\mathbf{u},\nabla_h\mathbf{u}\right)}{\partial u_{x_j}}\right)_{ji}\frac{\partial\mathbf{v}_k}{\partial x_i}\mathbf{u}_k\mathbf{n}_i,\quad i,j,k = 1,...,d,$$

and

$$\mathbf{G}(\mathbf{u}) = \left(\frac{\partial \mathbf{F}_i^V(\mathbf{u}, \nabla_h \mathbf{u})}{\partial u_{x_j}} \right)_{ij}, \quad i, j = 1, ..., d.$$

4.4 Numerical Experiments

We consider some further numerical examples in order to both validate the proposed DGFEM for turbulent flows, and to develop a reliable method to raise the Reynolds number of a numerical solution. In Section 4.4.1, we in introduce "the law of the wall," to better analyse the interaction between the turbulent flow and the boundary walls. We provide comparisons between the DGFEM and the results of Wilcox [145] in Section 4.4.2, evaluating the ability of the scheme to simulate simple turbulent flows. These initial comparisons indicate the appropriateness of the Lax-Friedrichs numerical flux for turbulent flows, as well as allowing for any adjustments to the scheme to be made before considering industrially relevant geometries in Chapter 5. We build on the ideas developed in Section 3.4, again, with all numerical calculations carried out using AptoFEM, in conjunction with both MUMPS and ParMETIS.

In Section 4.4.3, we develop a highly efficient algorithm to reliably produce simulations of high Reynolds number turbulent flows. In particular, we consider a continuation type approach, manipulating the nonlinear solver using low Reynolds number numerical solutions to increase the rate at which the solver converges for high Reynolds number flows. We remark that, for industrial simulations that require a number of CPU days to complete, failure of the nonlinear solver to converge at all is unacceptable. As such, this algorithm is of the utmost importance to the overall success of this work.

Finally, in Section 4.4.4, we consider the experimental results presented in [107], employing the algorithm developed in Section 4.4.3 to improve the reliability of the numerical solver. We compare the near wall velocities, assessing the behaviour of the numerical solution close to the wall. This is important as it provides an insight into the accuracy of the numerical flux and the turbulence model, showing if the velocity of the flow is being affected in the correct manner.

4.4.1 The Law of the Wall

The "law of the wall" is a valuable tool in understanding and analysing turbulent flows. The result is obtained by dimensional analysis of the near-wall flow, and was first proposed by Theodore von Kármán in 1930 [136]. It states that the average velocity of a turbulent

flow at a point close to the wall, is proportional to the logarithm of the distance from the wall to the same point. In particular, the two dimensionless quantities of interest are $y^+ = \frac{yu_\tau}{\nu}$ and $u^+ = \frac{u}{u_\tau}$, where y is the measured distance from the wall, $u_\tau = \sqrt{\frac{\tau_w}{\rho}}$ is the shear velocity, and τ_w is the local wall shear stress. Indeed, plotting u^+ against y^+, we expect to see a distinctive viscous sublayer close to the wall where $u^+ \approx y^+$, a log-law region where $u^+ = \frac{1}{\kappa}\ln(y^+) + C^+$ holds for a constant C^+ and the Von Kármán constant κ, as well as a buffer layer separating the two regions.

4.4.2 Turbulent Channel

In this section, we consider a simple channel domain, to compare the results generated from the companion software provided in Wilcox [145], and those from the DGFEM. The software from [145] is useful in developing or modifying turbulence models, as it provides a turbulent ordinary differential equation (ODE) solver, as well as, a range of test cases for simple flow problems, most notably for incompressible turbulent flows. This test case provides an initial indication on the accuracy of the proposed DGFEM for turbulent flows.

Formally, consider a two-dimensional rectangular channel domain $\Omega \subset \mathbb{R}^2$, with a half width $R = 1$ and a length of $L = 60$. This is shown in Figure 4.4.1. As in Sections 3.5.1 and 3.5.3, this test case is dimensionless, as we are only interested in the behaviour of the flow at the chosen Reynolds numbers, rather than attempting to model a physical experiment. We define boundaries such that $\Gamma = \Gamma_D \cup \Gamma_N$.

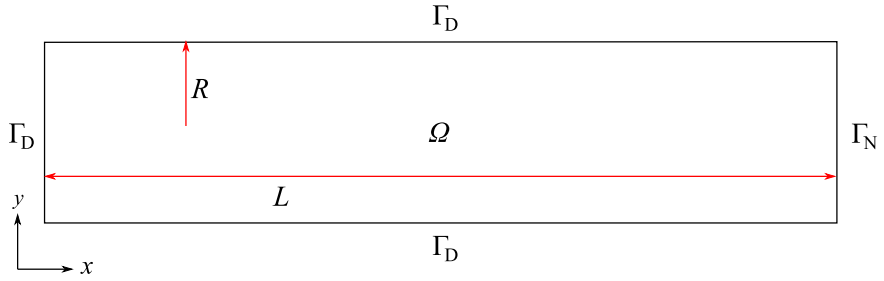

Figure 4.4.1: Rectangular channel domain, with a length of 60 and a half width $R = 1$.

The channel walls are modelled using the no-slip Dirichlet boundary condition, that is, $u_1, u_2 = 0$ at the wall. Following [145], we define $k = 0$ and $\tilde{\omega} = \ln\left(\frac{60\nu}{\beta_1 d_1^2}\right)$ at the wall; where d_1 is the width (measured perpendicular to the wall) of the boundary element of the computational mesh. The inlet conditions for u_1, u_2, k and ω are provided by the Wilcox software, and the outflow is modelled using the stress-free Neumann condition

(3.1.3). The comparison is made by sampling a slice of the DGFEM solution at $x = 54$, such that the flow is allowed to fully develop. A point before the outflow boundary is chosen, as to eliminate any inaccuracies associated with the domain outflow conditions. Then, a comparison is drawn between the solution produced by the Wilcox software and the DGFEM slice, analysing their near-wall behaviour. In this test case, calculations are carried out on a graded triangular mesh $\mathcal{T}_h$, consisting of 720 elements. In particular, 31 equally spaced nodes subdivide the x-axis, while 13 nodes are placed across the channel width, as shown in Figure 4.4.2.

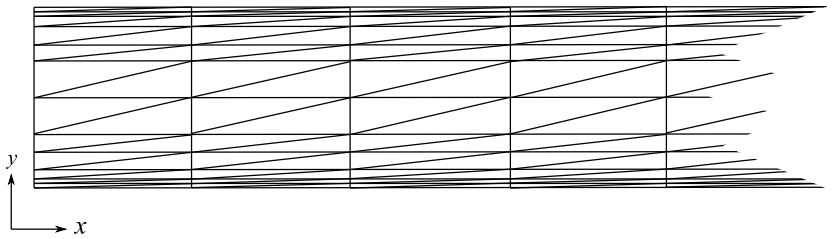

Figure 4.4.2: Two-dimensional graded mesh $\mathcal{T}_h$, 720 triangular elements.

In order to provide a useful comparison, the turbulent flow is simulated for Reynolds numbers $Re = 50$, $Re = 1,500$, $Re = 20,130$, and $Re = 258,300$. A sample of 101 uniformly spaced points is collected across the domain at the point $x = 54$. Then, a comparative sample is taken from the Wilcox solution, and the quantities of u^+ and $y^+$are calculated. Figures 4.4.3, 4.4.4, 4.4.5 and 4.4.6 show the Wilcox solution using a broken black line, whilst the DGFEM solution is represented by the solid red line.

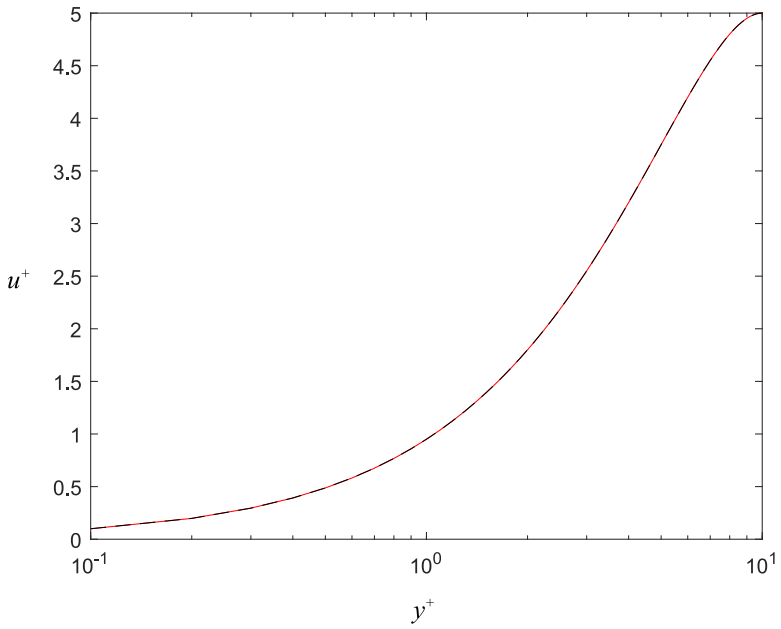

Figure 4.4.3: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 50$. DGFEM solution is shown by a solid red line, whilst the Wilcox solution is denoted by a broken black line.

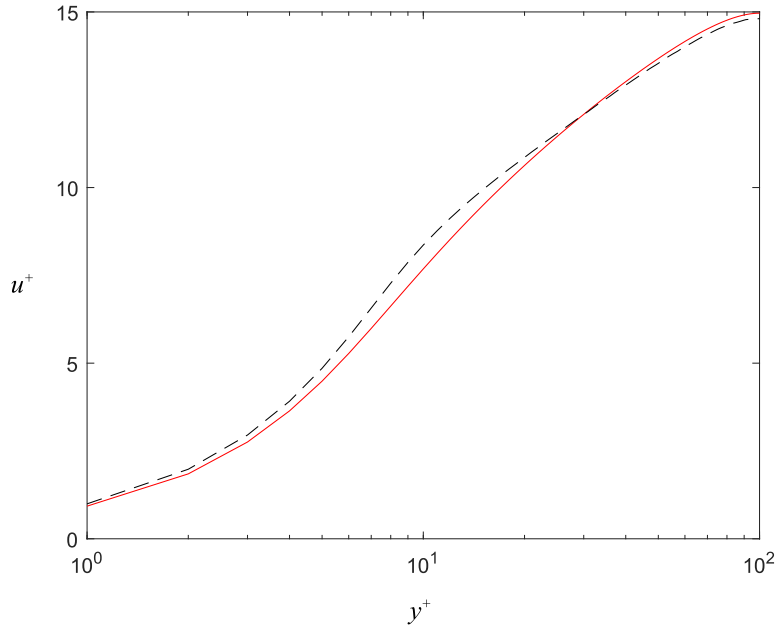

Figure 4.4.4: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 1,500$. DGFEM solution is shown by a solid red line, whilst the Wilcox solution is denoted by a broken black line.

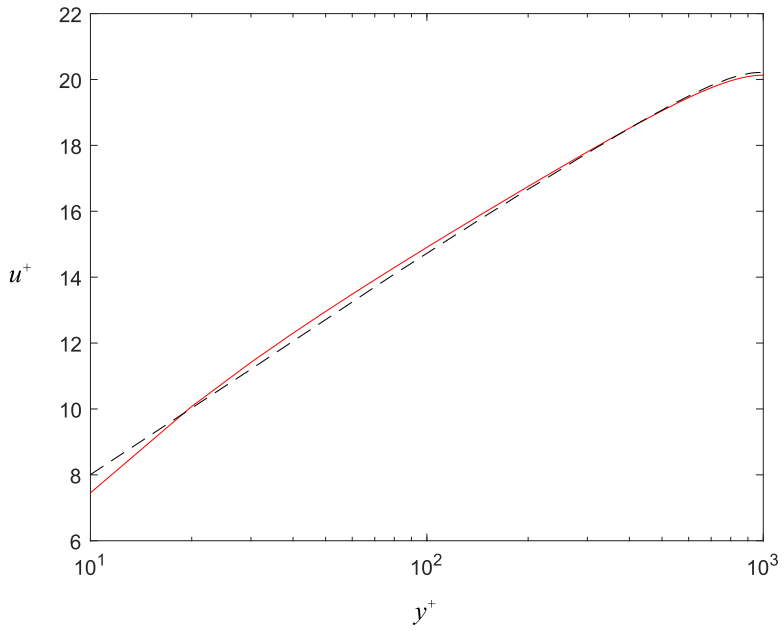

Figure 4.4.5: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 20,130$. DGFEM solution is shown by a solid red line, whilst the Wilcox solution is denoted by a broken black line.

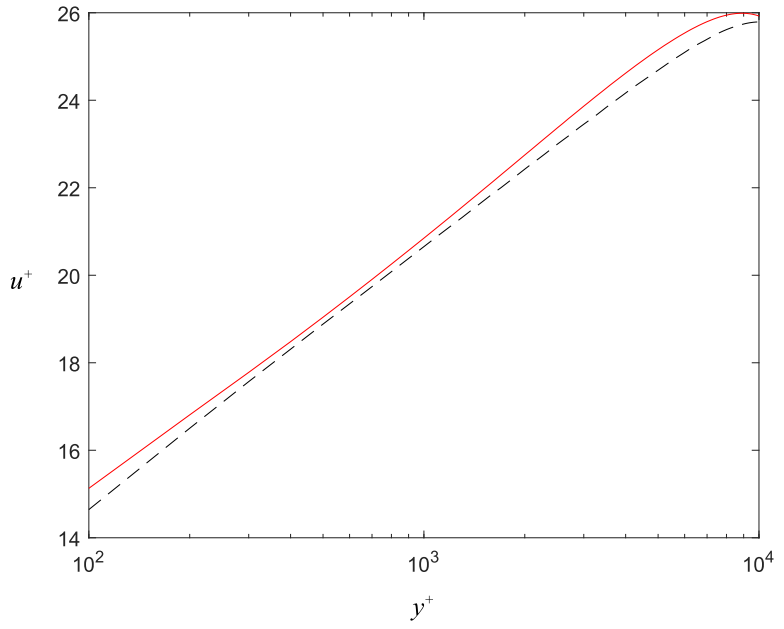

Figure 4.4.6: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 258,300$. DGFEM solution is shown by a solid red line, whilst the Wilcox solution is denoted by a broken black line.

The DGFEM appears to approximate the turbulent flow well, shown in Figures 4.4.3 and 4.4.4. However, as the Reynolds number increases, both the Wilcox solver and the DGFEMs ability to correctly capture the viscous sublayer, diminishes. The familiar S-shaped curve expected in Figures 4.4.5 and 4.4.6 is completely missing. See Figure 4.4.10 for the expected graph shape. This is due to a lack of resolution at the domain wall, indicated by the differing length scales of the y^+ variable, particularly in Figures 4.4.5 and 4.4.6. To resolve this, in the case of the DGFEM, we would increase the mesh density at the wall, allowing us to capture the viscous sublayer correctly. However, as the Wilcox solver is restricted to the current mesh density, we do not do this here.

Secondly, as the Reynolds number is increased, the DGFEM solution begins to exhibit some differences compared to the Wilcox solution. This is most notable in Figure 4.4.6, with the DGFEM predicting larger u^+ values. This overshoot could be explained by the differences between the methods of the DGFEM and the Wilcox solver. In particular, the Wilcox solver uses a one-dimensional ODE solution [145] to approximate a cross section of the two-dimensional flow, rather than calculating solution values at appropriately spaced intervals in the domain, such as in FVM and DGFEM. This method assumes that the flow is constant along the length of the domain. Whilst this is appropriate for low Reynolds number flows, for high Reynolds number flows, the solution may be somewhat affected by forces acting parallel to the flow [141].

These numerical experiments have shown that the Lax-Friedrichs flux is indeed suitable for incompressible turbulent flows, showing good agreement overall between the velocity profiles for the Wilcox and DGFEM solutions. The inability of the DGFEM to capture accurately, the viscous sublayer for high Reynolds number flows suggests that the distance between the channel wall and the first internal mesh node should decrease, as the Reynolds number increases. Indeed, this makes incompressible flows ideally suited for automatic h-refinement. We explore the most effective uses of this refinement strategy in Chapter 6.

4.4.3 Continuation Experiment

In this numerical experiment, we develop an algorithm to reliably produce accurate solutions to high Reynolds number flows. Large scale industrial simulations typically require several days of CPU time to complete. They are usually carried out under strict time constraints, with little to no time for overrun. In the case of DGFEMs, which rely on non-linear solvers to iteratively progress an initial estimate into the true solution, this concern is particularly poignant. Failure of the nonlinear solver to converge could invalidate hours, if not days of CPU time, which is simply unacceptable from a business perspective. As such, in this numerical experiment, we seek to address these concerns.

In the case of high Reynolds number turbulent flows, if the initial estimate of the numerical solution is too different from the actual numerical solution, then it is highly unlikely that the nonlinear solver will converge due to the stiffness of the system. As such, we present a continuation type approach, using a lower Reynolds number solution to approximate a higher Reynolds number solution. This is a variation on the natural parameter continuation method. Alternative approaches include, pseudo-arclength and Gauss–Newton methods, but these are not explored in this work due to the additional computational requirements associated with these methods. For our natural approach, a domain with appropriate boundary conditions is selected. Then, the initial Reynolds number of the flow is chosen through adjustments to the kinematic viscosity. An initial estimate of zero is selected for the nonlinear solver, and a laminar numerical solution is calculated. Next, a turbulent solution is calculated for the same Reynolds number, using the laminar solution as an initial estimate for the nonlinear solver. The Reynolds number of the flow is increased, and another turbulent solution is calculated using the previous lower Reynolds number turbulent solution as an initial estimate for the nonlinear solver. The amount by which the Reynolds number can be increased between progressive solves without the nonlinear solver failing to converge is unclear. As such, we explore for the remainder of this work, particularly in Chapters 6 and 7, the optimum way to choose this value.

Additionally, we wish to address the concerns arising in Section 4.4.2, namely, the DGFEM failing to capture the viscous sublayer for high Reynolds number flows. Therefore, in this section, we generate our own inflow conditions and increase the mesh density at the domain boundaries. As such, we consider a channel domain $\Omega \subset \mathbb{R}^2$, with channel half width $R = 1$, and length $L = 120$, as shown in Figure 4.4.7. The channel walls are modelled using the Dirichlet no-slip condition, while the outlet uses a stress free Neumann condition. Initially, the Reynolds number is set to $Re = 300$, and a Poiseuille flow inlet condition located at $x = 0$, with peak velocity $u_{\text{peak}} = 13.88$. Recall, the Reynolds number is calculated according to Definition 3.1.1, with characteristic length scale $L_C := R$.

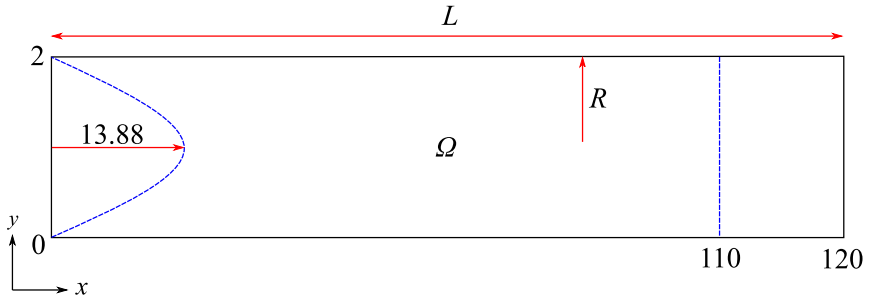

Figure 4.4.7: Channel geometry. $R = 1$, $L = 10$.

The solution is calculated on a graded mesh $\mathcal{T}_h$, consisting of $10,000$ triangular elements, as shown in Figure 4.4.8. In particular, 51 uniformly spaced nodes divide the x-axis, while 101 nodes divide the y-axis.

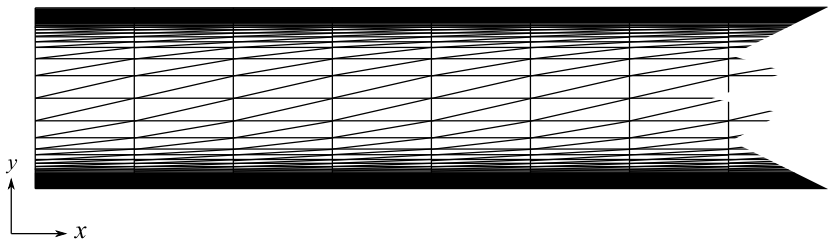

Figure 4.4.8: Two-dimensional graded mesh $\mathcal{T}_h$, $10,000$ triangular elements.

Once the solution is calculated, a slice sample is taken at $x = 110$, to provide turbulent inlet conditions for the next numerical experiment. For the first turbulent solve, uniform values of 1 were chosen for the initial inlet conditions of k and ω, while subsequent solves used the values taken from the solution slice. The Reynolds number is increased by 10% for each successive solve. We found this value sufficient for the channel geometry, with the nonlinear solver always converging to the solution. However, we note that for more complicated geometries, this value may be too large, and may need to be adjusted to ensure solver reliability. In order to satisfy the inf-sup condition of stable pairs, a polynomial degree of two was selected for the velocity components of the solution, whilst linear polynomials represented the pressure and turbulent terms.

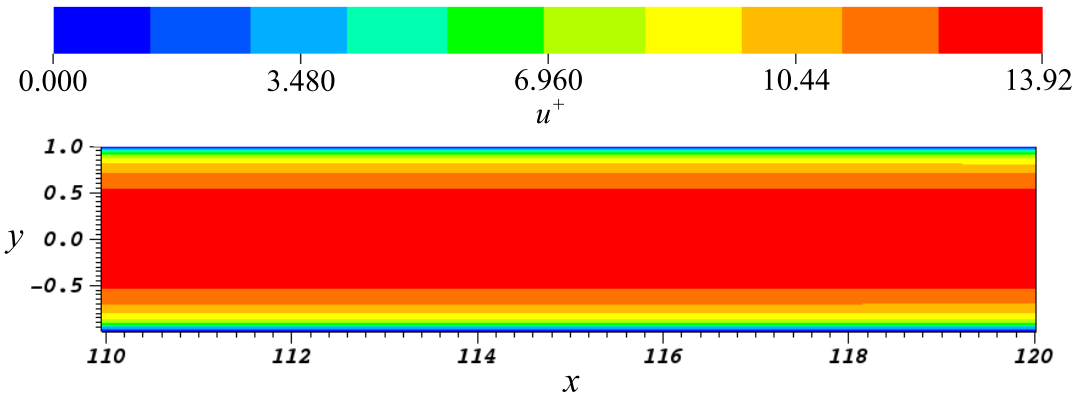

Figure 4.4.9: Colour plot of the scaled axial velocity u^+, $Re = 115,966$.

Figure 4.4.9 is a colour plot of the scaled axial velocity, $Re = 115,966$. Importantly, the peak axial velocity is very similiar to the peak inlet condition, suggesting that the solution is correct. If the peak velocity increases significantly as the Reynolds number is raised, then this suggests a lack of mesh resolution at the domain wall, resulting in an incorrect final numerical solution. Finally, we sample the solution slice at $x = 110$, and consider the "law of the wall," plotting y^+ against u^+ in Figure 4.4.10. In particular, Figure 4.4.10 demonstrates that through progressive increases in Reynolds number, the DGFEM solution is able to retain the correct viscous sublayer representation even when the Reynolds number is very large. The characteristic S-curve is shown, with a viscous sublayer close to the wall where $u^+ \approx y^+$, and a straight line representing the log-law region at $10^2 \leq y^+ \leq 10^3$.

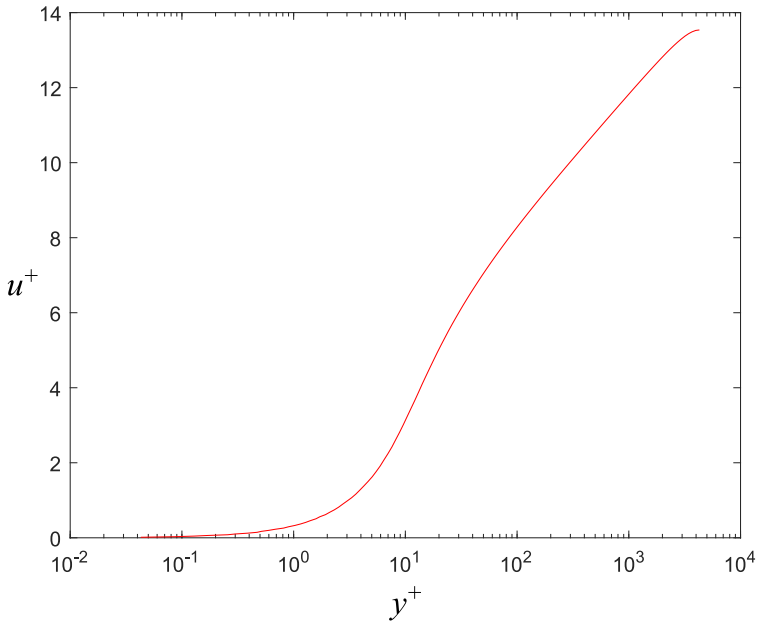

Figure 4.4.10: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 115,966$.

Additionally, the continuation algorithm we proposed, ensured that the nonlinear solver converged for each Reynolds number attempted, up to the target of $Re = 115,966$. This is significant, because in the case of large scale industrial simulations, it is not uncommon for the solver to be left unattended for up to a week [93]. Should the nonlinear solver fail to converge for a particular Reynolds number, then this will result in severe delays elsewhere in the production queue. We continue to develop and refine this algorithm throughout this work, especially when we consider more complicated geometries in Chapters 5, 6 and 7.

4.4.4 Channel Flow Validation

Following the development of the DGFEM in Section 4.4.3, and the successful addition of the continuation algorithm, we now seek to validate our work against experimental test data. We consider the results presented in [107], a time averaged turbulent flow through a channel domain with Reynolds number $Re = 2,777$. We carry out a two-dimensional numerical simulation, approximating the three-dimensional experiment, by considering a longitudinal cross section of the three dimensional flow. Consider a channel domain $\Omega \subset \mathbb{R}^2$, with channel half width $R = 2.04\,\text{cm}$, and length $L = 1060\,\text{cm}$, as shown in Figure 4.4.11. No-slip boundary conditions are used to model the walls of the channel, and the stress free Neumann condition is employed for the domain outlet located at $x = 1060\,\text{cm}$.

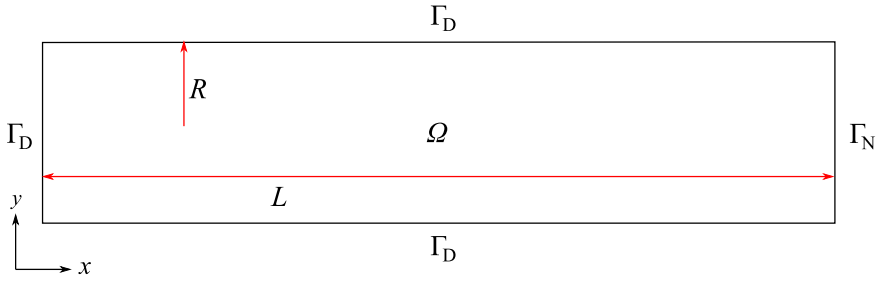

Figure 4.4.11: Channel geometry. $R = 2.04\,\text{cm}$, $L = 1060\,\text{cm}$.

The domain is partitioned into a mesh $\mathcal{T}_h$, graded towards the channel walls and consisting of $50,000$ triangular elements. In particular, the horizontal x-axis is divided by 251 uniformly spaced nodes; while the y-axis is partitioned with 101 nodes. This is shown in Figure 4.4.12.

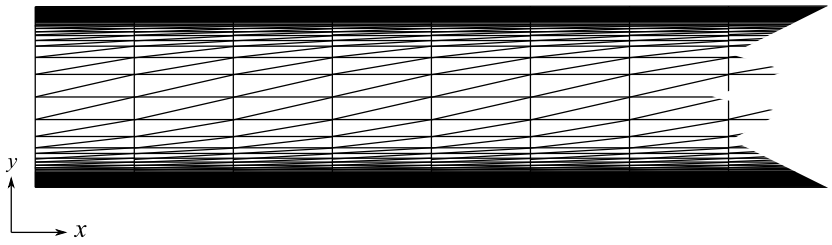

Figure 4.4.12: Two-dimensional graded mesh $\mathcal{T}_h$, $50,000$ triangular elements.

We follow the experimental conditions prescribed in [107], setting the peak inlet velocity $u_{\text{peak}} = 11.8\,\text{cm/s}$, and the kinematic viscosity $\upsilon = 0.008930\,\text{cS}$. We implement the

continuation algorithm outlined in Section 4.4.3, setting the initial kinematic viscosity $\upsilon_{\text{initial}} = 1.0\,\text{cS}$. We then reduce this value by 10% for each successive solve, increasing the Reynolds number until the target value of $Re = 2,777$. Finally, the axial velocity is sampled by way of a cross sectional slice located at $x = 1016\,\text{cm}$, 200 channel heights downstream from the inlet. The velocity is made dimensionless by calculating u^+, then plotted against the scaled distance from the wall y^+ in Figure 4.4.13.

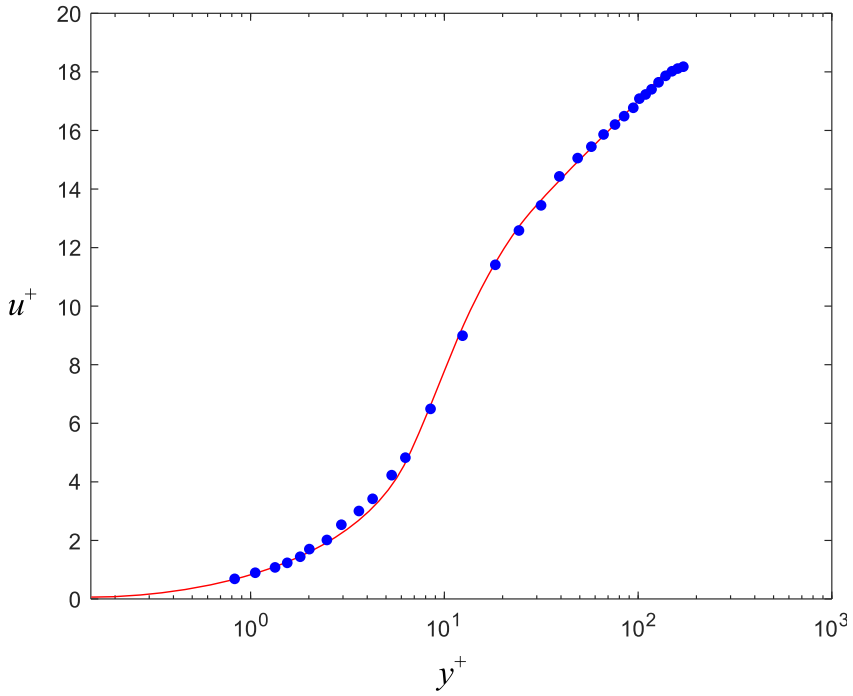

Figure 4.4.13: Semi-log plot comparing the scaled axial velocity u^+, to the scaled distance from the domain wall y^+, $Re = 2,777$. The experimental data [107] is shown with blue dots, while the DGFEM solution is represented by a solid red line.

The numerical solution is a good approximation the experimental data, especially towards the centre of the channel. However, the solution also appears to under approximate the flow velocity close to the channel wall. This could be a result of the choice of numerical flux, indicating that a new choice may need to be considered as we move to consider more complicated geometries. On the other hand, since we are considering a two-dimensional approximation to the three-dimensional experiment, the variation may simply be due to some subtle three-dimensional flow features. In general, Figure 4.4.13 suggests that the turbulence model and numerical flux are combining to produce numerical solutions that are representative of the physical experiment, correctly representing the near wall behaviour of the flow.

The implementation of the continuation algorithm proposed in Section 4.4.3 has proven to be very successful for this simple geometry, ensuring that the nonlinear solver converged for each of the intermediary Reynolds numbers up to the target value. This is shown in Table 4.2, which details the number of iterations required by the nonlinear Newton solver at particular Reynolds numbers in the continuation process.

Reynolds Number Re	Number of Newton iterations
24 (Laminar model)	4
24 (Turbulent model)	6
100	3
500	3
$1,900$	4
$2,100$	4
$2,777$	3

Table 4.2: Number of Newton iterations required for each Reynolds number, Re.

As shown in Table 4.2, the number of Newton iterations is reduced significantly by the use of a more accurate solution estimate. Once the initial turbulent solution is obtained, $Re = 24$, the number of nonlinear iterations reduces to around $2 - 3$ for the majority of the continuation process. There is an increase in the number of nonlinear iterations around the turbulent transition value, $Re \approx 2,000$, as the flow undergoes a number of physical changes. This increase in the number of Newton iterations is driven by a decrease in accuracy of the solution estimate compared to the numerical solution, thus requiring additional nonlinear iterations at these Reynolds numbers. However, as expected, once the flow is fully turbulent, the number of iterations returns to the usual number.

The continuation algorithm is refined and developed as the geometries become more complicated, as well as to include h-refinement passes in Chapter 6 and 7. However, the performance in these initial experiments, indicates that it may address the concerns voiced by industry representatives regarding DGFEM for turbulent high Reynolds number flows.

4.5 Concluding Remarks

We have considered an implementation of the $k - \omega$ turbulence model for the RANS equations, using the interior penalty DGFEM framework. The choice of turbulence model was clear when we considered both the preferences of industry representatives [93], as well as the $k-\omega$ models straight-forward implementation, being able to apply it throughout the

boundary layer without near-wall modifications. This greatly simplified the development of the DGFEM, allowing us to employ the techniques discussed in [25, 26, 118]. We use $\tilde{\omega} = \ln(\omega)$ to ensure a smoother near-wall distribution of ω , as well as, $\overline{k} = \max(k, 0)$ and $\tilde{\omega}_r = \max(\tilde{\omega}, \tilde{\omega}_{r_0})$, such that we guarantee the positivity of the normal turbulent stresses. These improvements enhance the stability of the numerical solver, with Figure 4.4.13 demonstrating that the DGFEM implementation of the turbulence model correctly represents the physics of turbulent flows, with accurate near wall flow velocities.

The initial test cases presented in Section 4.4 demonstrate the performance of the DGFEM solver for simple turbulent flows. Overall, they suggest that the choice of numerical flux is suitable for low to medium Reynolds number flows. In particular, the numerical solution demonstrates good agreement with the expected velocity profile in Figure 4.4.10, and the experimental data in Figure 4.4.13, despite the two-dimensional approximation of the flow. However, while the Lax-Friedrichs flux appears to be suitable for these simple geometries, further testing is required in order to validate its use on industrial test cases. As such, in Chapter 5, we move to consider more complicated geometries at higher Reynolds numbers.

The continuation algorithm proposed in Section 4.4.3 is effective at enhancing the numerical solver, enabling the reliable simulation of turbulent flows. The difficulty arises when we consider high Reynolds number flows, as they require numerical solutions to be found for extremely stiff systems. This means that we are unable to simply input the flow parameters and find the desired solution, as the nonlinear solver fails to converge to a solution. The initial estimate of the numerical solution is too different from the actual numerical solution. As such, we use low Reynolds number solutions to approximate high Reynolds number ones. We remark, that once an initial low Reynolds number turbulent solution has been found, the amount by which the Reynolds number can be increased for the next solve, such that we still ensure that the nonlinear solver converges, is unclear. In these initial test cases, we found that a value of 10% worked well, with larger values causing the nonlinear solver to converge slowly or not all. This problem was worst around the Reynolds numbers associated with the transition of the flow from laminar to turbulent, suggesting that the value by which the Reynolds number is increased should not be constant. We reassess the algorithm when we consider more complicated geometries and h-refinement in Chapter 6, with further adjustments for polytopic meshes in Chapter 7.

Chapter 5

Curvilinear Elements

In this chapter, we set out to build upon the results of the previous chapter by developing a computationally inexpensive approach to curvilinear elements. DGFEMs offer many advantages over current generation industrial FVM solvers, among which is there ability to handle complex curved geometries. As discussed previously, near wall representation of the flow is crucial to producing an accurate numerical approximation to the average flow, since the value of ω changes very rapidly as we approach the wall. Therefore, the use of elements with polynomial isosurfaces as faces is of much interest to industry representatives, since the current methods typically employ faceting of many linear elements in areas of high curvature. This is both ineffective and computationally expensive, especially in the present context of RANS.

5.1 Mesh Design

Consider a domain $\Omega \subset \mathbb{R}^d$, and let $\mathcal{T}_h$ be a triangulation of Ω consisting of quadrilateral elements $\kappa \in \mathcal{T}_h$. Previously, we only considered a single affine mapping, F_κ, between the reference element $\hat{\kappa}$, and the physical element $\kappa \in \mathcal{T}_h$. Now let us further consider some transitional element $\tilde{\kappa}$ and another transformation mapping Q_κ, such that $Q_\kappa\left(F_\kappa\left(\hat{\kappa}\right)\right) = Q_\kappa\left(\tilde{\kappa}\right) = \kappa$. In order to achieve the desired curvilinear physical element, let F_κ produce a standard anisotropic element by way of an affine mapping, whilst Q_κ is a diffeomorphism such that it produces the correct warping of $\tilde{\kappa}$ to represent κ. Roughly speaking, therefore, F_κ introduces the size of the element and Q_κ perturbs its shape, but does not change its size substantially, as shown in Figure 5.1.1.

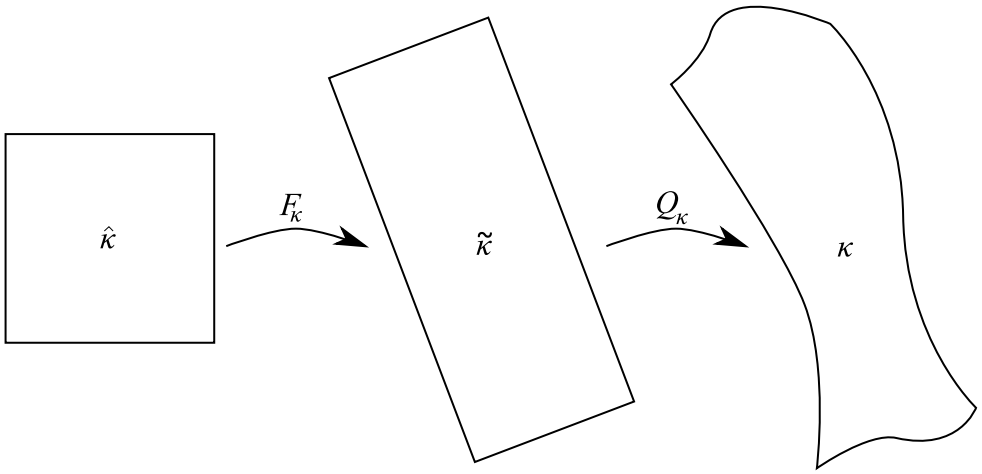

Figure 5.1.1: The mappings F_κ from the reference element $\hat{\kappa}$ to the transitional element $\tilde{\kappa}$, and Q_κ from $\tilde{\kappa}$ to the physical element κ.

Further, let $\hat{f}$, $\tilde{f}$ and f denote a face on the element $\hat{\kappa}$, $\tilde{\kappa}$ and κ, respectively. Also, let $\partial\kappa$ be the set of all element faces of the triangulation $\mathcal{T}_h$, such that $f \in \partial\kappa$.

5.2 The Discontinuity-Penalisation Function

Next, we consider in greater detail the discontinuity-penalisation function denoted by σ in (4.3.4). Following the previous notation, we define the discontinuity-penalisation function as $\sigma\colon \partial\kappa \to \mathbb{R}$. The precise definition of σ is dependent upon the local mesh size and the local polynomial degree. Assuming that the mesh $\mathcal{T}_h$ is shape regular, and the polynomial degree is constant over $\mathcal{T}_h$, we chose $\sigma = O\left(\frac{m^2}{h}\right)$ in (4.3.4), where, m is the polynomial degree and h is the mesh size. Generally speaking, in order to ensure convergence of the proposed DGFEM, σ must be chosen sufficiently large; however, smaller values of σ can lead to better conditioning of the stiffness matrix and also, possibly, smaller numerical error in certain instances. The remainder of this chapter is dedicated to discussing a balanced choice of σ such that the DGFEM remains stable when we introduce curvilinear elements into the mesh $\mathcal{T}_h$. A good choice of σ is crucial for the success of the proposed approach.

5.3 Inverse Estimates

Inverse estimates are widely used in the error analysis of numerical methods for the discretisation of PDEs. Consider a simplicial or tensor product element $\kappa \in \mathcal{T}_h$, and let $\left(X(\kappa), \|\cdot\|_{X(\kappa)}\right)$ and $\left(Y(\kappa), \|\cdot\|_{Y(\kappa)}\right)$ be two normed function spaces such that $\mathcal{R}_m \subset X(\kappa) \subset Y(\kappa)$. Then, classical inverse estimates are bounds of the form

$$\|v\|_{X(\kappa)} \leq C(C_r, m)\, h_\kappa^{-s} \|v\|_{Y(\kappa)} \tag{5.3.1}$$

for all $v \in \mathcal{R}_m$ and some $s \geq 0$. The constant $C(C_r, m)$ depends at most on the shape regularity constant C_r and the polynomial degree m. Clearly if $m \to \infty$, then we expect $C(C_r, m) \to \infty$. For specific knowledge of the dependence of the constant $C(C_r, m)$ on the polynomial degree m for various pairs of spaces $X(\kappa)$ and $Y(\kappa)$, we refer to [34, 42, 124, 139] and the references cited therein.

In order to progress with our analysis of the penalty parameter, we recall Lemma 2.8.4, in particular

$$\|\hat{v}\|_{\hat{f}}^2 \leq C_{\text{inv}} m^2 \|\hat{v}\|_{L^2(\hat{\kappa})}^2, \tag{5.3.2}$$

which is of a similar form to (5.3.1). Then, for element $\kappa \in \mathcal{T}_h$, we have

$$\|\hat{v}\|_{\hat{f}}^2 = \int_f v^2 \left|J_{F_f}^{-1}\right| \left|J_{Q_f}^{-1}\right| \,\mathrm{d}s \leq C_{\text{inv}} m^2 \int_\kappa v^2 \left|J_{F_\kappa}^{-1}\right| \left|J_{Q_\kappa}^{-1}\right| \,\mathrm{d}\mathbf{x} = C_{\text{inv}} m^2 \|\hat{v}\|_{L^2(\hat{\kappa})}^2. \tag{5.3.3}$$

Using m_f and m_κ to denote the volume of the face f and the volume of the element κ, respectively. The Jacobians of the affine mapping F_κ and F_f may be represented as $\left|J_{F_\kappa}^{-1}\right| = \frac{m_{\widehat{\kappa}}}{m_{\widetilde{\kappa}}}$ and $\left|J_{F_f}^{-1}\right| = \frac{m_{\widehat{f}}}{m_{\widetilde{f}}}$, since this is essentially a scaling of the element κ and face f. As $\left|J_{Q_\kappa}^{-1}\right|$ and $\left|J_{Q_f}^{-1}\right|$ are not constant, we consider the maximum and minimum values in order to bound the left- and right-hand sides.

We bound the left-hand side of (5.3.3) from below:

$$\|\hat{v}\|_{\hat{f}}^2 \geq \min_{\mathbf{x} \in f} \left|J_{Q_f}^{-1}\right| \frac{m_{\widehat{f}}}{m_{\widetilde{f}}} \int_f v^2 \,\mathrm{d}s,$$

and rearrange to the form

$$\|\hat{v}\|_{\hat{f}}^2 \geq \frac{1}{\max\limits_{\mathbf{x} \in f} \left|J_{Q_f}\right|} \frac{m_{\widehat{f}}}{m_{\widetilde{f}}} \int_f v^2 \,\mathrm{d}s.$$

Similarly, we may bound the right-hand side of (5.3.3) from above

$$\|\hat{v}\|^2_{L^2(\hat{\kappa})} \leq \frac{1}{\min\limits_{\mathbf{x}\in\kappa} |J_{Q_\kappa}|} \frac{m_{\widehat{\kappa}}}{m_{\widetilde{\kappa}}} \int_\kappa v^2 \, \mathrm{d}\mathbf{x}.$$

Finally, combining the above upper and lower bounds together and applying (5.3.3), we are left with

$$\frac{1}{\max\limits_{\mathbf{x}\in f} \left|J_{Q_f}\right|} \frac{m_{\widehat{f}}}{m_{\widetilde{f}}} \|v\|^2_f \leq C_{\mathrm{inv}} m^2 \frac{1}{\min\limits_{\mathbf{x}\in\kappa} |J_{Q_\kappa}|} \frac{m_{\widehat{\kappa}}}{m_{\widetilde{\kappa}}} \|v\|^2_{L^2(\kappa)},$$

or

$$\|v\|^2_f \leq C_\gamma m^2 \|v\|^2_{L_2(\kappa)}, \tag{5.3.4}$$

with

$$C_\gamma = C_{\mathrm{inv}} \frac{\max \left|J_{Q_f}\right|}{\min |J_{Q_\kappa}|} \frac{m_{\widetilde{f}}}{m_{\widetilde{\kappa}}} \frac{m_{\widehat{\kappa}}}{m_{\widehat{f}}}.$$

Since (5.3.4) is derived from the inverse estimate taken in the isotropic setting, we may therefore define the new penalty parameter such that $\sigma = C_\sigma \frac{m^2}{h}$, with $C_\sigma \geq C_\gamma C_f$. Here, C_f is the cardinality of the element neighbours, defined as $C_f = \max\limits_{\kappa\in\mathcal{T}_h} \mathrm{card}\,\{f \in \Gamma_{\mathrm{int}} \cup \Gamma_{\mathrm{D}} : f \subset \partial\kappa\}$. Hence, C_σ is of the same form as the standard isotropic penalty parameter derived in Section 2.8, but is able to account for the curvature of the element.

Since Q_κ is a nonlinear diffeomorphism, the Jacobians $|J_{Q_\kappa}|$, $\left|J_{Q_f}\right|$ are not constant with respect to the spatial variables. Thus, in order to ensure that (5.3.4) holds, we must find a bound on the term $\dfrac{\max \left|J_{Q_f}\right|}{\min |J_{Q_\kappa}|}$. We do this by limiting the choice of Q_κ to polynomial functions with real coefficients. This ensures that the Jacobians exist for all points $\mathbf{x} \in \kappa \cup \partial\kappa$, and that κ is a non-degenerate element. Therefore, simply by the properties of polynomials, the term $\dfrac{\max \left|J_{Q_f}\right|}{\min |J_{Q_\kappa}|}$ is bounded from above and below on the domain κ, as all the entries in the Jacobian matrices J_{Q_f} and J_{Q_κ} are polynomials. Note that the degree of polynomials in Q_f can be sufficiently high to ensure general enough element shapes to capture curved geometries.

The only remaining question is how to fix a value of C_{inv}. We refer to Schwab [124], who places a lower bound of $C_{\mathrm{inv}} = \sqrt{6}$, for a two-dimensional quadrilateral reference element $\hat{\kappa}$. In order to ensure coercivity, we select $C_{\mathrm{inv}} = 2.5$, following the consensus in the literature of $C_\sigma = 10$ for isotropic mesh elements.

Within the literature there exists two distinct choices for the discontinuity-penalisation function: one suitable for isotropic elements, defined as $\sigma = C_\sigma \frac{m^2}{h}$, and the anisotropic

parameter detailed in [67, 77]. By setting $C_\gamma = C_{\text{inv}} \frac{m_f}{m_\kappa}$ in the approach proposed above, where $m_f = \min\{m_{f_1} ... m_{f_d}\}$, $i = 1, ..., d$, we recover the anisotropic penalty parameter of [67, 77]. Indeed, [77] shows that this parameter is stable for affine mappings combined with C^1-diffeomorphisms, as long as the Jacobian of the diffeomorphism is sufficiently close to the identity matrix. Therefore, for a discontinuity-penalisation function defined by

$$\sigma = C_\sigma \frac{m^2}{h},\ C_\sigma \geq C_\gamma C_f;$$

we recover the isotropic parameter if

$$C_\gamma = C_{\text{inv}},$$

and the anisotropic parameter if

$$C_\gamma = C_{\text{inv}} \frac{m_f}{m_\kappa}.$$

5.4 Numerical Experiments

With the addition of a more precise discontinuity-penalisation function, we are able to consider more complex geometries, representing domain boundaries with curvilinear elements. In particular, we allow the use of cubic polynomials to approximate element boundaries, allowing for meshes to be generated from point cloud data using cubic splines. In the case of industrial meshes, machine parts are typically designed with piece-wise quadratic boundaries, and, as such, we normally only need to consider elements with quadratic faces. We note here, that for the following test cases, we limit the use of curvilinear mesh elements to those whose faces intersect the domain boundary, i.e., $\partial\kappa \subset \partial\Omega$.

We introduce the industrial numerical test case in Section 5.4.1, the 1.5 Stage Aachen Turbine [135, 138, 65]. We consider the flow through a stator passage, as we are, presently, unable to represent the mixing plane required to transition from a stationary to rotational flow. However, this serves to validate the DGFEM on industrial type meshes. Addtionally, in Section 5.4.2, we consider a three-dimesnsional backwards-facing step test case, in order to validate the DGFEM scheme on three-dimensional meshes.

5.4.1 Turbine Stator Cascade

We consider the flow through the 1.5 Stage Aachen Turbine [135, 138, 65]; a reference gas turbine used for power generation. The 1.5 stage has cylindrical end walls, with hub and tip radii of 0.245 m and 0.30 m, respectively. Further, the turbine blade span, $H = 0.055$ m.

The stator blades are pitchwise stacked at the trailing edge, while the rotor blades are pitchwise stacked at their centroid, located at $x = 0.025265\,\text{m}$ and $y = 0.013456\,\text{m}$ from the rotor blade leading edge. The stator blades have a chord length $c_s = 0.062\,\text{m}$, an axial chord length $c_x = 0.04425\,\text{m}$, and a pitch to chord ratio $\frac{s_s}{c_s} = 0.77$; giving 36 blades around the cascade. The rotor blades have a chord length $c_r = 0.060\,\text{m}$ and a pitch to chord ratio $\frac{s_r}{c_r} = 0.67$; giving 41 blades around the cascade. The downstream stator blades are clocked by 3° in the positive direction of rotation of the rotor, with respect to the upstream stator blades.

The two-dimensional cacsade schematic is shown in Figure 5.4.1. In this numerical example, we consider the flow through the first stator passage only, as inclusion of the rotar passage and second stator, would require the developement of a mixing plane to transition the flow from a stationary to rotaional frame of reference; which is beyond the scope of this work.

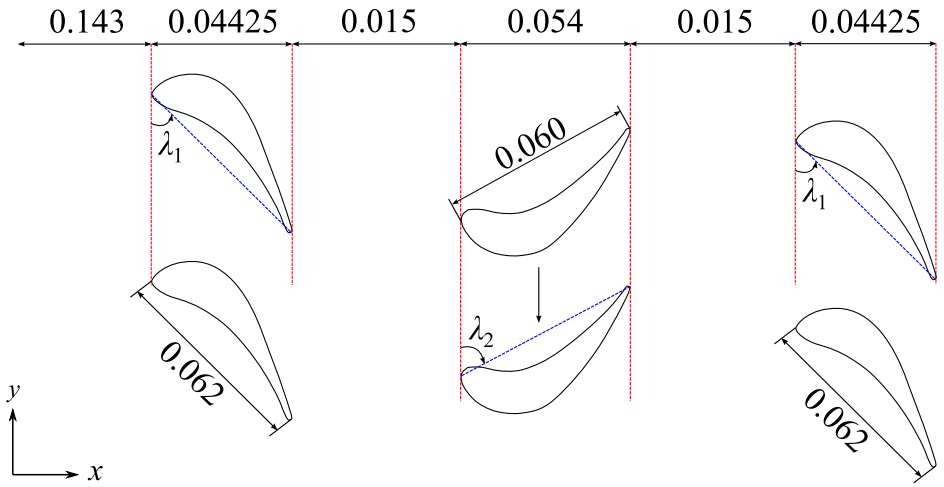

Figure 5.4.1: Schematic of the Aachen Turbine on the cascade plane [138]. All lengths are stated in metres. The angle subtended by the y-axis and the stator and rotor chords are $\lambda_1 = 45.5°$ and $\lambda_2 = 62°$ respectively.

In particular, we consider the mid-height flow between the initial stator baldes, with a pitch $s_s = 0.0476\,\text{m}$. The initial stator section is designed to receive an inlet flow parallel to the horizontal axis, such that the flow is axial. The design Reynolds number, based on the chord length and exit velocity is $Re = 6.8 \times 10^5$.

The mesh for this test case consists of an inlet channel of length $l_{in} = 0.143\,\text{m}$, and an outlet channel of length $l_{out} = 0.18575\,\text{m}$. The outlet length is chosen such that the flow may fully develop before passing through the domain outflow boundary. The blade passage is specified using a cloud of point data [137], detailing the dimensions of the pressure and

suction side of the blade. These data points are then used as nodes to create piecewise-cubic splines to approximate the curvature of the physical stator blade. Figure 5.4.2 details the geometry of the stator passage to be simulated. The pressure side of the blade (the upper boundary), requires that a discontinuity be introduced into the spline at the point $x = 0.04398\,\text{m}$ and $y = -0.00042\,\text{m}$, due to the shape of the boundary. As such, two separate cubic splines are required to completely define the geometry of the pressure side.

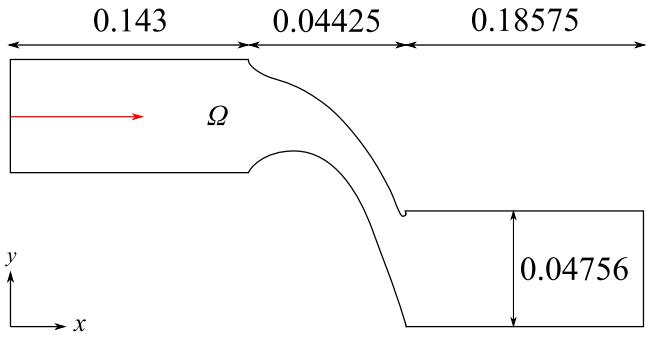

Figure 5.4.2: Geometry of a single blade domain for a stator cascade. All lengths are stated in metres.

The computational domain Ω is subdivided into a mesh $\mathcal{T}_h$, consisting of $53,000$ quadrilateral elements. Elements with faces that intersect the domain boundary are chosen to have cubic approximations for these faces, matching closely the geometry of the stator blade. That is to say, the blade geometry is represented by piecewise cubic polynomials. The computational mesh $\mathcal{T}_h$ is shown in Figures 5.4.3 and 5.4.4.

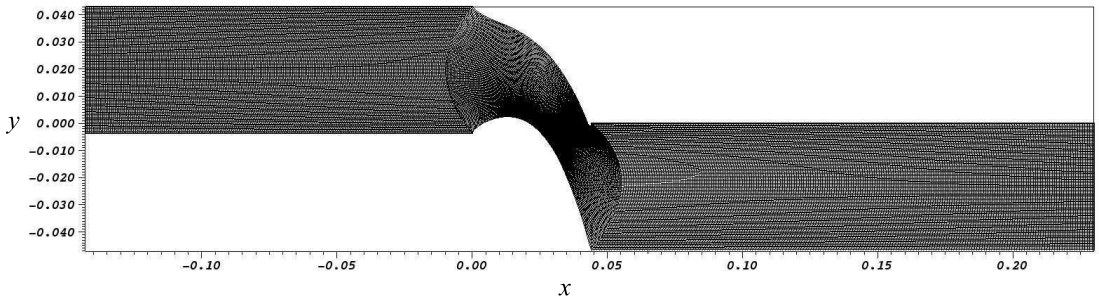

Figure 5.4.3: Computational mesh for the Aachen Turbine stator passage, $(53,000$ elements).

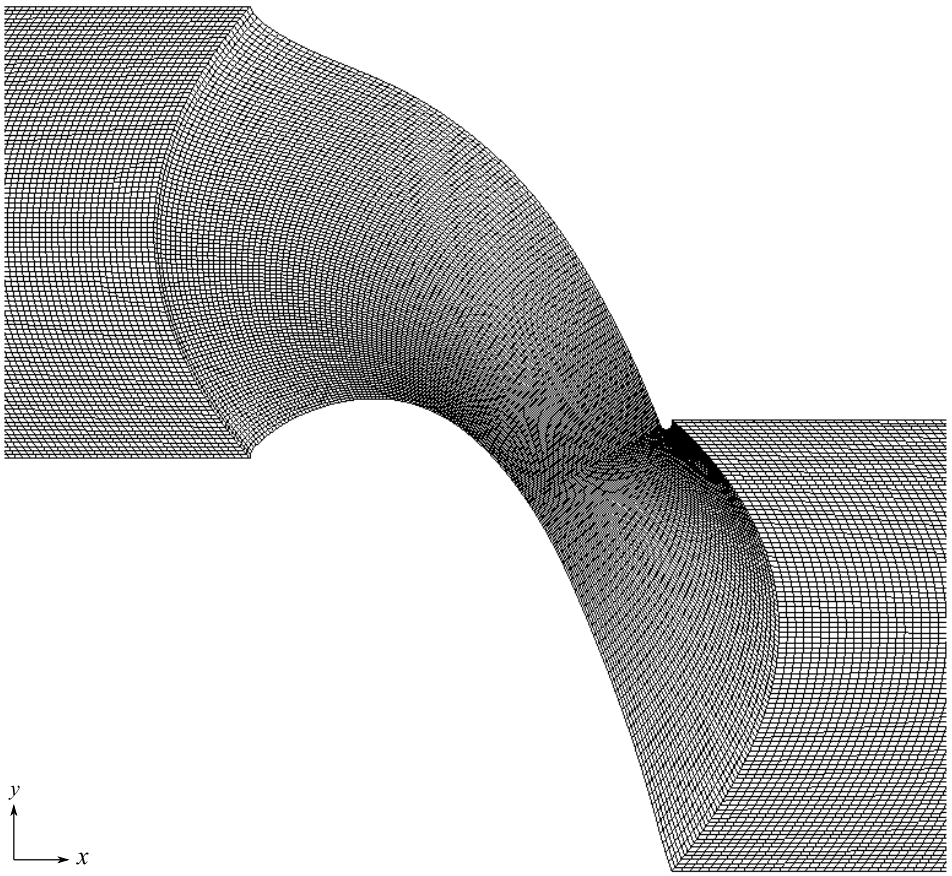

Figure 5.4.4: Central portion of the computational mesh.

Periodic boundary conditions, denoted Γ_{P}, are introduced for the straight horizontal boundaries, representing the geometry of the three-dimensional stator section of the turbine. In particular, we set

$$\begin{aligned} \mathbf{u}(x, 0.043047) &= \mathbf{u}(x, 0.004052) \text{ on } \Gamma_{\mathrm{P}_1}, \\ \mathbf{u}(x, 0) &= \mathbf{u}(x, -0.047560) \text{ on } \Gamma_{\mathrm{P}_2}. \end{aligned} \tag{5.4.1}$$

The blade edges are defined using a no-slip condition (3.1.2), and the stress-free condition (3.1.3) is used on the domain outlet located at $x = 0.23\,\mathrm{m}$. The domain inlet located at $x = -0.143\,\mathrm{m}$, is a turbulent free-stream condition with an average inlet velocity $u_\infty = 45.0\,\mathrm{m/s}$ [13]. These are summarised in Figure 5.4.5.

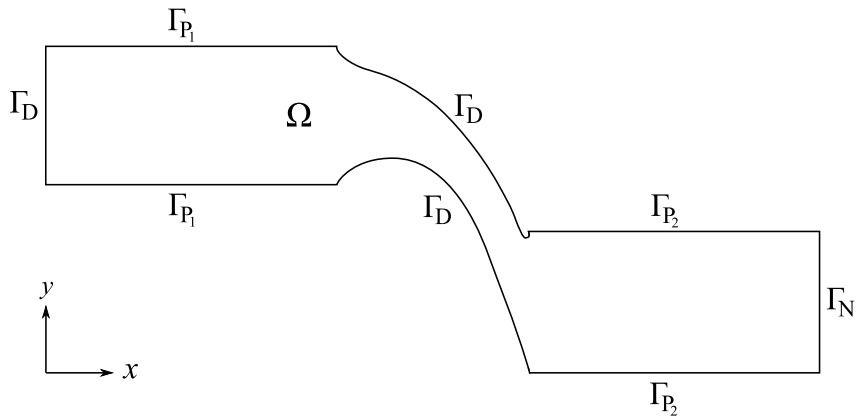

Figure 5.4.5: Boundary diagram of the computational domain.

We employ the continuation algorithm outlined in Section 4.4, increasing the Reynolds number by 10% between successive nonlinear solves until the target Reynolds number $Re = 6.8 \times 10^5$ is reached. In particular, we use the Newton based iterative solver detailed in Section 3.4, where the stopping condition is defined such that the l_2-norm of the nonlinear residual vector is reduced by 8 orders of magnitude. Each linear problem that arises is handled by the MUMPS nonsymmetric direct solver in combination with a ParMETIS reordering of the associated Jacobian matrix, along with the libraries supplied by the OpenBLAS software.

To better approximate the blade edges, we use cubic approximations for the element faces along the blade boundary, stabilising the jumps via the proposed curvilinear penalty parameter. We present colour plots of the simulated flow in Figures 5.4.6, 5.4.7 and 5.4.8; with a Reynolds number $Re = 6.8 \times 10^5$.

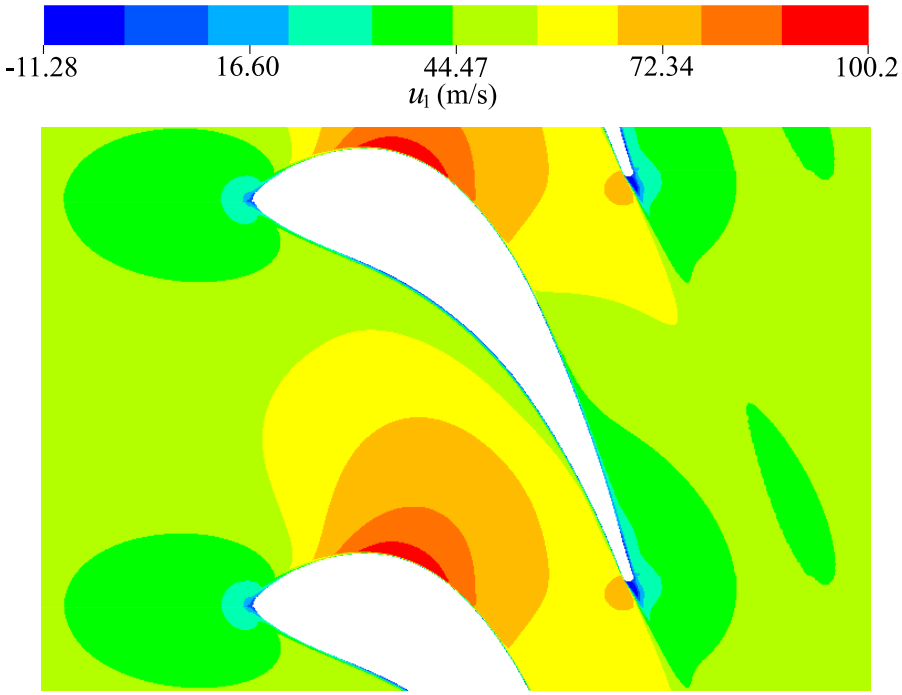

Figure 5.4.6: Colour plot of the axial velocity u_1, computed on $\mathcal{T}_h$. $Re = 680,000$.

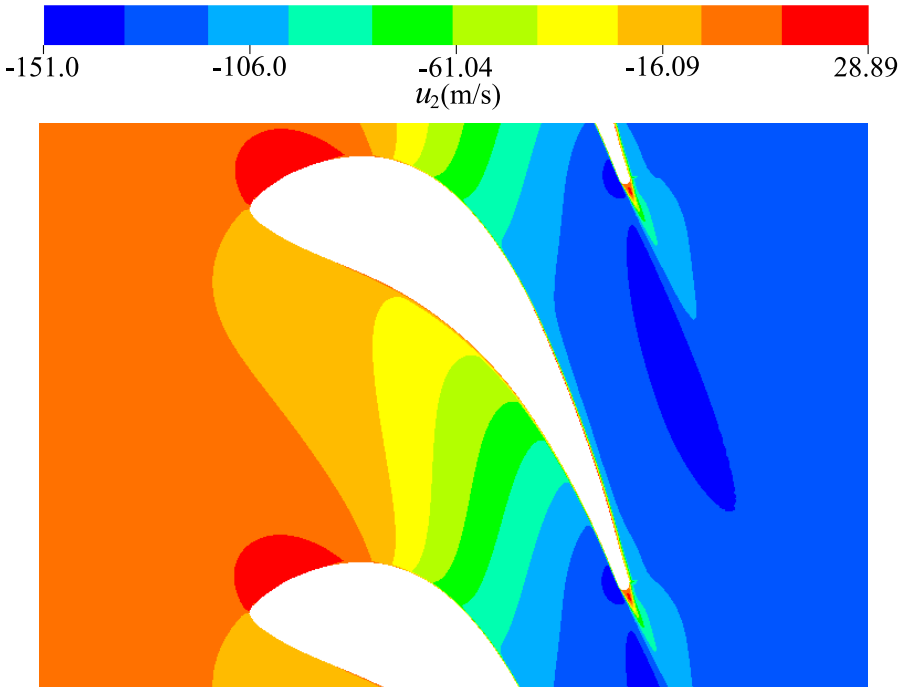

Figure 5.4.7: Colour plot of the vertical velocity u_2, computed on $\mathcal{T}_h$. $Re = 680,000$.

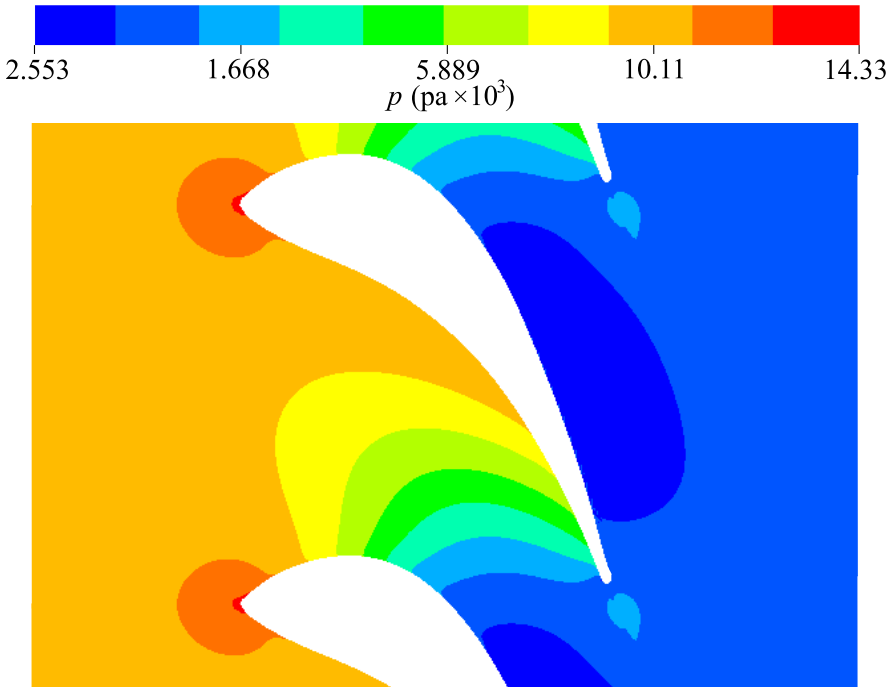

Figure 5.4.8: Colour plot of the static pressure p, computed on $\mathcal{T}_h$. $Re = 680,000$.

The computed solutions approximate the turbulent flow well, capturing the long thin boundary layer. In particular, the absolute pitchwise flow angle at the midspan of the blades, measured in the centre of the flow passage, 0.008 m behind the first stator, is observed to be 19.4° [100]. Sampling the data in Figures 5.4.6 and 5.4.7, the numerical simulation approximates this value to be 19.86°. This approximation is reasonably close to the experimental value, especially as in the literature [100], the value of the flow angle is often overestimated by steady state numerical models. In Chapter 6, we see if this approximation can be improved by way of adaptive mesh refinement.

To illustrate the effectiveness of choosing the penalty parameter using the technique described as opposed to the traditional isotropic or anisotropic parameter, we consider flows through the stator cascade in Figures 5.4.9, 5.4.10 and 5.4.11, with target Reynolds numbers $9,000$, $36,000$ and $680,000$, respectively. The continuation algorithm allows the Reynolds number to increase by 10% between successive nonlinear solves. As such, we compute solutions for the continuation algorithm using the curvilinear parameter, then, 5 solves before the target Reynolds number is reached, we switch to the penalty parameter that is being tested. If the initial estimate of the nonlinear solver is too similar to the numerical solution at the new Reynolds Number, then the nonlinear solver will overcome the instabilities associated with using the incorrect penalty parameter and converge regardless.

By considering the 5 previous Reynolds numbers, we achieve a better representation of the perfomance of the different parameters on meshes with curved boundaries. Additionally, for this testing, the stopping condition of the nonlinear solver is defined such that the l_2-norm of the nonlinear residual vector is reduced by 6 orders of magnitude.

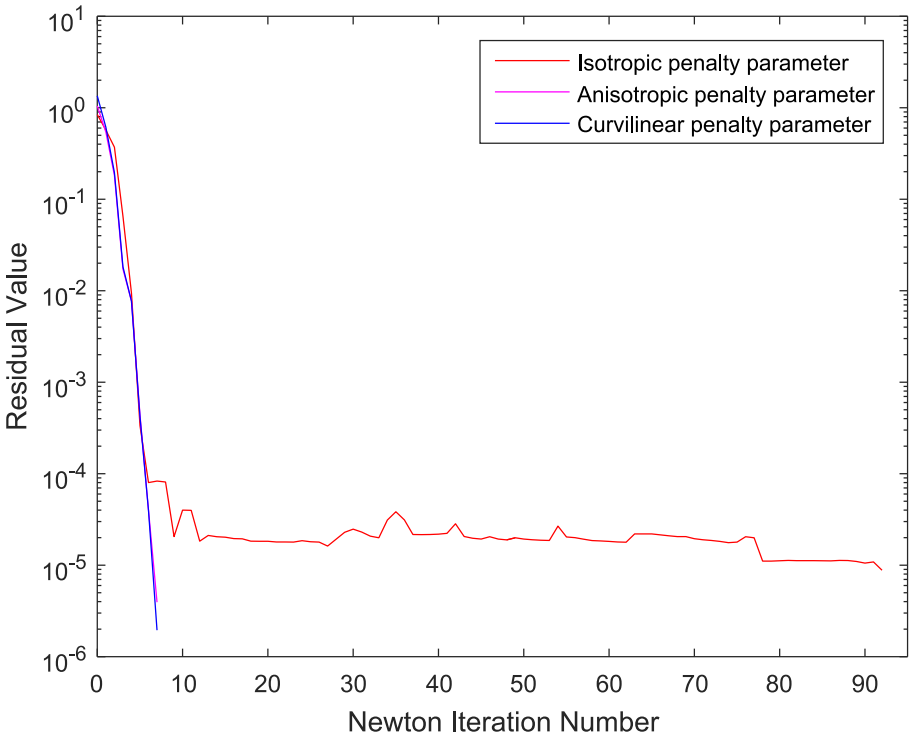

Figure 5.4.9: Newton solver residual values for isotropic, anisotropic and curvilinear penalty parameters, $Re = 9,000$.

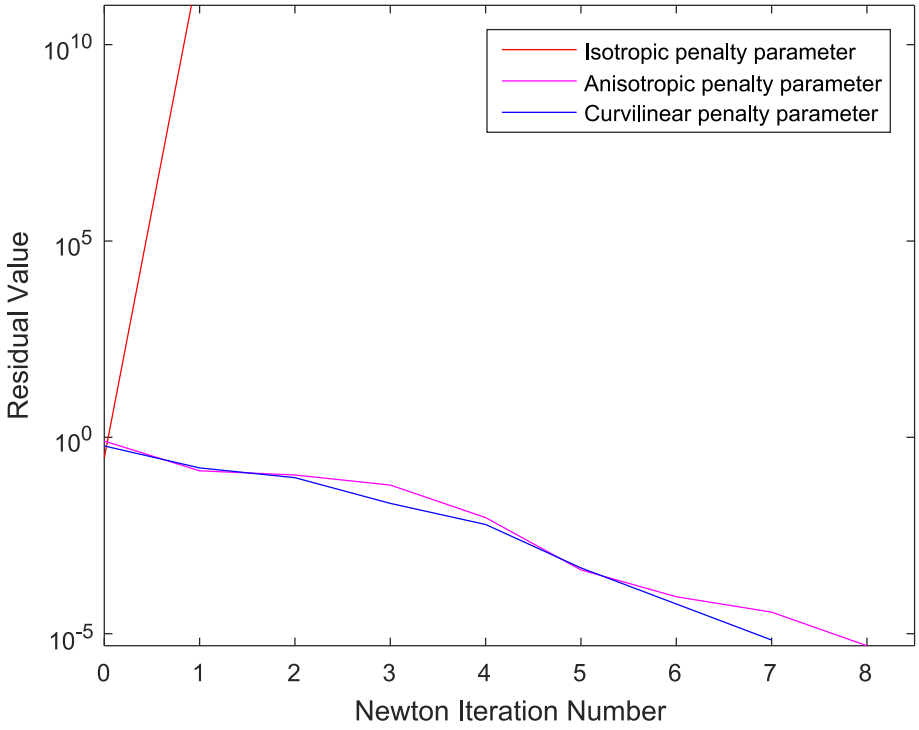

Figure 5.4.10: Newton solver residual values for isotropic, anisotropic and curvilinear penalty parameters, $Re = 36,000$.

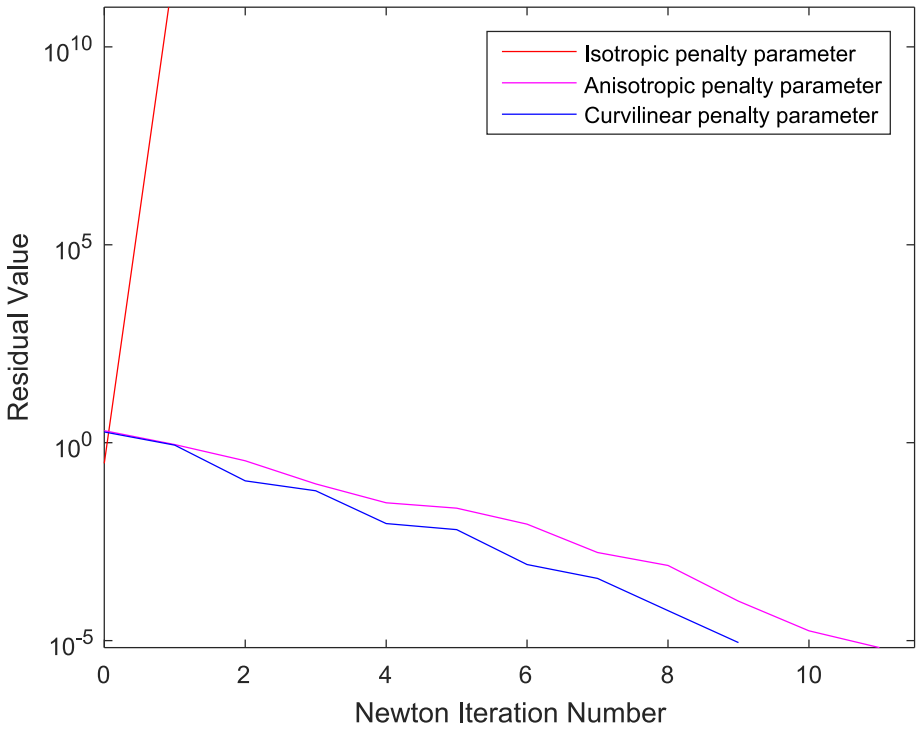

Figure 5.4.11: Newton solver residual values for isotropic, anisotropic and curvilinear penalty parameters, $Re = 680,000$.

Figure 5.4.8 shows a similiar performance between the anisotropic and curvilinear penalty parameters, with the isotropic parameter inhibiting the efficiency of the nonlinear solver. This is due to the overwhelmingly large proportion of anisotropic type elements in the computational mesh, (Figure 5.4.3), compared to isotropic type elements. As such, the isotropic parameter is unable to stabilise the DGFEM sufficiently. Further, when the Reynolds number increases, the small proportion of curvilinear elements, combined with the increased stiffness of the mass matrix due to the higher Reynolds numbers, means that a disparity emerges between the anisotropic and curvilinear parameters. Due to the small proportion of curvilinear elements compared to anisotropic elements, the anisotropic parameter performs reasonably well, stabilising the method enough to ensure solution convergence. However, the curvilinear parameter provides better conditioning of the stiffness matrix, meaning that solutions require fewer nonlinear iterations. This is shown in Figures 5.4.10 and 5.4.11.

5.5 Concluding Remarks

In Chapter 5, we have demonstrated that it is possible to create a highly efficient implementation of the Interior Penalty DGFEM method that is stable on meshes with curvilinear elements. In particular, the proposed alterations to the definition of the penalty parameter (5.3.4) stabilise the DGFEM for elements with general polynomial faces. Additionally, to do so, we do not require the computation of any new quantities which are not already calculated by the computer implementation of the DGFEM method. As such, we achieve a far better representation of the domain geometry, avoiding the need to facet large numbers of standard straight edged elements in areas of the mesh with high curvature boundaries.

To demonstrate the effectiveness of choosing the penalty parameter in this way, we considered as an industrial relevant test case, the flow through the 1.5 Stage Aachen Turbine stator cascade in Section 5.4.1. We showed that the continuation algorithm proposed in Section 4.4.3, could be successfully applied to high Reynolds number turbulent flows through complex geometries. Figures 5.4.9, 5.4.10 and 5.4.11 demonstrate the benefits of choosing the penalty parameter in this way for meshes with curved elements. As the Reynolds number, and, consequently, the stiffness of the problem increases, the method by which the penalty parameter is chosen becomes more and more important. The stator mesh in Figure 5.4.3, contains a relatively low percentage of isotropic and curvilinear elements, compared to anisotropic elements. As such, the isotropic parameter failed to stabilise the curved mesh sufficiently, once the Reynolds number reached $Re \approx 10,000$. For the target Reynolds number $Re = 680,000$, there was a clear benefit to using the curvilinear parameter over the anisotropic one, reducing the number of iterations required by the nonlinear

solver. The apparent success of the anisotropic parameter in Figures 5.4.10 and 5.4.11, is due to the low percentage of curvilinear elements in Figure 5.4.3. However, if a larger percentage of elements were of the curvilinear variety, we would expect to see a far greater benefit to choosing the curvilinear parameter, with the anisotropic parameter performing much like the isotropic parameter does in Figures 5.4.9, 5.4.10 and 5.4.11.

Simulating high Reynolds number flows for industrial problems typically requires a fine packing of mesh elements close to blade boundaries in order to accurately represent the boundary layer. While, on the other hand, typically needing only a relatively coarse mesh compared to the boundary layer region, to represent laminar flow areas. An efficient method of generating these types of meshes is by using automated refinement strategies. However, the increased number of degrees of freedom comes with the trade off of extended computation time. Therefore, by combining the efficiency of curvilinear elements, which are better able to capture the underlying geometry of domain compared to straight edged elements, with a suitable refinement strategy, we seek to improve the overall efficiency of the DGFEM solver. This is the discussion of Chapter 6.

Chapter 6

Adaptive Mesh Refinement

In this chapter we present an adaptive mesh refinement technique based upon a goal-oriented, dual-weighted-residual (DWR) *a posteriori* error estimate. In particular, we seek to approximate the error with respect to a specified target functional, which in turn may be used to drive a mesh refinement algorithm through the minimisation of the this error. The choice of functional should be relevant to the user's need and the structure of the underlying PDEs. Examples include, point values, local averages, flux integrals of the solution, lift or drag coefficients of an aerofoil, and so on. The idea was originally introduced by Eriksson, Estep, Hansbo and Johnson [59], with subsequent contributions from Becker and Rannacher [30, 31] and others. The method is derived via a duality argument, comparable to those employed for the derivation of *a priori* error bounds for FEMs.

In the present context, the aim of an adaptive mesh refinement routine is to decrease the error of the target functional; which, if chosen correctly, will lead to better approximations of boundary layer flows around aerofoils.

6.1 DWR *a Posteriori* Error Estimation

We begin by considering the discontinuous Galerkin discretisation of the turbulent incompressible flow equations (4.3.4). We rewrite (4.3.4) in a concise form notationally, that is, find $\mathbf{u}_h \in \mathbf{V}_{h,m}$, such that for all $\mathbf{v}_h \in \mathbf{V}_{h,m}$, we have

$$\mathcal{N}(\mathbf{u}_h, \mathbf{v}_h) = 0. \tag{6.1.1}$$

Further, we assume that the discretisation (6.1.1) is consistent. That is to say, that the analytical solution $\mathbf{u}$ satisfies

$$\mathcal{N}(\mathbf{u}, \mathbf{v}) = 0$$

$\forall \mathbf{v} \in \mathbf{V}$. Here, $\mathbf{V}$ is a suitably chosen function space that includes the analytical solution $\mathbf{u}$ to (4.3.1) and (4.3.2), and satisfies $\mathbf{V}_{h,m} \subset \mathbf{V}$. Further to this, consider a target functional $J : \mathbf{V} \to \mathbb{R}$. In particular, $J(\mathbf{u})$ is the true value of the functional computed on the analytical solution $\mathbf{u}$. In order to derive an error representation formula for $J(\mathbf{u})$, we follow the procedure in [84]. Assuming that $J(\cdot)$ is differentiable, we introduce the mean value linearisation $\bar{J}(\cdot, \cdot; \cdot)$ of $J(\cdot)$

$$\bar{J}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h) = J(\mathbf{u}) - J(\mathbf{u}_h) = \int_0^1 J'[\theta\mathbf{u} + (1-\theta)\mathbf{u}_h](\mathbf{u} - \mathbf{u}_h)\, \mathrm{d}\theta. \tag{6.1.2}$$

Here $J'[\mathbf{w}](\cdot)$ denotes the Fréchet derivative of $J(\cdot)$ evaluated at some $\mathbf{w} \in \mathbf{V}$. We then proceed as in [84], introducing the mean value linearisation of $\mathcal{N}(\cdot, \cdot)$, given by

$$\begin{aligned} \mathcal{M}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h, \mathbf{v}) &= \mathcal{N}(\mathbf{u}, \mathbf{v}) - \mathcal{N}(\mathbf{u}_h, \mathbf{v}) \\ &= \int_0^1 \mathcal{N}'_{\mathbf{u}}[\theta\mathbf{u} + (1-\theta)\mathbf{u}_h](\mathbf{u} - \mathbf{u}_h, \mathbf{v})\, \mathrm{d}\theta \quad \forall \mathbf{v} \in \mathbf{V}. \end{aligned} \tag{6.1.3}$$

Similarly to before, $\mathcal{N}'_{\mathbf{u}}[\mathbf{w}](\cdot, \mathbf{v})$ denotes the Fréchet derivative of $\mathbf{u} \to \mathcal{N}(\mathbf{u}, \mathbf{v})$, for $\mathbf{v} \in \mathbf{V}$ fixed, and for some $\mathbf{w} \in \mathbf{V}$. In practice, a suitable approximation to $\mathcal{N}'_{\mathbf{u}}[\mathbf{w}](\cdot, \cdot)$ may need to be computed. For a more comprehensive review, we refer to [79, 80, 82], and the references cited therein. In order to proceed in this formal setting, we assume that the linearisation (6.1.3) is well-defined in this instance. As such, we introduce the dual problem: find $\mathbf{z} \in \mathbf{V}$ such that

$$\mathcal{M}(\mathbf{u}, \mathbf{u}_h; \mathbf{w}, \mathbf{z}) = \bar{J}(\mathbf{u}, \mathbf{u}_h; \mathbf{w}) \quad \forall \mathbf{w} \in \mathbf{V}. \tag{6.1.4}$$

Next, we assume that the solution to (6.1.4) is unique, noting that the validity of this assumption depends on the definition of $\mathcal{M}(\mathbf{u}, \mathbf{u}_h; \cdot, \cdot)$ and the choice of target functional [81]. Following [80], we have the subsequent error analysis, under the assumption that the dual problem (6.1.4) is well-posed.

Theorem 6.1.1. *Let* $\mathbf{u}$ *and* $\mathbf{u}_h$ *denote the solutions of* (4.0.1) *and* (4.3.4) *respectively, and suppose that the dual problem* (6.1.4) *is well-posed. Then the error representation formula*

is

$$J(\mathbf{u}) - J(\mathbf{u}_h) = -\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \mathbf{z}_h), \tag{6.1.5}$$

for all $\mathbf{z}_h \in \mathbf{V}_{h,m}$.

Proof. The proof here follows as in [80]. Choosing $\mathbf{w} = \mathbf{u} - \mathbf{u}_h$ in (6.1.4), recalling the linearisation performed in (6.1.2) and using the Galerkin orthogonality property [80], that is to say, by consistency, $\mathcal{N}(\mathbf{u}, \mathbf{v}_h) - \mathcal{N}(\mathbf{u}_h, \mathbf{v}_h) = 0$ for all $\mathbf{v}_h \in \mathbf{V}_{h,m}$, we have

$$\begin{aligned} J(\mathbf{u}) - J(\mathbf{u}_h) &= \bar{J}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h) = \mathcal{M}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h, \mathbf{z}) \\ &= \mathcal{M}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h, \mathbf{z} - \mathbf{z}_h) = -\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \mathbf{z}_h), \end{aligned}$$

for all $\mathbf{z}_h \in \mathbf{V}_{h,m}$. □

We now define, for each $\kappa \in \mathcal{T}_h$, the local error estimator η_κ, which contains the face and element residuals multiplied by the dual solution. Therefore, we have

$$J(\mathbf{u}) - J(\mathbf{u}_h) = -\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \mathbf{z}_h) \equiv \sum_{\kappa \in \mathcal{T}_h} \eta_\kappa. \tag{6.1.6}$$

where

$$\begin{aligned} \eta_\kappa = &\int_\kappa \mathbf{R}(\mathbf{u}_h) \cdot \boldsymbol{\phi}_h \, \mathrm{d}\mathbf{x} + \int_{\partial\kappa \setminus \Gamma} \left(\mathbf{F}^C(\mathbf{u}_h) \cdot \mathbf{n}_\kappa - \mathcal{H}\left(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa\right) \right) \cdot \boldsymbol{\phi}_h^+ \, \mathrm{ds} \\ &+ \int_{\partial\kappa \cap \Gamma} \left(\mathbf{F}^C(\mathbf{u}_h) \cdot \mathbf{n}_\kappa - \mathcal{H}\left(\mathbf{u}_h^+, \mathbf{u}_\Gamma\left(\mathbf{u}_h^+\right), \mathbf{n}_\kappa\right) \right) \cdot \boldsymbol{\phi}_h^+ \, \mathrm{ds} \\ &+ \frac{1}{2} \int_{\partial\kappa \setminus \Gamma} \left(\left(\mathbf{G}_\Gamma^{\mathrm{T}}\left(\mathbf{u}_h^+\right) \nabla_h \boldsymbol{\phi}_h^+ \right) : [\mathbf{u}_h] - \left[\mathbf{F}^V(\mathbf{u}_h, \nabla \mathbf{u}_h) \right] \cdot \boldsymbol{\phi}_h^+ \right) \mathrm{ds} \\ &- \int_{\partial\kappa \setminus \Gamma} \sigma[\mathbf{u}_h] : \boldsymbol{\phi}_h^+ \otimes \mathbf{n}_\kappa \, \mathrm{d}s - \int_{\partial\kappa \cap (\Gamma \setminus \Gamma_{\mathrm{N}})} \sigma\left(\mathbf{u}_h^+ - \mathbf{u}_\Gamma\left(\mathbf{u}_h^+\right)\right) \cdot \boldsymbol{\phi}_h^+ \, \mathrm{ds} \\ &- \int_{\partial\kappa \cap \Gamma_{\mathrm{N}}} \left(\mathbf{F}^V\left(\mathbf{u}_h^+ \nabla \mathbf{u}_h^+\right) \cdot \mathbf{n}_\kappa - g_{\mathrm{N}} \right) \cdot \boldsymbol{\phi}_h^+ \, \mathrm{ds} \\ &+ \int_{\partial\kappa \cap (\Gamma \setminus \Gamma_{\mathrm{N}})} \left(\mathbf{G}_\Gamma^{\mathrm{T}}\left(\mathbf{u}_h^+\right) \nabla_h \boldsymbol{\phi}_h^+ \right) : \left(\mathbf{u}_h^+ - \mathbf{u}_\Gamma\left(\mathbf{u}_h^+\right) \right) \otimes \mathbf{n}_\kappa \, \mathrm{ds}. \end{aligned} \tag{6.1.7}$$

where $\boldsymbol{\phi}_h = \mathbf{z} - \mathbf{z}_h$, $\forall \mathbf{z}_h \in \mathbf{V}_{h,m}$. $\mathbf{R}(\mathbf{u}_h) \mid_\kappa = -\nabla \mathbf{F}^C(\mathbf{u}_h) + \nabla \cdot \mathbf{F}^V(\mathbf{u}_h, \nabla \mathbf{u}_h)$, $\kappa \in \mathcal{T}_h$, denotes the elementwise residual.

A straightforward method of constructing meshes that are specifically designed for the efficient control of the error in the computed target functional $J(\cdot)$, is to apply the triangle inequality to the result of Theorem 6.1.1. This allows us to deduce the so-called weighted error estimator.

Theorem 6.1.2. *Let* $\mathbf{u}$ *and* $\mathbf{u}_h$ *denote the solutions of* (4.0.1) *and* (4.3.4) *respectively, and suppose that the dual problem* (6.1.4) *is well-posed. Then, the following error bound holds*

$$|J(\mathbf{u}) - J(\mathbf{u}_h)| \leq \sum_{\kappa \in \mathcal{T}_h} |\eta_\kappa| , \tag{6.1.8}$$

for all $\mathbf{z}_h \in \mathbf{V}_{h,m}$*, and for* η_κ *defined in* (6.1.6) *and* (6.1.7)*.*

We remark that the error representation formula (6.1.5) and the error bound (6.1.8) depend on the unknown analytical solutions $\mathbf{u}$ and $\mathbf{z}$, to the primal and dual problems, respectively. Therefore, the next section will be concerned with deriving suitable approximations to these quantities.

6.2 Approximating the Dual Solution

This section is dedicated to the discussion of the various methods by which we can make these approximations. We proceed as [80], defining $\hat{\mathbf{z}} \in \mathbf{V}$ to be the solution of the linearised dual problem. We note that the linearisations that lead to $\mathcal{M}(\mathbf{u}, \mathbf{u}_h; \cdot, \cdot)$ and $\bar{J}(\mathbf{u}, \mathbf{u}_h; \cdot)$ are performed about $\mathbf{u}_h$, and result in $\mathcal{N}'[\mathbf{u}_h](\cdot, \cdot)$ and $J'[\mathbf{u}_h](\cdot)$, respectively. Therefore, the linearised dual problem is defined as: find $\hat{\mathbf{z}} \in \mathbf{V}$ such that

$$\mathcal{N}'[\mathbf{u}_h](\mathbf{w}, \hat{\mathbf{z}}) = J'[\mathbf{u}_h](\mathbf{w}) \ \forall \mathbf{w} \in \mathbf{V}. \tag{6.2.1}$$

We then construct a discretisation using discontinuous Galerkin finite elements, approximating the dual problem as: find $\hat{\mathbf{z}}_h \in \hat{\mathbf{V}}_h$ such that

$$\mathcal{N}'[\mathbf{u}_h](\mathbf{w}_h, \hat{\mathbf{z}}_h) = J'[\mathbf{u}_h](\mathbf{w}_h) \ \forall \mathbf{w}_h \in \hat{\mathbf{V}}_h. \tag{6.2.2}$$

We remark that there are several approaches to producing numerical approximations to the solution of the dual problem $\hat{\mathbf{z}}_h$. It is important to note here that, $\hat{\mathbf{z}}_h$ cannot be computed on the same finite element space as was employed for the primal problem, due to the definition of $\mathbf{u}_h$. Since any resulting error representation formula will be the same, producing an error estimate that is zero. In particular, $\mathcal{N}(\mathbf{u}_h, \hat{\mathbf{z}}_h) = 0$ in this case. Therefore, we now discuss the three main approaches [87] to approximating $\hat{\mathbf{z}}_h$. The first method is to fix the polynomial degree m such that it is constant in both the calculation of $\mathbf{u}_h$ and $\hat{\mathbf{z}}_h$, but evaluate $\hat{\mathbf{z}}_h$ on a sequence of dual finite element meshes, which, in general, differ from the one used to calculate the primal solution. The second technique is to compute $\hat{\mathbf{z}}_h$

using piecewise polynomials of higher degree $\hat{m} > m$ than those used in the numerical approximation of the primal solution $\mathbf{u}_h$. Here, the underlying mesh $\mathcal{T}_h$ is the same for both calculations. Finally, a variant of the second method exists, in which the approximate dual solution is calculated on the same underlying mesh $\mathcal{T}_h$, with the same polynomial degree m; however, patchwise recovery techniques are employed to improve the accuracy of $\hat{\mathbf{z}}_h$ [21, 22, 31].

In this work we focus on the second technique, because it generally leads to highly efficient error estimation without excessive computational overhead [87]. In particular, if the mesh refinement parameters are chosen such that the number of degrees of freedom employed in the dual finite element space is approximately the same as the number of degrees of freedom in the new primal finite element space after adaptive mesh refinement has taken place, then the computational cost of calculating the dual solution is nearly equivalent to the cost of a single Newton step in the computation of $\mathbf{u}_h$ on the newly generated adaptive mesh. Specifically, the calculation of the dual solution is a linear problem. Let

$$\begin{aligned}
\hat{\mathbf{V}}_h &= \left\{ \mathbf{v} \in \left[L^2(\Omega) \right]^{d+2} : \mathbf{v}\,|_\kappa \in \left[\mathcal{Q}_{\hat{m}}(\kappa) \right]^{d+2} \ \forall \kappa \in \mathcal{T}_h \right\}, \\
\hat{Q}_h &= \left\{ q \in L^2(\Omega) : q\,|_\kappa \in \mathcal{Q}_{\hat{m}-1}(\kappa) \ \forall \kappa \in \mathcal{T}_h \right\},
\end{aligned}$$

with $\hat{m} > m$. We recall that we seek solutions for the d velocity variables, as well as the two turbulent variables, k and $\tilde{\omega}$. A question to ask is whether the error introduced by the linearisation about the discrete solution $\mathbf{u}_h$, and the approximation $\hat{\mathbf{z}}_h$ of the dual solution $\mathbf{z}$, severely affects the efficiency of the DWR method. As such, we proceed as in [80], introducing three terms

$$\begin{aligned}
J(\mathbf{u}) - J(\mathbf{u}_h) &= -\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \mathbf{z}_h) \\
&= -\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \hat{\mathbf{z}}) - \mathcal{N}(\mathbf{u}_h, \hat{\mathbf{z}} - \hat{\mathbf{z}}_h) - \mathcal{N}(\mathbf{u}_h, \hat{\mathbf{z}}_h - \mathbf{z}_h),
\end{aligned} \tag{6.2.3}$$

noting that the above are indeed strict equalities. The first term, $-\mathcal{N}(\mathbf{u}_h, \mathbf{z} - \hat{\mathbf{z}})$, represents the error incurred through the linearisation of the dual problem, and is expected to be small in cases where the analytical solution $\mathbf{u}$ is smooth [83]. The second term, $-\mathcal{N}(\mathbf{u}_h, \hat{\mathbf{z}} - \hat{\mathbf{z}}_h)$, represents the error due to the numerical approximation of the linearised dual solution. This error term is of a higher-order than the approximate error representation if the dual solution is sufficiently regular and is approximated by higher order polynomials than those used in the primal problem. The final term is the approximate error representation formula that is actually computed in practice. We remark, that since we are considering incompressible flows, the analytical solution $\mathbf{u}$ is expected to be formally smooth everywhere (although in practice one may witness steep gradients). However, when considering compressible flows, in particular those containing shocks, where the analytical

solution $\mathbf{u}$ is not smooth, the dual solution often is, allowing the error to be controlled remarkably well [80].

Thus, replacing the dual solution $\mathbf{z}$ in (6.1.5) with our approximation $\hat{\mathbf{z}}_h$, we have the following approximate error representation formula

$$J(\mathbf{u}) - J(\mathbf{u}_h) \approx -\mathcal{N}(\mathbf{u}_h, \hat{\mathbf{z}}_h - \mathbf{z}_h) \equiv \sum_{\kappa \in \mathcal{T}_h} \hat{\eta}_\kappa. \tag{6.2.4}$$

This in turn leads to an analogous formula for the approximate error bound:

$$|J(\mathbf{u}) - J(\mathbf{u}_h)| \lesssim \sum_{\kappa \in \mathcal{T}_h} |\hat{\eta}_\kappa|. \tag{6.2.5}$$

Finally, we define the absolute error with respect to the target functional as $|J(\mathbf{u}) - J(\mathbf{u}_h)|$, and the relative error as $\frac{|J(\mathbf{u}) - J(\mathbf{u}_h)|}{|J(\mathbf{u})|}$.

6.3 The Adaptive Refinement Strategy

We now discuss the development of an adaptive refinement and coarsening algorithm that is able to efficiently control the error of a target functional $J(\cdot)$. As such, we employ the approximate error bound defined in (6.2.5), to determine (approximately) whether the desired degree of accuracy has been reached. In particular, we introduce a tolerance TOL, with the aim of the computation being the reduction of the error, so that $|J(\mathbf{u}) - J(\mathbf{u}_h)| \leq$ TOL. Therefore, we introduce the algorithm's stopping criterion of

$$\sum_{\kappa \in \mathcal{T}_h} |\hat{\eta}_\kappa| \leq \text{TOL}. \tag{6.3.1}$$

If the stopping criterion (6.3.1) is not met for a particular finite element mesh $\mathcal{T}_h$, then the element-wise error indicators $\hat{\eta}_\kappa$ are used to drive mesh refinement and coarsening.

The complete algorithm follows the cycle described below:

1. Construct the initial mesh $\mathcal{T}_h$.
2. Compute the solution $\mathbf{u}_h \in \mathbf{V}_{h,m}$ on the current mesh.
3. Compute the approximation $\hat{\mathbf{z}}_h \in \hat{\mathbf{V}}_h$ on the current mesh, where $\hat{\mathbf{V}}_{h,\hat{m}}$ is a finite element space defined analogous to $\mathbf{V}_{h,m}$, except consisting of piecewise (discontinuous) polynomials of degree $\hat{m} > m$.

4. Evaluate the error bound $\sum_{\kappa \in \mathcal{T}_h} |\hat{\eta}_\kappa|$.

5. If $\sum_{\kappa \in \mathcal{T}_h} |\hat{\eta}_\kappa| \leq \text{TOL}$ then stop. Else, refine and coarsen a fixed fraction of the total number of elements of the current mesh according to the elementwise error indicators $|\hat{\eta}_\kappa|$, hence constructing a new mesh. Go to 2.

The algorithm, therefore, requires three user defined parameters: TOL, the percentage of elements to be refined, $U\%$, and the percentage of elements to be coarsened, $L\%$. This known commonly as fixed fraction refinement, and we limit the number of hanging nodes per element face to one. Other strategies could be employed in place of this, such as refining the elements which contribute to a user-defined percentage of the total error.

6.4 Isotropic h-Refinement

Various isotropic subdivisions of elements are by far the most widely used mesh modifications. They involve the subdivision of a mesh element κ into smaller, similar sized, elements aiming to retain shape regularity. The popularity of this strategy is due to its ease of use, since the maximum angle condition for the standard FEM is never violated, and, in the case of triangles, the removal of hanging nodes is also possible by splitting the neighbouring element. We refer to Figures 6.4.1, 6.4.2 and 6.4.3 for illustration.

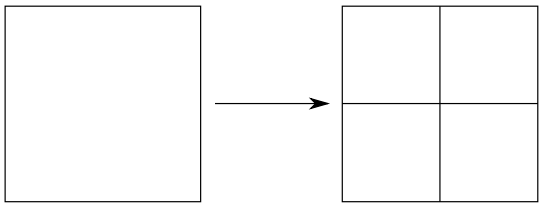

Figure 6.4.1: Isotropic refinement of a quadrilateral element.

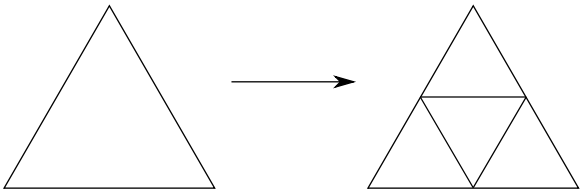

Figure 6.4.2: Isotropic refinement of a triangular element.

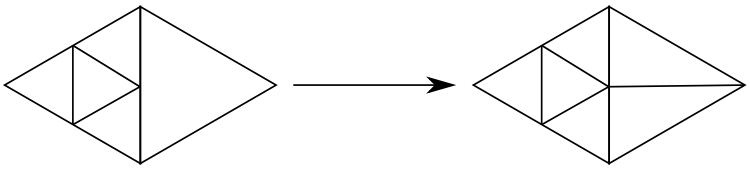

Figure 6.4.3: Removal of a hanging node in a triangular mesh.

We now introduce some mesh smoothing techniques. These are applied after the creation of each new finite element mesh, and serve the purpose of producing a more monotonic error convergence, especially in the functional estimation setting. The first is the removal of refined islands in the mesh, and the second is the refinement of unrefined islands. These techniques are detailed below in Figures 6.4.4 and 6.4.5, respectively.

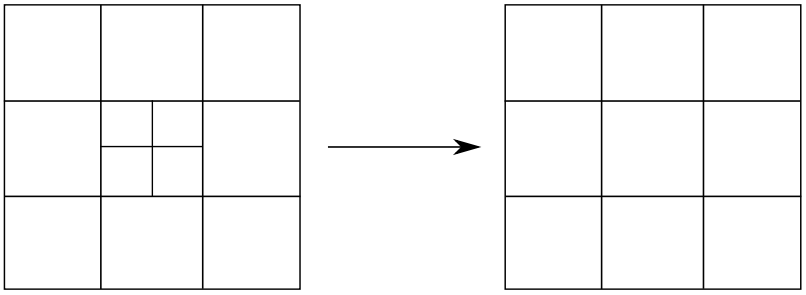

Figure 6.4.4: Removal of a refinement island.

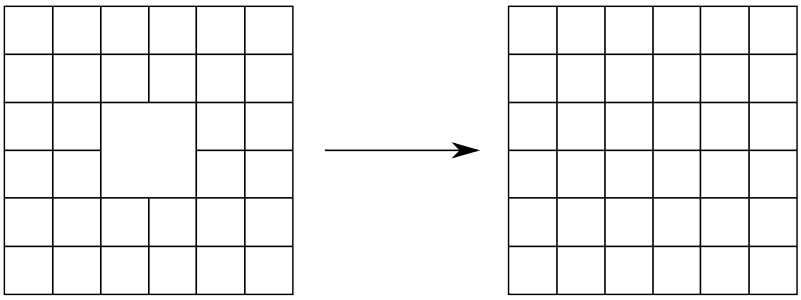

Figure 6.4.5: Refinement of an unrefined island.

However, it should be noted that even though we are considering an isotropic approach to h-refinement, we do not limit our refinement to only isotropic type elements. In particular, the majority of elements in Figure 5.4.3 are quite anisotropic, yet are still considered for refinement in the isotropic approach proposed above. We make the following distinction, with isotropic refinement splitting an element into similar shaped sub-elements of equal

size. Whereas, anisotropic refinement is the division of an element with the aim of creating anisotropic sub-elements, i.e. Figure 6.4.6 and 6.4.7.

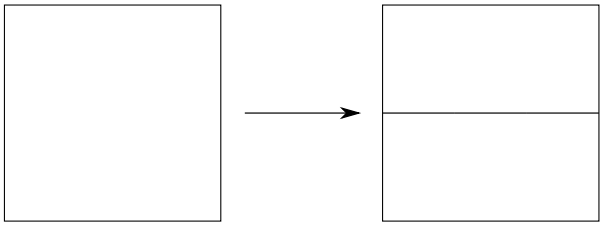

Figure 6.4.6: Anisotropic refinement of a quadrilateral element. (Type 1).

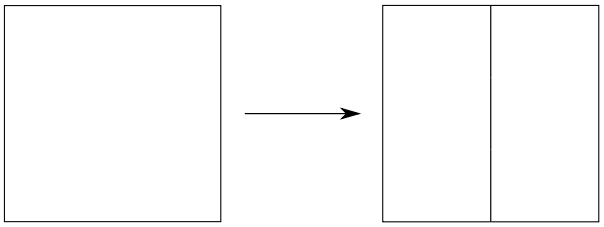

Figure 6.4.7: Anisotropic refinement of a quadrilateral element. (Type 2).

This of particular importance with respect to the polytopic meshes seen in Chapter 7, as the choice of penalty parameter (7.3.2) also allows for anisotropic elements. The refinement of polytopic elements is carried out by calculating the element centroid, then the element is split along the coordinate axes. In the case where the polygonal mesh is constructed via mesh agglomeration, the point chosen is a node close to the centroid in the underlying mesh. This is shown in Figure 6.4.7.

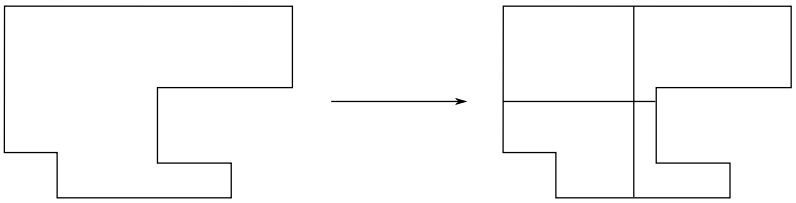

Figure 6.4.8: Refinement of general polygonal elements.

6.5 Target Functionals

In general, the choice of target functional is dependent upon the physical properties of the underlying system or the particular concerns of the user. For instance, if we are considering a domain containing an aircraft wing, we may be interested in the lift coefficients, or indeed the drag at a certain point on the wing surface. When considering turbulent incompressible flows, we are conscious of the rapid change in pressure and ω close to boundary walls. As such, a target functional could be chosen such that the adaptive algorithm prioritises mesh refinement close to boundary walls. For this study, we consider the profile drag as a target functional. We refer to [114] and the references cited therein for a discussion of the suitability of various in-flight profile-drag measurements. In particular, the deficit in total pressure is related to the profile drag coefficient c_d by the following expression

$$c_d = \frac{1}{cq_0} \int_{wake} (p_{t_0} - p_t) \ \mathrm{ds}, \tag{6.5.1}$$

where c is the aerofoil chord length, q_0 is the initial dynamic pressure, p_{t_0} is the total pressure up stream of the aerofoil and p_t is the total pressure measured down stream of the aerofoil. The wake integral refers to an integral calculated on a cross section of the domain behind the aerofoil, and is only valid when the wake pressure deficit is measured sufficiently far from the trailing edge. The laws of continuity in the free-stream flow mean that measurements of static and total pressures may be taken anywhere in the free-stream. As such, in the following numerical experiments, we use the domain inlet conditions to measure the initial pressures p_{t_0}, and calculate the wake pressure deficit p_t on the domain outlet.

6.6 Numerical Experiments

In this section, we conduct numerical experiments to analyse the effectiveness of the choice of target functional $J(\mathbf{u})$, as well as the effectiveness of employing a DWR approach for solution refinement. All numerical calculations are carried out using AptoFEM, in conjunction with both MUMPS and ParMETIS. These experiments also incorporate the proposed penalty parameter detailed in Chapter 5, enabling us to better handle the curved boundaries of the various domains.

6.6.1 NACA 0012 Aerofoil

The NACA aerofoils are a series of aircraft wings developed by the National Advisory Committee of Aeronautics. The numerical classification of each wing describes the parameters of the equation used to generate the aerofoil shape. The 0012 denotes a symmetric wing with a maximum thickness of 12% of the chord length. Whilst having very limited real world uses, this aerofoil is a common test case for fluid simulations, and, as such, is of interest to us. The formula describing the shape of a general NACA $00't'$ aerofoil is given by

$$y_t = 5t\left(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4\right), \tag{6.6.1}$$

x denotes the position along the aerofoil between 0 and 1, whilst y_t is the distance of the wing surface from the central axis. We note here, that for the purposes of this numerical simulation we have made the usual correction to (6.6.1), switching the final term for $-0.1036x^4$ to account for the lack of a trailing edge design. This modification results in the smallest change to the overall shape of the aerofoil, whilst still achieving the desired result of a zero thickness trailing edge. Consider an incompressible, turbulent, free-stream flow around a NACA 0012 aerofoil. Figure 6.6.1 shows the usual schematic of the aerofoil, with a chord length $c = 1$. We consider dimensionless quantities for this numerical experiment, analysing the performance of the DGFEM solver in correctly representing the turbulent boundary layer of the aerofoil.

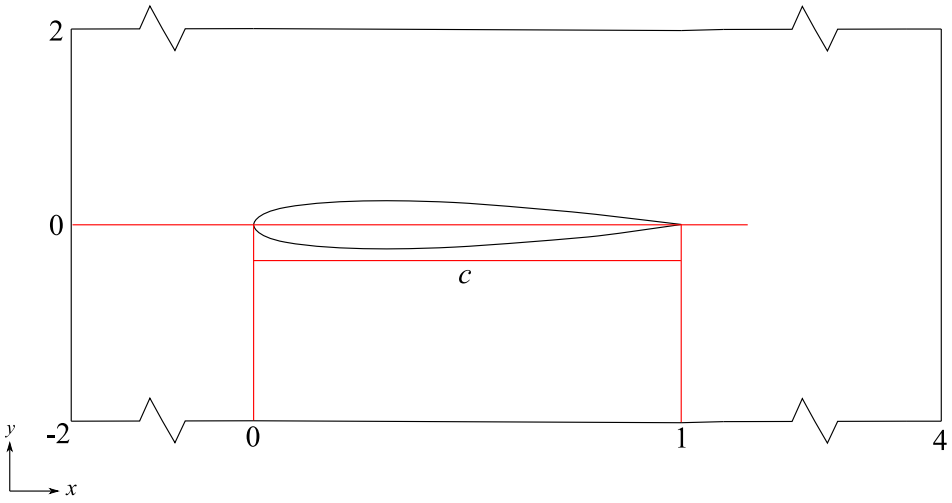

Figure 6.6.1: Schematic of the NACA 0012 aerofoil. Chord length $c = 1$, angle-of-attack $\alpha = 0°$.

The simulation domain $\Omega \subset \mathbb{R}^2$ is defined as the rectangle $[-2.0\ 2.0] \times [-2.0\ 4.0]$, with the aerofoil located along the central axis between $x = 0$. The angle-of-attack α° (the angle between the aerofoil chord and the horizontal x-axis) is adjusted by way of the inlet conditions, changing the direction of the inlet flow accordingly. An initial validation test case is carried out using the data collected in [98], comparing the time-averaged velocity measurements of the boundary layer on the suction side of the aerofoil for differing angles-of-attack, $\alpha = 3^\circ$ and $\alpha = 6^\circ$. The Reynolds number for the simulation, based on the aerofoil chord length, is $Re = 23,000$. Following [98], the velocity is measured at pre-defined points along the aerofoil, shown as filled circles in Figure 6.6.2, then normalised using the free stream velocity to produce the mean velocity.

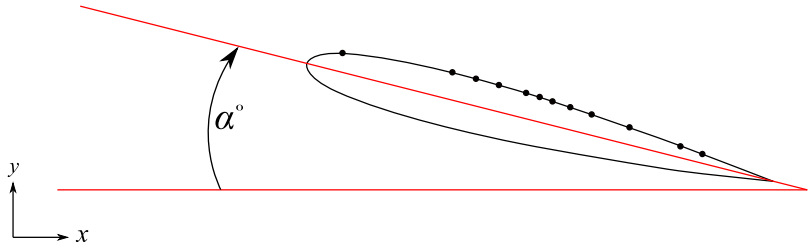

Figure 6.6.2: Schematic of the NACA 0012 aerofoil, showing measurement points with filled circles.

For this validation test case, we consider a triangulation $\mathcal{T}_h^{\text{initial}}$ consisting of $11,000$ quadrilateral elements shown in Figure 6.6.3. We define the inlet boundary located at $x = -2.0$, to be a turbulent free stream with average velocity $u_\infty = 13.909$. The direction of the inlet velocity is adjusted to represent the differing angles of attack. The aerofoil is faceted with curved elements as detailed in Chapter 5, with a no-slip boundary condition. Periodic boundaries are used for the domain outflows at $y = -2.0$ and $y = 2.0$, and the main outflow at $x = 4.0$, uses the stress-free Neumann condition.

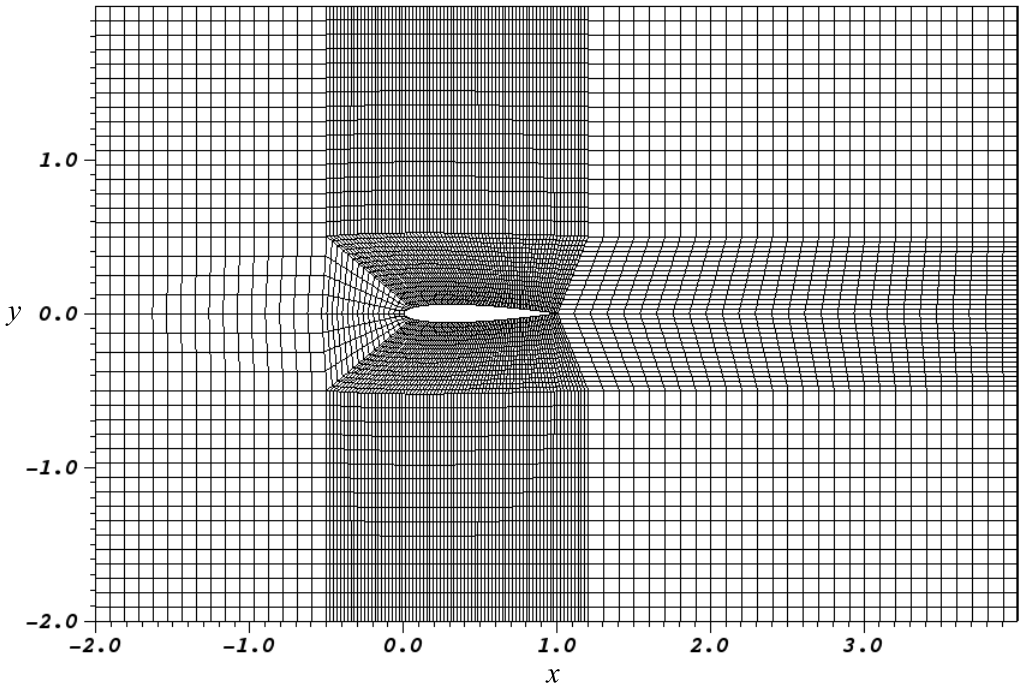

Figure 6.6.3: Two-dimensional mesh of the NACA 0012 aerofoil $\mathcal{T}_h^{\text{initial}}$, consisting of $11,000$ quadrilateral elements.

In order to achieve the target Reynolds number $Re = 23,000$, the continuation algorithm outlined in in Section 4.4.3 is used, along with the curvilinear penalty parameter (5.3.4). We consider the performance of the DGFEM method on the initial mesh, before applying the DWR refinement approach, with the aim of improving the accuracy of the boundary layer representation. We note that the initial mesh $\mathcal{T}_h^{\text{initial}}$, has a rather coarse mesh resolution in the direction perpendicular to the wall, as shown in Figure 6.6.3. This significantly reduces the number of solution data points within the boundary layer measurement, prompting the need for solution refinement.

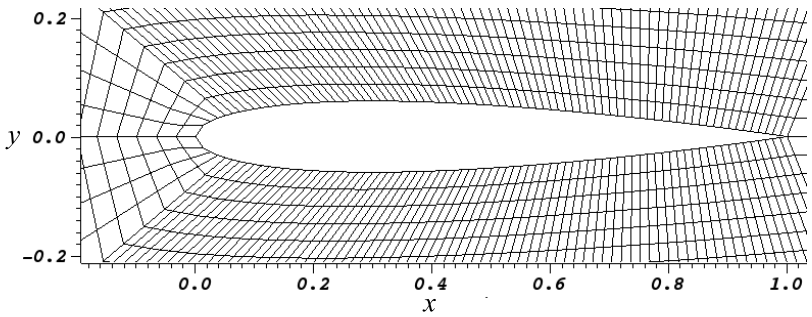

Figure 6.6.4: Near-wall close-up of NACA 0012 aerofoil mesh $\mathcal{T}_h^{\text{initial}}$.

Figures 6.6.5 and 6.6.6 show the boundary layer mean velocities at different distances $\frac{S}{c}$, from the aerofoil's leading edge. In particular, $\frac{S}{c}$ is the normalised distance along the aerofoil surface from the leading edge to the boundary layer measurement stations, while $\frac{h}{c}$ is the normalised perpendicular distance of the measurement station from the aerofoil surface. Figure 6.6.5 has an angle-of-attack $\alpha = 3°$, while Figure 6.6.6 has an angle-of-attack $\alpha = 6°$.

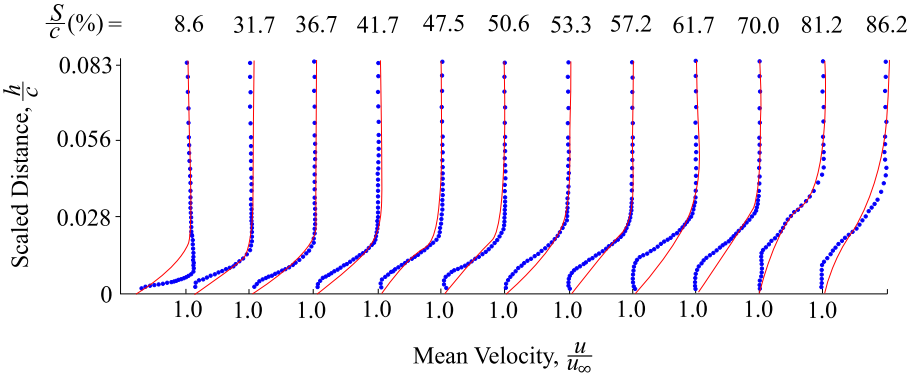

Figure 6.6.5: Mean boundary layer velocities calculated on the mesh, $\mathcal{T}_h^{\text{initial}}$. Angle-of-attack $\alpha = 3°$. The DGFEM solution is shown with a solid red line, and the experimental data points [98] are represented with blue circles.

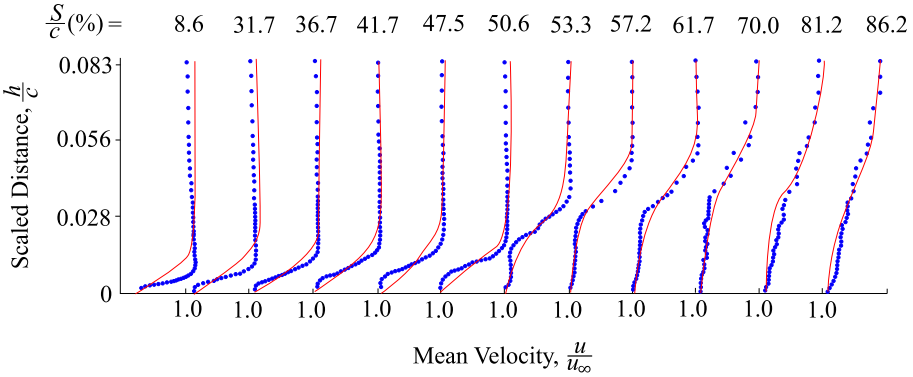

Figure 6.6.6: Mean boundary layer velocities calculated on the mesh, $\mathcal{T}_h^{\text{initial}}$. Angle-of-attack $\alpha = 6°$. The DGFEM solution is shown with a solid red line, and the experimental data points [98] are represented with blue circles.

As expected, in both Figures 6.6.5 and 6.6.6, the lack of mesh resolution severely affects the accuracy of the DGFEM solution close to the aerofoil. The polynomial representation of the DGFEM solution, quadratic polynomials in the case of the velocities, allows for a reasonable approximation of the near-wall behaviour. However, the relatively large distance

between the first mesh node and the aerofoil surface, clearly affects the accuracy of the approximation close to the leading edge, with the DGFEM solution under approximating the near-wall velocities for both angles-of-attack. Specifically, the DGFEM solver creates a quadratic representation of the solution, joining the zero velocity at the aerofoil surface with a positive velocity at the first mesh node. However, it appears that the velocity profile between the two nodes is too complex to be represented by a single quadratic, and, as such, we consider mesh refinement to improve the accuracy of the DGFEM solution.

We apply 5 passes of the DWR refinement algorithm, refining the 20% of elements with the largest approximate absolute error value according to (6.5.1), whilst coarsening the 10% of elements with the lowest approximate absolute error value. The value of (6.5.1) is calculated on the domain outlet for ease of implementation, although in reality, any point sufficiently far downstream from the aerofoil would be suitable. Figures 6.6.7 and 6.6.8 show the refined meshes for $\alpha = 3^\circ$ and $\alpha = 6^\circ$, respectively.

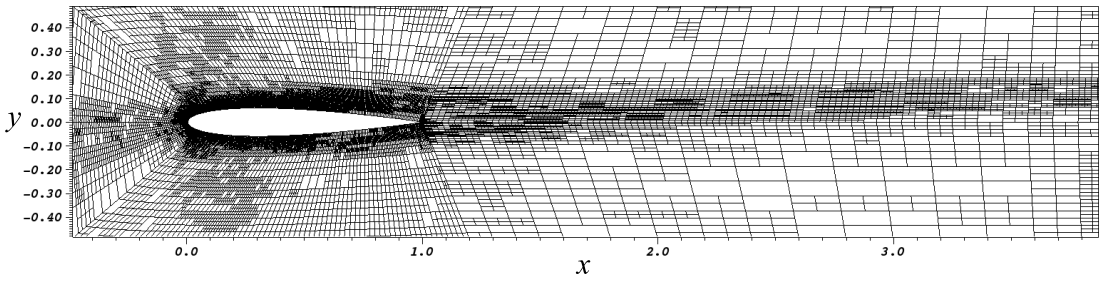

Figure 6.6.7: Two-dimensional refined mesh $\mathcal{T}_h^{\text{refined},3}$ after 5 h-refinement passes. Angle-of-attack $\alpha = 3^\circ$.

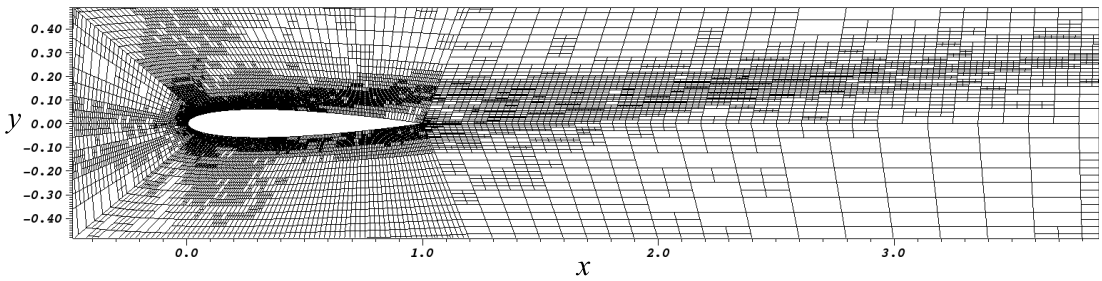

Figure 6.6.8: Two-dimensional refined mesh $\mathcal{T}_h^{\text{refined},6}$ after 5 h-refinement passes. Angle-of-attack $\alpha = 6^\circ$.

The refined mean velocity profiles shown in Figures 6.6.9 and 6.6.10, are extracted from the solutions calculated on the refined meshes shown in Figures 6.6.7 and 6.6.8.

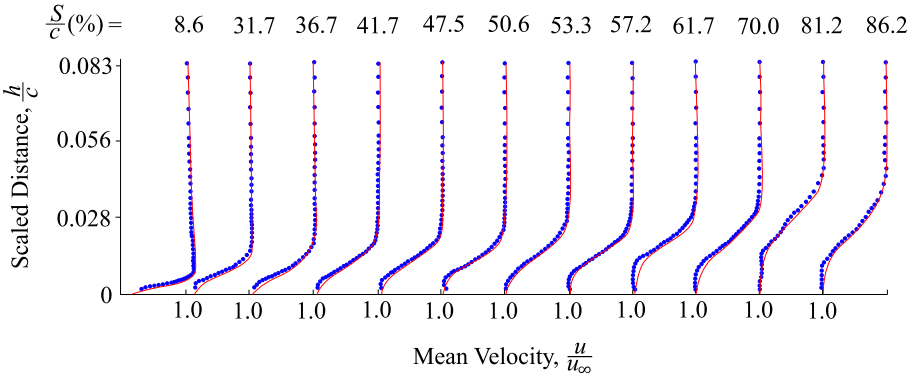

Figure 6.6.9: Mean boundary layer velocities calculated on the mesh, $\mathcal{T}_h^{\text{refined},3}$, following 5 h-refinement passes. Angle-of-attack $\alpha = 3°$. The DGFEM solution is shown with a solid red line, and the experimental data points [98] are represented with blue circles.

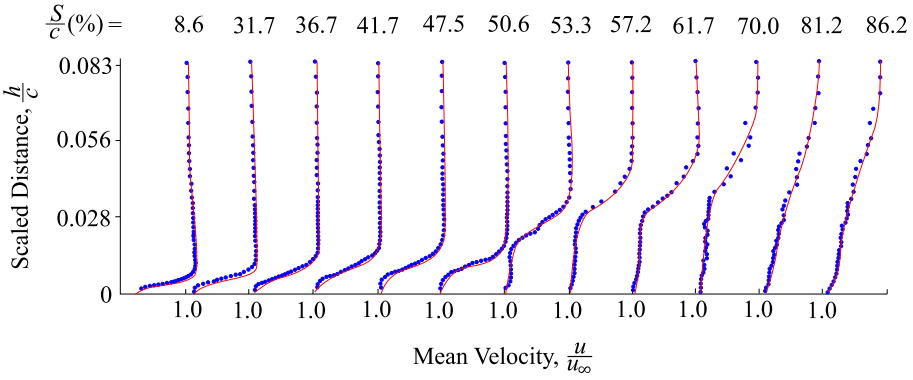

Figure 6.6.10: Mean boundary layer velocities calculated on the mesh, $\mathcal{T}_h^{\text{refined},6}$, following 5 h-refinement passes. Angle-of-attack $\alpha = 6°$. The DGFEM solution is shown with a solid red line, and the experimental data points [98] are represented with blue circles.

Figures 6.6.9 and 6.6.10 show a significant improvement in the accuracy of the mean velocity profiles across the boundary layer. This is due to the increased number of mesh nodes within the boundary layer region, as shown in Figures 6.6.7 and 6.6.8, compared to the initial mesh $\mathcal{T}_h^{\text{initial}}$ shown in Figure 6.6.4. Close to the aerofoil leading edge, the representation of the velocity profile is markedly more accurate, capturing the 'cubic-like' curve that develops in the profile as we move towards the trailing edge of the wing. However, even after refinement, the DGFEM is still under representing the boundary layer, over-estimating the velocity of the flow close to the aerofoil leading edge. This issue is apparent at both angles-of-attack, see Figures 6.6.9 and 6.6.10, suggesting that the issue is associated with the choice of turbulence model or numerical flux. Also, the proposed DGFEM uses a RANS implementation, which may smooth the velocity transition from

boundary layer to free stream conditions, contributing to the reduced size of the boundary layer in the numerical simulation.

A clear advantage of the DGFEM is the ability to use the same symmetric initial mesh $\mathcal{T}_h^{\text{initial}}$, for simulations with different boundary conditions/angles-of-attack. The flexibility of the DWR approach, along with the proposed refinement algorithm, allows us to create automatically customised meshes that increase the accuracy of the DGFEM solution, without resorting to the use of mesh creation programs. This feature is particularly important from an industrially perspective, allowing multiple simulations of blades and aerofoils at different angles-of-attack to be carried out, without additional input from engineers, further automating the blade design process.

We now wish to explore the performance of the DWR algorithm with respect to the target functional (6.5.1). As such, we consider a neutral angle-of-attack $\alpha = 0^\circ$, a turbulent free stream inlet with an average velocity $u_\infty = 13.909$, and an improved continuation algorithm.

In this numerical experiment, we incorporate the continuation algorithm proposed in Chapter 4, with the refinement algorithm in Section 6.3. We balance the increase in Reynolds number with the reduction in absolute error according to the target functional (6.5.1). To do this, we calculate the low Reynolds number laminar solution to begin the continuation process, as detailed in Section 4.4.3. However, once the Reynolds number has increased to $Re = 1,200$, we apply an isotropic h-refinement of the mesh, refining 20% of the elements with the highest approximate absolute error value according to (6.5.1), whilst coarsening the 10% of elements with the lowest approximate absolute error value. Once again, the value of (6.5.1) is calculated on the domain outlet for ease of implementation. The continuation algorithm continues, increasing the Reynolds number by 10% between successive solves, until the target Reynolds number $Re = 23,000$ is reached, with two further mesh refinement passes being made at $Re = 10,000$ and $Re = 20,000$. Following the findings of these numerical experiments, a fully detailed explanation of the DWR continuation algorithm can be found in Section 6.7, as well as, in its final form, in Section 8.1.

We begin with the initial mesh $\mathcal{T}_h^{\text{initial}}$ shown in Figures 6.6.1, 6.6.3 and 6.6.4, consisting of $11,000$ quadrilateral elements. Via the above, proposed continuation method, we produce the final continuation mesh $\mathcal{T}_h^{\text{refined},0}$, consisting of $23,500$ elements shown in Figure 6.6.11.

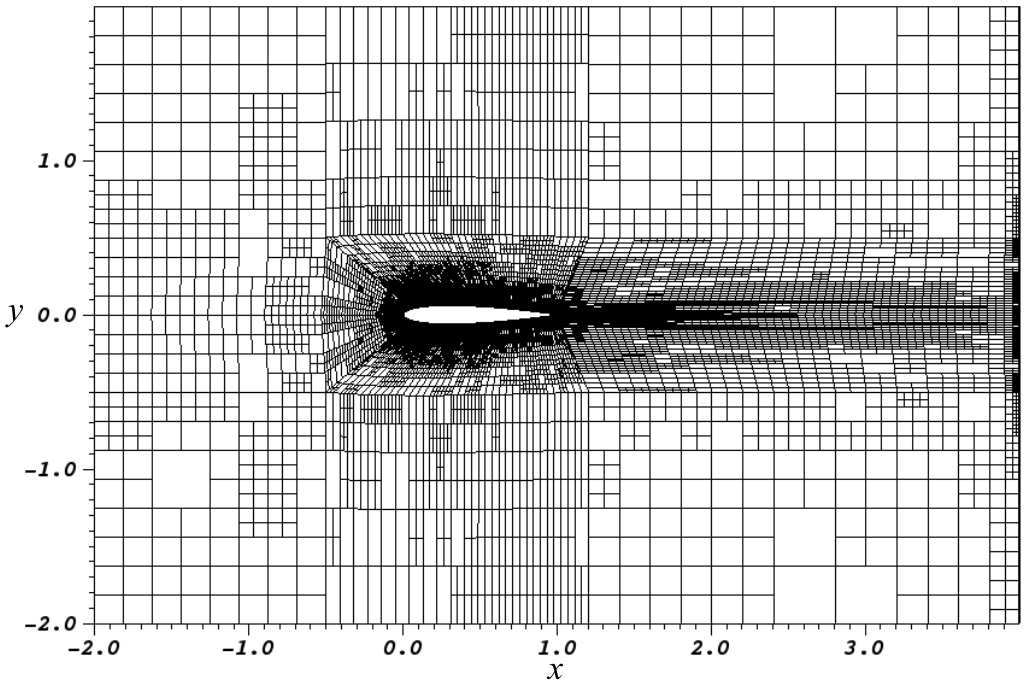

Figure 6.6.11: Final continuation mesh $\mathcal{T}_h^{\text{refined},0}$, consisting of $23,500$ elements.

As mentioned above, the Reynolds number is increased by 10% between successive nonlinear solves. However, if the DGFEM fails to converge for a particular Reynolds number, the percentage by which we increase the value is halved until the DGFEM converges; after which, the value is set back to 10% for the next solve. Incorporating mesh refinement/coarsening into the continuation process, we achieved a Reynolds number of $23,000$ in 40 nonlinear solves, compared to the 81 solves required when using continuation alone, as in Section 4.4.3. The final continuation mesh $\mathcal{T}_h^{\text{refined},0}$ does have some unexpected refinement around the domain outflow boundary. This is due to the position we chose to sample the target functional and not representative of any underlying flow features.

As we are considering an angle-of-attack $\alpha = 0^\circ$, we present solutions of the scaled axial velocity u_1 in Figure 6.6.12, and the scaled static pressure p in Figure 6.6.13. These solutions provide the most useful information, with the axial velocity in Figure 6.6.12 showing the turbulent boundary layer along the aerofoil trailing edge. Figure 6.6.13 demonstrates one of the major challenges of the DGFEM in this numerical test case: resolving the pressure at the tip of the aerofoil leading edge.

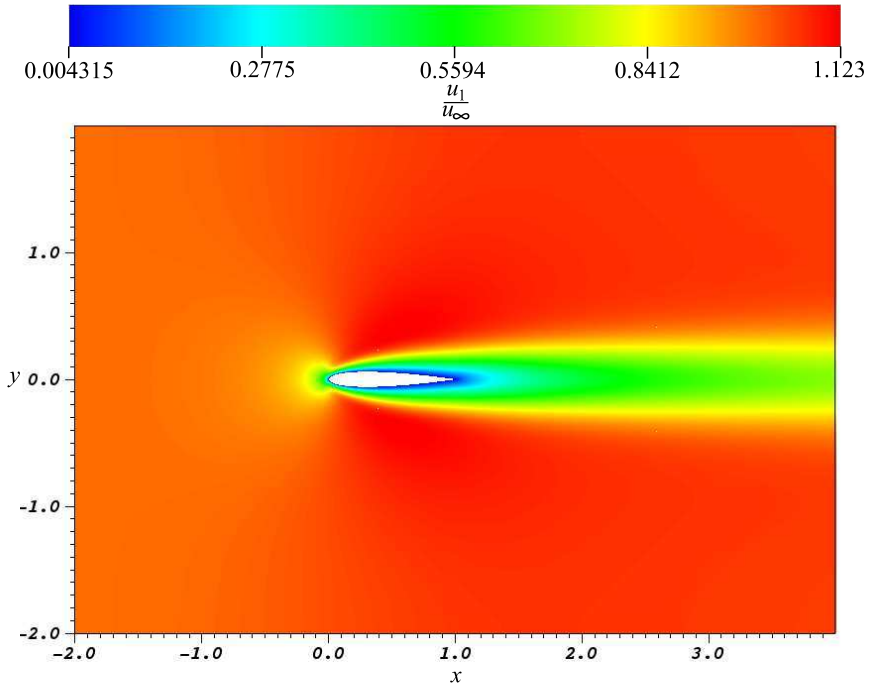

Figure 6.6.12: Colour plot of the scaled axial velocity $\frac{u_1}{u_\infty}$ on the final continuation mesh $\mathcal{T}_h^{\text{refined},0}$. $Re = 23,000$.

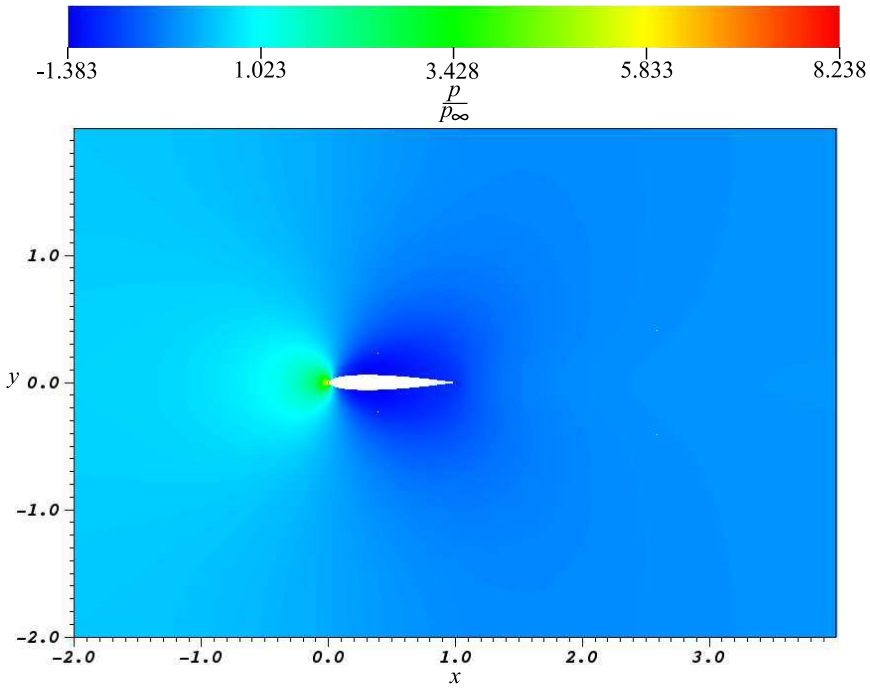

Figure 6.6.13: Colour plot of scaled static pressure $\frac{p}{p_\infty}$ on the final continuation mesh $\mathcal{T}_h^{\text{refined},0}$. $Re = 23,000$.

Figure 6.6.13 shows, as expected, a large spike in the pressure at the leading edge of the aerofoil. The numerical solution is sufficiently accurate for engineering applications, with comparable mesh density to the results shown in Figure 6.6.9 and 6.6.10. However, the sudden increase in pressure at the wing tip, suggests that further mesh refinement would improve the solution resolution close to the aerofoil. Therefore, we refine the final result reducing the error further for the target Reynolds number of $23,000$. For industrial applications we would repeat this process until a user-specified tolerance is met. However, in this purely academic setting, we wish to analyse the effectiveness of (6.5.1) as a target functional. As before, after each successful solve we coarsen the 10% of elements with the smallest approximate absolute error value, and refine isotropically the 20% of elements with the largest approximate absolute error value.

We note that it is possible in this test case to consider the value of the relative error with respect to the target functional, but this value is not readily available and first requires evaluating a solution to the problem on a very fine mesh. In general, for large-scale industrial problems, the number of degrees of freedom is often very large, and as such, it is not always possible to consider a finer mesh due to restrictions placed on the number of computational resources. However, in this academic setting we are able to first compute a solution to the problem on a fine uniform mesh consisting of $176,000$ quadrilateral elements, and then calculate an approximation $J(\mathbf{u}_{h,\text{fine}})$ to the target functional $J(\mathbf{u})$. We compute $J(\mathbf{u}_{h,\text{fine}}) = 1553.564681$ with a Reynolds number of $23,000$. This value is representative of the order of magnitude of $J(\mathbf{u})$, and serves to validate the refinement technique we have proposed by approximating the relative error. Additionally, we consider the total error with respect to the target functional, $|J(\mathbf{u}) - J(\mathbf{u}_h)| \approx \left|\sum_{\kappa\in\mathcal{T}_h} \hat{\eta}_\kappa\right|$, as well as the error bound, $\sum_{\kappa\in\mathcal{T}_h} |\hat{\eta}_\kappa|$.

Number of Degrees of Freedom	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$	$\frac{\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert}{\lvert J(\mathbf{u}_{h,\text{fine}})\rvert}$	$\left\lvert\sum_{\kappa\in\mathcal{T}_h} \hat{\eta}_\kappa\right\rvert$
$958,220$	2.516973	1.620128×10^{-3}	7.764768×10^{-2}
$1,385,140$	0.554820	3.571271×10^{-4}	4.393811×10^{-3}
$1,795,780$	0.218668	1.407524×10^{-4}	1.529754×10^{-3}
$2,338,800$	0.195745	1.240663×10^{-4}	9.151673×10^{-4}

Table 6.1: The approximate absolute and relative error values with respect to the target functional (6.5.1).

In the literature, DWR *a posteriori* error estimation has been studied for compressible flows [80, 83, 84], as well as for laminar incompressible flows [36, 46] with high levels of success. As such, having applied these ideas to turbulent incompressible flows, we expect similar

error convergence rates. In general, as the number of degrees of freedom is increased, we observe a reduction in the error of the target functional, as shown in Table 6.1. The mesh density in Figures 6.6.14 and 6.6.15, demonstrates the efficiency of the proposed approach to solution refinement, with targeted refinement along the blade edges. This is reflected in the numerical solutions of Figures 6.6.16, 6.6.17, 6.6.18 and 6.6.19, with changes to solution values occurring close to the aerofoil boundary. In particular, the value of $\tilde{\omega}$ requires a far greater mesh density in order to be accurately represented, due to the rate at which the solution value increases as it approaches the aerofoil boundary. We also note the behaviour of the static pressure solution in Figure 6.6.19, requiring additional packing of elements around the aerofoil leading edge to accommodate the spike in pressure, see Figure 6.6.15.

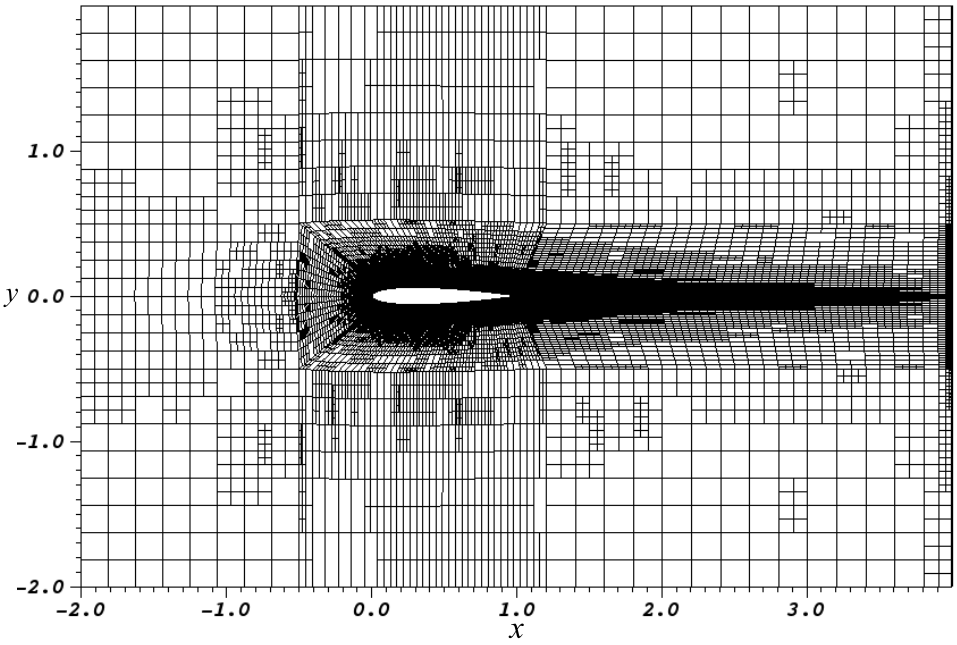

Figure 6.6.14: Final refinement mesh $\mathcal{T}_h^{\text{final}}$, $2,338,800$ degrees of freedom.

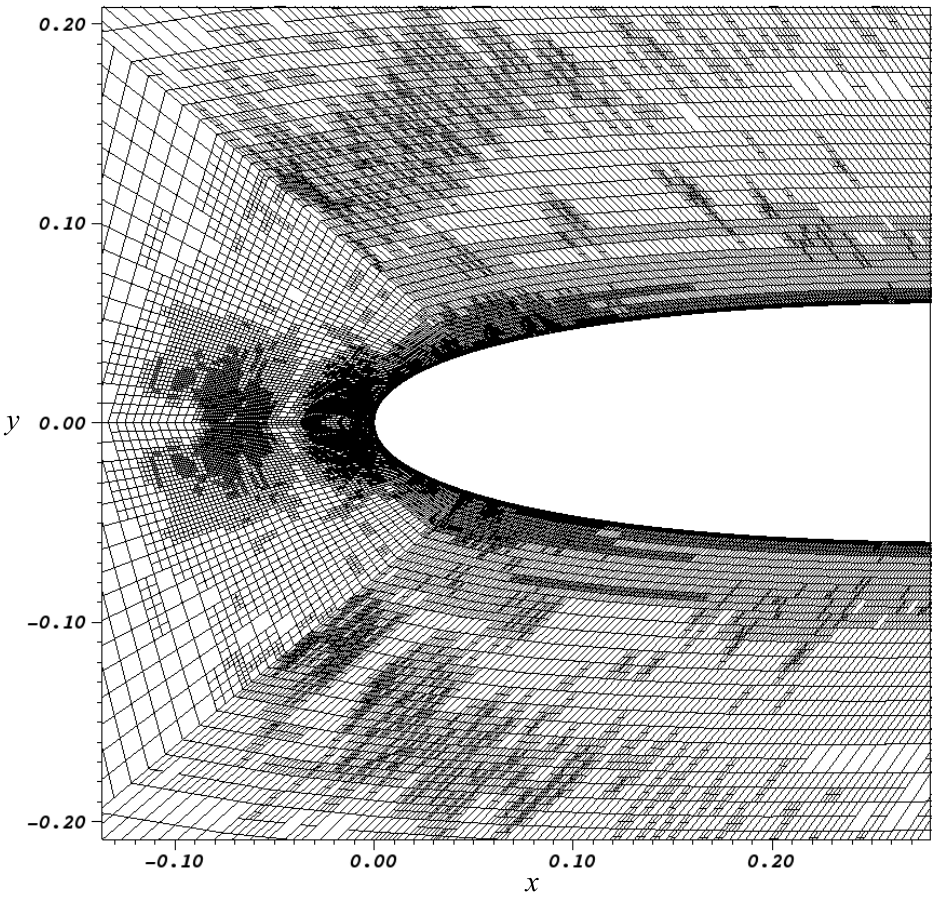

Figure 6.6.15: Close-up of the blade leading edge of $\mathcal{T}_h^{\text{final}}$.

We present the refined solutions computed on the mesh $\mathcal{T}_h^{\text{final}}$ shown in Figures 6.6.14 and 6.6.15 for a Reynolds number of $23,000$. Figure 6.6.16, 6.6.17, 6.6.18 and 6.6.19 are colour plots of the scaled axial velocity $\frac{u_1}{u_\infty}$, the turbulent kinetic energy k, the logarithm of the dissipation per unit turbulence kinetic energy $\tilde{\omega}$, and the scaled static pressure $\frac{p}{p_\infty}$, respectively.

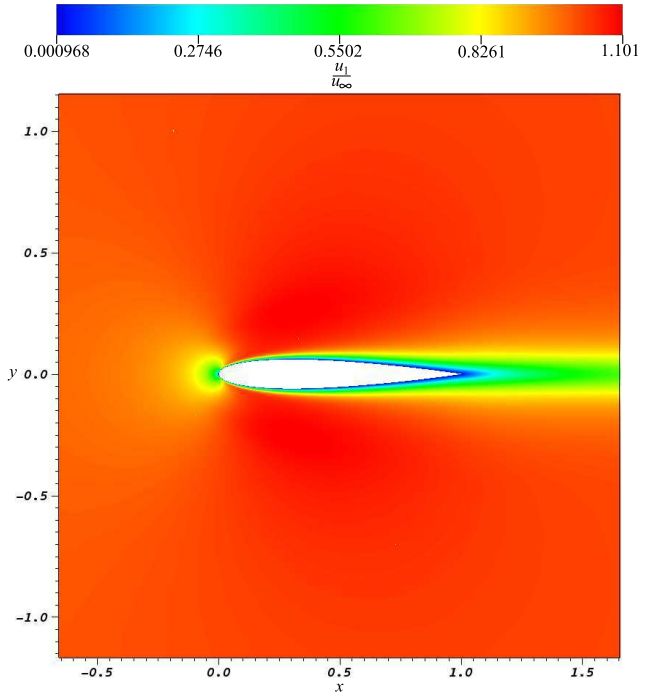

Figure 6.6.16: Colour plot of the scaled axial velocity $\frac{u_1}{u_\infty}$, computed on $\mathcal{T}_h^{\text{final}}$. $Re = 23,000$.

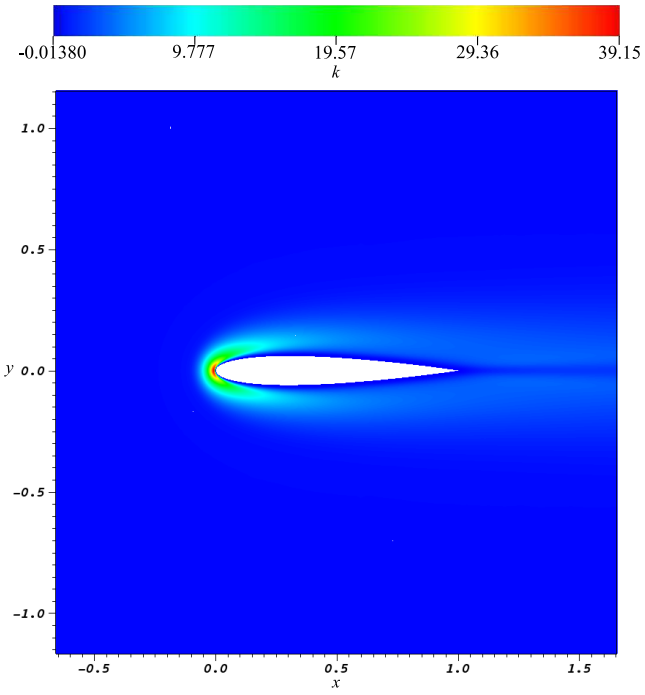

Figure 6.6.17: Colour plot of the non-dimensional turbulent kinetic energy k, computed on $\mathcal{T}_h^{\text{final}}$. $Re = 23,000$.

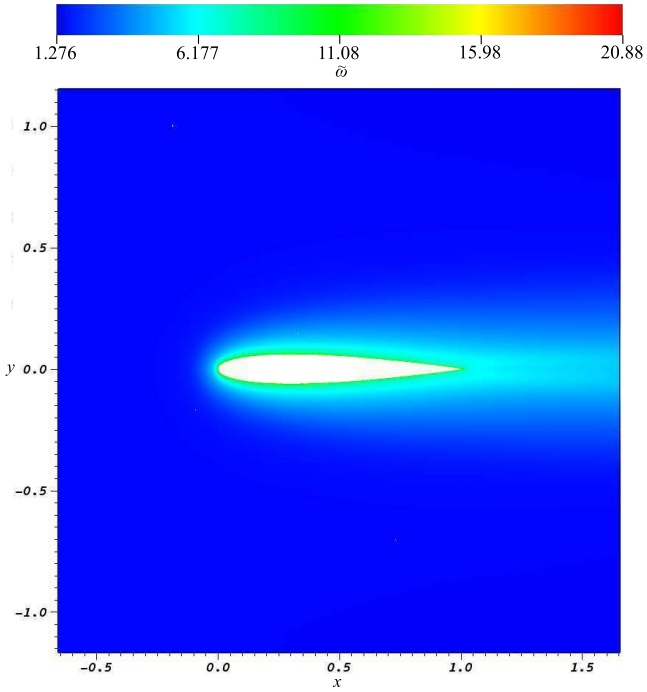

Figure 6.6.18: Colour plot of the non-dimensional logarithm of the dissipation per unit turbulence kinetic energy $\tilde{\omega}$, computed on $\mathcal{T}_h^{\text{final}}$. $Re = 23,000$.

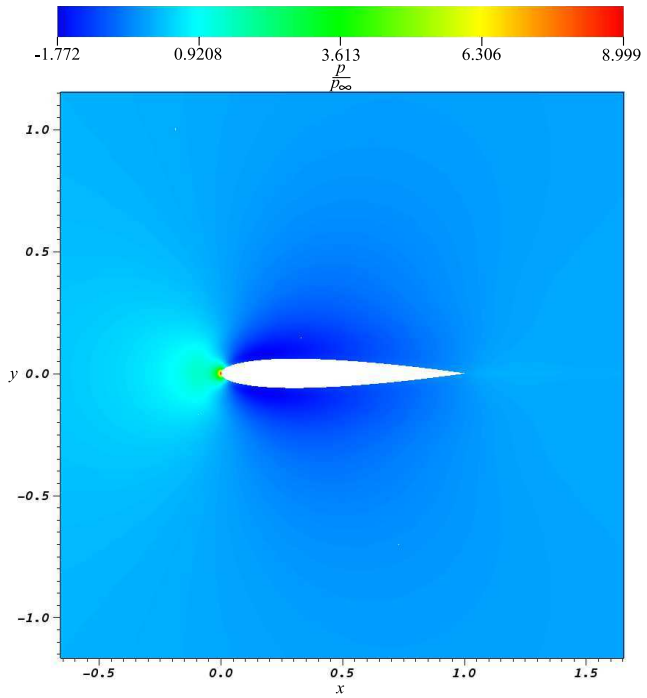

Figure 6.6.19: Colour plot of the scaled static pressure $\frac{p}{p_\infty}$, computed on $\mathcal{T}_h^{\text{final}}$. $Re = 23,000$.

Observation of the numerical results in Figures 6.6.16, 6.6.17, 6.6.18 and 6.6.19, compared to the mesh density of $\mathcal{T}_h^{\text{final}}$ in Figure 6.6.15, show good agreement between rapid changes in solution values and increased mesh density. In particular, the tip of the aerofoil leading edge is heavily targeted by the refinement algorithm to account for the spike in static pressure p. A comparison of these results is shown in Figure 6.6.20. Upstream from this mesh area, another patch of refinement accounts for the high turbulent kinetic energy k ahead of the aerofoil tip. However, due to restrictions placed on computational resources, we were unable to allow further refinement of the mesh around the blade edge to better represent the solution of $\tilde{\omega}$.

Additionally, Figure 6.6.20 demonstrates the improvement in solution accuracy through utilising the proposed automatic refinement technique; see Section 6.3. The area of the domain around the blade tip is notorious for large and very rapid changes in pressure, making capturing an accurate solution here very difficult. With regards to compressible flows, this is also the area where shocks are likely to develop, presenting further challenges to any proposed techniques.

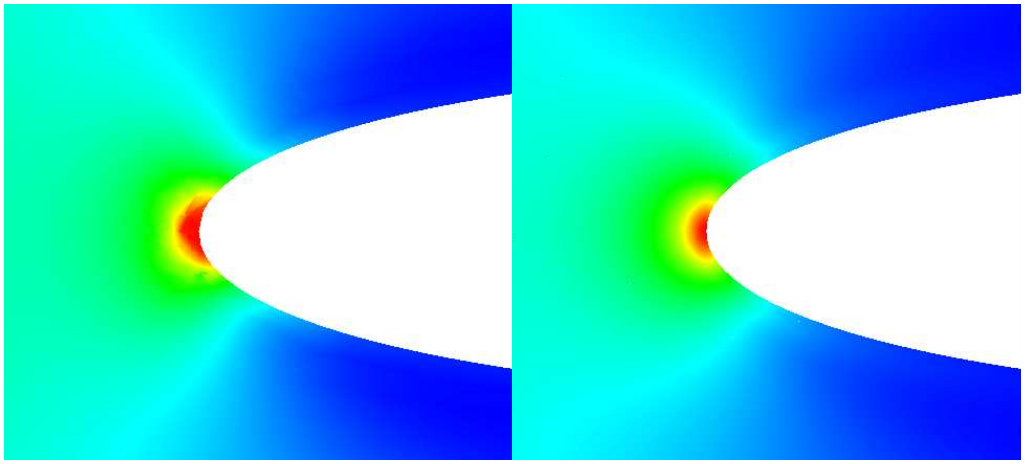

Figure 6.6.20: Colour plot of the scaled static pressure $\frac{p}{p_\infty}$, $Re = 23,000$. Left - $958,219$ degrees of freedom. Right - $2,338,800$ degrees of freedom.

To complete the analysis, we consider laminar flows to validate the effectiveness of (6.5.1) as a target functional. By removing the turbulent variables, in particular $\tilde{\omega}$, we require fewer mesh elements perpendicular to the aerofoil edge to accurately represent the numerical solution, since the velocity and pressure solutions change more gradually as they approach the boundary.

In this numerical experiment, the inlet conditions are replaced with a laminar free stream flow, with mean velocity $u_\infty = 13.909$ in the axial direction. The angle-of-attack reflects

the previous experiment with $\alpha = 0°$. The continuation algorithm proposed in Section 4.4.3 is used, except that we do not calculate a turbulent solution, and instead evolve the initial laminar solution until the target Reynolds number $Re = 23,000$ is reached. The setup of the numerical experiment mirrors the previous, with the schematic shown in Figure 6.6.1, and the initial mesh $\mathcal{T}_h^{\text{initial}}$ shown in Figures 6.6.3 and 6.6.4.

The continuation solution is then refined according to the algorithm in Section 6.6.3, coarsen the 10% of elements with the smallest approximate absolute error value, and refine isotropically the 20% of elements with the largest approximate absolute error value. The approximate absolute error value, for each new mesh that the solution is calculated on, is displayed in Table 6.2 and plotted in Figure 6.6.21.

Degrees of Freedom	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$
704,310	0.260202
918,690	0.111816
1,191,930	0.077346
1,558,050	0.059951
2,031,000	0.048782

Table 6.2: Table showing the error bound with respect to the target functional (6.5.1) for a laminar flow with $Re = 23,000$.

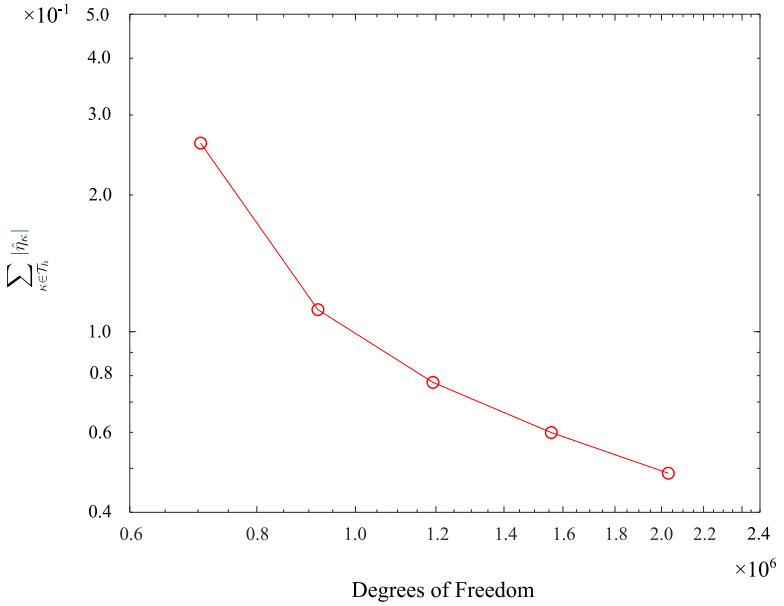

Figure 6.6.21: Convergence of the adaptive refinement algorithm, in particular, the approximate absolute error in the functional of interest.

Table 6.2 shows a decrease in the approximate absolute error value as the number of degrees of freedom increases. This is expected, since the near-wall velocity and pressure solutions vary far less as they approach the aerofoil edges compared to the turbulent variable $\tilde{\omega}$ shown in Figure 6.6.18. Fewer mesh elements perpendicular to the boundary are required to accurately represent the velocity and pressure solutions compared to the number required for the $\tilde{\omega}$ solution. As such, we see a continued reduction in error values of the laminar solution (Table 6.2), as there is sufficient mesh density to represent the numerical solution of all the computed variables (u_1, u_2 and p). Whereas, in the case of the turbulent flow, a far greater mesh density is required to represent the $\tilde{\omega}$ solution accurately, resulting in the larger error values shown in Table 6.1. The performance of the refinement algorithm in Table 6.2 suggests that it is behaving as expected, increasing the accuracy of the numerical solution as the number of degrees of freedom is increased.

We now consider the performance of the refinement algorithm on an industrially relevant numerical example.

6.6.2 Turbine Stator Cascade

In this numerical test case, we seek to improve the accuracy of the DGFEM on the 1.5 Stage Aachen Turbine test case [135, 138, 65], presented initially in Section 5.4.1.

Consider the numerical test case previously presented in Section 5.4.1. We proceed in a similar fashion as in Section 5.4.1, defining the geometry of the two-dimensional stator blade passage of the 1.5 Stage Aachen Turbine. See Figure 6.6.22. In particular, we consider the mid-height flow between the initial stator blades, with a pitch $s_s = 0.0476\,\text{m}$. The initial stator section is designed to receive an inlet flow parallel to the horizontal axis, such that the flow is axial. The design Reynolds number, based on the chord length and exit velocity is $Re = 6.8 \times 10^5$.

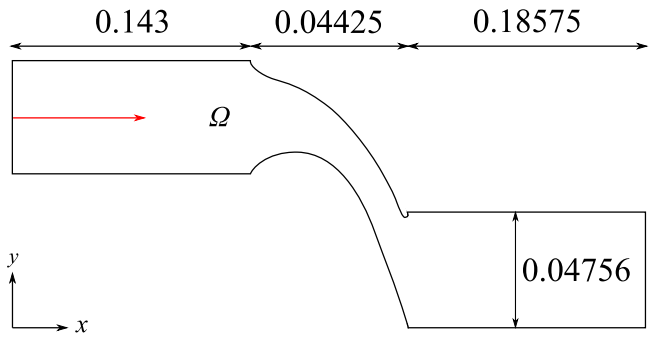

Figure 6.6.22: Aachen turbine stator cascade geometry [138]. All lengths are stated in metres.

The mesh for this test case consists of an inlet channel of length $l_{in} = 0.143\,\mathrm{m}$, and an outlet channel of length $l_{out} = 0.18575\,\mathrm{m}$. The outlet length is chosen such that the flow may fully develop before passing through the domain outflow boundary. The blade passage is specified using a cloud of point data [137], detailing the dimensions of the pressure and suction side of the blade. These data points are then used as nodes to create piecewise-cubic splines to approximate the curvature of the physical stator blade.

We partition the domain $\Omega \subset \mathbb{R}^2$ into a coarse mesh $\mathcal{T}_h^{\text{initial}}$ consisting of $13,250$ quadrilateral elements, see Figure 6.6.23. Again, periodic boundary conditions are used for the straight horizontal boundaries, whilst the blade edges are defined using a no-slip condition. A stress-free condition is applied to the domain outlet and a free-stream turbulent flow condition is used on the inlet located at $x = -0.143\,\mathrm{m}$.

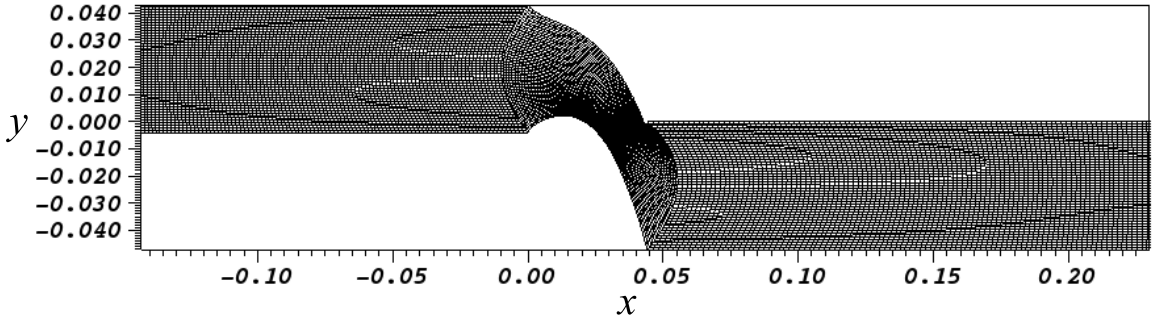

Figure 6.6.23: Initial mesh $\mathcal{T}_h^{\text{initial}}$ consisting of $13,250$ quadrilateral elements.

As proposed in Section 6.6.1, we employ a continuation strategy increasing the Reynolds number of the flow by 10% after each successive solve, along with a mesh refinement/coarsening strategy after every 10 successive solves. We coarsen the 5% of elements with the smallest approximate absolute error values according to (6.5.1), and refine isotropically the 10% of elements with the largest error values. We remark that the reduced refinement values compared to Section 6.6.1 is due to the number of elements positioned between the stator blades compared to the rest of mesh. The reduced values ensure that any refinement is targeted in the correct areas of the mesh, preventing over refining of the solution. The target functional is then computed on the domain outlet located at $x = 0.22998\,\mathrm{m}$. The numerical solution at each continuation step is calculated using a Newton based iterative solver, where the stopping condition is defined such that the l_2-norm of the nonlinear residual vector is reduced by 8 orders of magnitude.

The final mesh $\mathcal{T}_h^{\text{refined}}$ consists of $34,788$ quadrilateral elements and is shown in Figure 6.6.24, alongside a comparison to the initial mesh $\mathcal{T}_h^{\text{initial}}$.

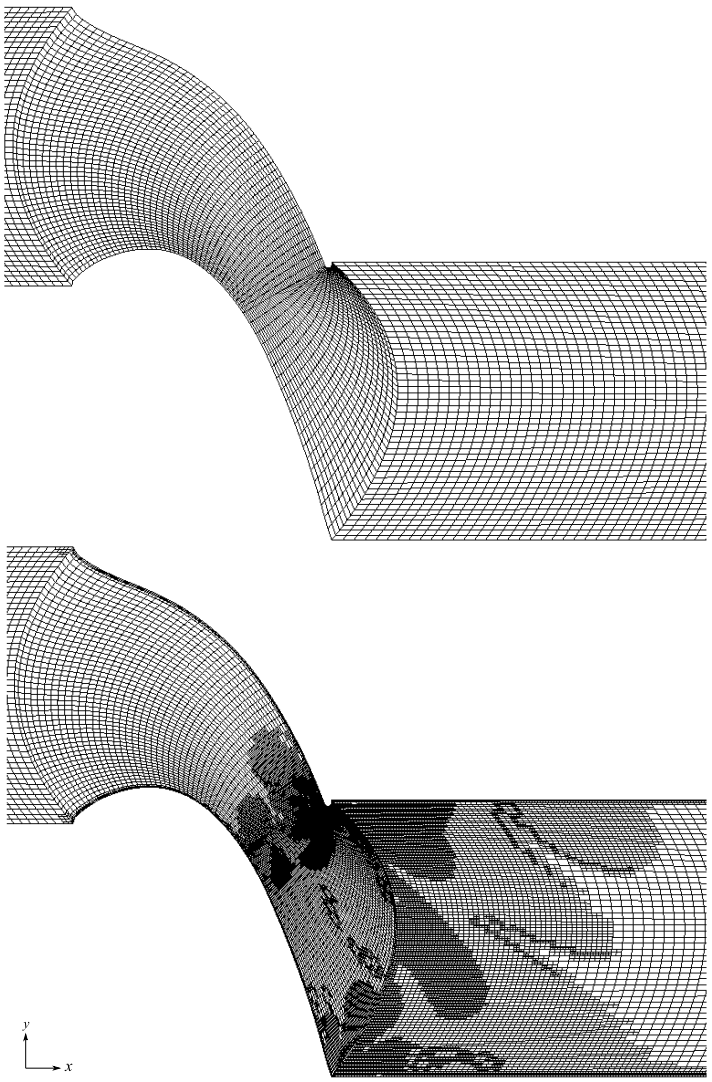

Figure 6.6.24: Comparison of initial mesh $\mathcal{T}_h^{\text{initial}}$ consisting of $13,250$ quadrilateral elements (top), and the final refined mesh $\mathcal{T}_h^{\text{refined}}$ with $34,788$ quadrilateral elements (bottom).

Due to the manner in which the blade profile has been defined using cubic spline approximations, we needed to place a discontinuity close to the blade trailing edge on the pressure side. This is located at at the point $x = 0.04398\,\text{m}$ and $y = -0.00042\,\text{m}$. As such, the mesh refinement algorithm has identified this as a source of numerical error, resulting in a dense packing of elements close to it. However, the algorithm also refined, as expected, the blade edges where the boundary layer forms, as well as, behind the stator trailing edge, where we expect to see vortices forming in the physical experiment. We present colour plots of the numerical solution in Figures 6.6.25, 6.6.26 and 6.6.27, of the axial velocity u_1, the vertical velocity u_2 and the static pressure p, respectively.

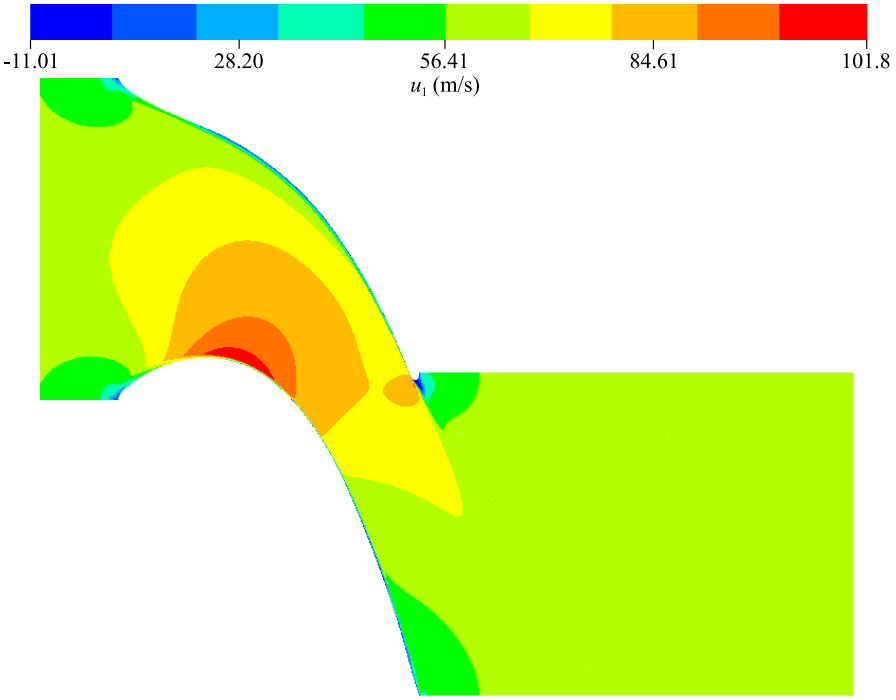

Figure 6.6.25: Colour plot of the axial velocity in the region of interest of $\mathcal{T}_h^{\text{refined}}$. $Re = 6.8 \times 10^5$.

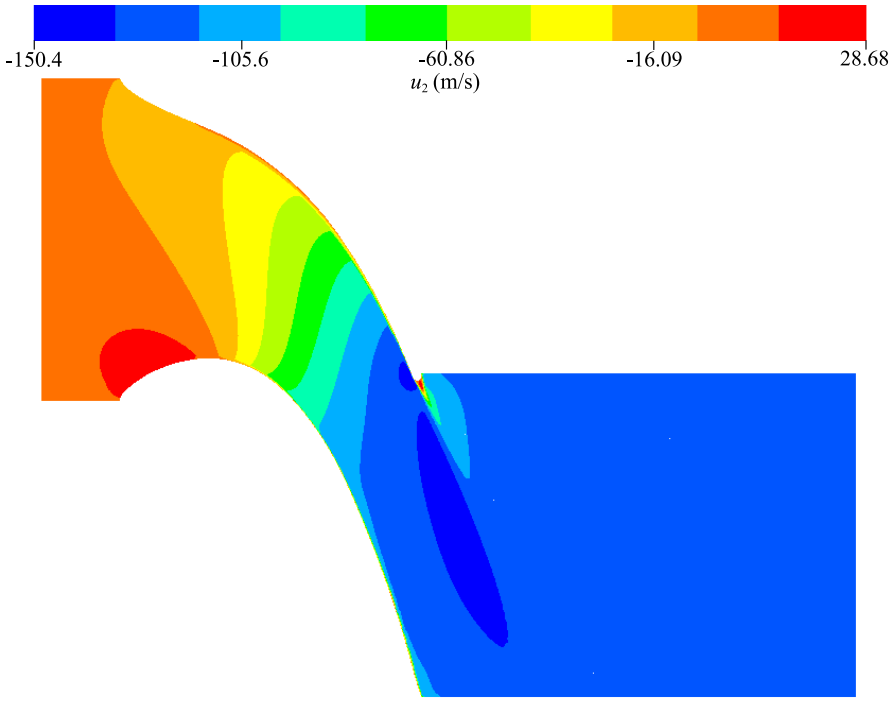

Figure 6.6.26: Colour plot of the vertical velocity in the region of interest of $\mathcal{T}_h^{\text{refined}}$. $Re = 6.8 \times 10^5$.

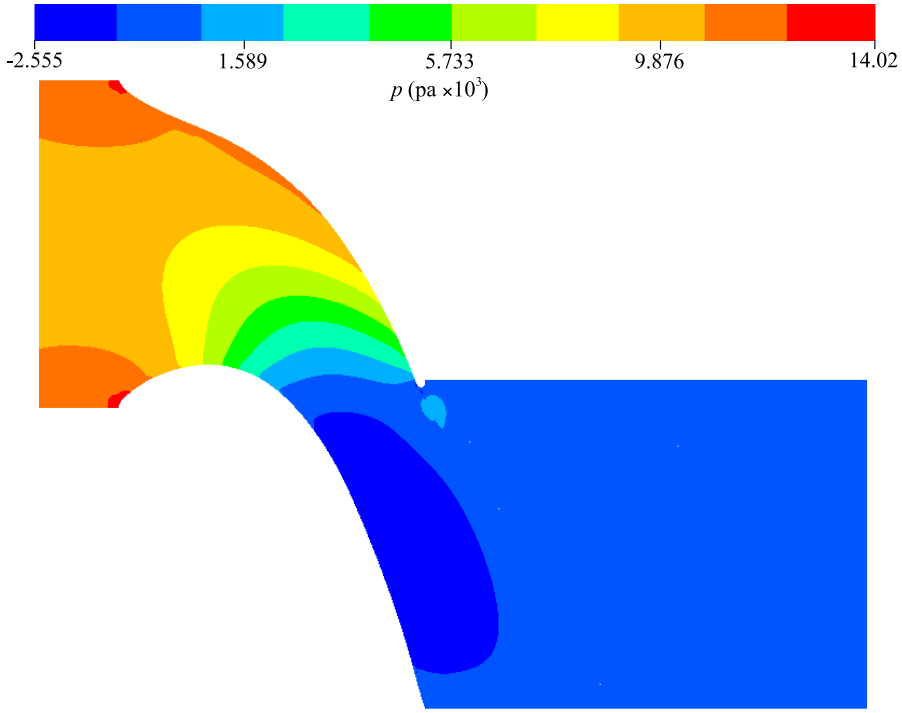

Figure 6.6.27: Colour plot of the static pressure in the region of interest of $\mathcal{T}_h^{\text{refined}}$. $Re = 6.8 \times 10^5$.

Additionally we compute the absolute pitchwise flow angle at the midspan of the blades, from the centre of the flow passage, 0.008 m behind the first stator, to be 19.55°. Compared to the numerical results presented in Section 5.4.1, we see an improvement in the numerical approximation of the flow direction, down from 19.86°. This is still an over estimate of the experimental value of 19.4°[138], but we are limited in this test case to a two-dimensional approximation of a three-dimensional flow. As such, various three-dimensional flow features are not represented in the solutions in Figures 6.6.25, 6.6.26 and 6.6.27, contributing to the numerical inaccuracy.

Further, we compute an approximation $J(\mathbf{u}_{h,\text{fine}}) = 224.218977$ to the target functional $J(\mathbf{u})$ on a fine uniform mesh consisting of 212,000 quadrilateral elements. Approximations to the absolute and relative error values are shown in Table 6.3.

Degrees of Freedom	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$	$\frac{\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert}{\lvert J(\mathbf{u}_{h,\text{fine}})\rvert}$
1,590,000 (Original mesh, Figure 5.4.3)	2.651334	1.182475×10^{-2}
1,046,310 (Final mesh $\mathcal{T}_h^{\text{refined}}$, Figure 6.6.24)	1.523670	6.795455×10^{-3}

Table 6.3: Comparison of the approximate absolute and relative error values compared to the functional of interest.

This numerical experiment demonstrates that we are able to achieve a greater degree of numerical accuracy with fewer degrees of freedom compared to the results in Section 5.4.1, through a targeted h-refinement strategy. In particular, we allow significant coarsening of the mesh in the inflow $x < 0$, and outflow $x > 0.05$ sections, resulting in a denser mesh around the blade edges and behind the trailing edge. This has the added benefit of requiring fewer degrees of freedom overall, compared to the original mesh in Figure 5.4.3. Additionally, Table 6.3 shows a significant reduction in numerical error using the DWR refinement method.

The use of standard elements limits the degree to which we are able to coarsen the mesh, particularly at the domain inlet and outlet boundaries, as we are unable to coarsen beyond the initial mesh density. Additionally, due to the mesh construction method, we require a single mesh element face to connect two data points from the cloud of data representing the blade geometry. As these points are packed relatively close together, the initial mesh has a far denser packing of elements parallel to the blade edges than we require. Also, as we do not know the exact equation describing the blade edges, we are unable to reduce the mesh density during the construction of the initial mesh, as this would likely result in misrepresentation of the blade geometry. Therefore, following the implementation of the refinement algorithm, we have a final mesh $\mathcal{T}_h^{\text{refined}}$ which has large areas of over-refinement. To tackle this issue, we consider polygonal mesh elements in Chapter 7, which are able to be coarsened beyond the initial mesh, by way of mesh agglomeration.

6.7 Concluding Remarks

The numerical experiments conducted in Section 6.6 suggest that the DWR approach to solution refinement is effective for turbulent incompressible flows. In particular, the refinement algorithm outlined in Section 6.3, is useful from an industrial perspective, allowing engineers to calibrate the solver to produce solutions of a required degree of accuracy. Additionally, the ability to switch the target functional to suit the problem, adds another layer of flexibility to the DGFEM.

The first numerical experiment in Section 6.6 validated the numerical solver on the NACA 0012 aerofoil for various angles of attack, with the refinement algorithm providing better approximations to the boundary layer flow, as shown in Figures 6.6.9 and 6.6.10. The coupling of the refinement algorithm with the continuation algorithm from Chapter 4 proved to be effective, reducing the number of nonlinear solves required by the numerical solver. It seems that the ability of the DGFEM to capture flow features as they develop with the increasing Reynolds number, is far more effective than simply refining the numerical solution once the target Reynolds number has been reached. We do note here, that there

is some trade off with the final solution accuracy, with the possibility of the refinement passes at low Reynolds numbers capturing features that do not exist in the flow at the target Reynolds number. Areas of the mesh that have been needlessly refined at low Reynolds numbers, are not necessarily coarsened in later refinement passes, with areas of laminar flow far away from the aerofoil taking priority. As such, moving forward, we wish to reduce the possibility of this affecting the efficiency of the numerical solver. Therefore, we limit the number of refinement passes made while the Reynolds number of the flow is below the fully turbulent threshold. We allow a single pass below this threshold, which orientates the mesh density of the initial mesh, to account for the differing angles-of-attack. In the case where we use a symmetric mesh for a zero angle-of-attack $\alpha = 0^\circ$ aerofoil, we skip this step as the mesh density is suitable to achieve a fully turbulent flow. An alternative approach would be to increase the percentage of elements that are coarsened by the algorithm to try to counteract this over-refinement of the final mesh. However, during development, we found that if the percentage of coarsened elements was too high, the Newton solver would fail to converge on the new mesh. Detailed below, is the hybrid DWR continuation algorithm we have developed.

Hybrid Continuation Algorithm

1. Choose a target Reynolds number and decide upon the inlet conditions for the final simulated flow. Generate the initial mesh of the geometry. Consider the angle-of-attack of the blade/aerofoil.

2. Solve on the initial mesh with the inlet conditions chosen above, a low Reynolds number ($50 \leq Re \leq 300$) laminar flow, without calculating the turbulence model variables k and $\tilde{\omega}$. The initial estimate of the numerical solution required for the nonlinear solver should be 0 for all variables.

3. Solve on the initial mesh, with the same inlet conditions and Reynolds number, a turbulent flow using the previous laminar solve as the initial guess for the nonlinear solver. The boundary conditions for the turbulent variable $\tilde{\omega}$ should be chosen according to the channel flow results for the appropriate Reynolds number.

4. Find the Reynolds number for which the flow transitions to be fully turbulent for the chosen geometry. Select a suitable target functional, as well as, a mesh refinement percentage, $U\%$, and a coarsening percentage, $L\%$.

5. Increase the Reynolds number of the flow, adjust the boundary conditions for $\tilde{\omega}$ and solve on the previous mesh. The value by which one is able to increase the Reynolds number varies depending on how similar the previous numerical solution is to the new solution. In general, we found an increase of 10% to be suitable. However, if

the nonlinear solver fails to converge, this value should be decreased by 50% until it converges. Repeat this step until the Reynolds number reaches the value for the first mesh refinement pass.

6. If the angle-of-attack $\alpha \neq 0°$, consider whether the mesh density of the initial mesh is suited to the current angle-of-attack. In particular, consider whether there is sufficient mesh density at the point where the aerofoil first meets the inlet flow. If there is not, refine the solution before the Reynolds number exceeds the value at which the flow becomes fully turbulent. Based on experience, we recommend a value $Re \geq 1000$, but less than the turbulent transition value.

7. Decide how many refinement passes are required in total, according to the choice of $U\%$ and $L\%$. Choose Reynolds numbers at which to perform each refinement pass, such that they are spaced uniformly up to the target Reynolds number. We recommend that the first of these refinements should be carried out soon after the fully turbulent transition value, $Re \approx 3,500$.

8. Refine the mesh. Calculate the numerical error according to the chosen target functional using the DWR method. Then, refine the $U\%$ of elements with the largest error values, and coarsen the $L\%$ of elements with the smallest error values.

9. Solve on the new refined mesh for the same flow parameters.

10. Increase the Reynolds number of the flow and solve. Repeat until the Reynolds number of the next refinement pass is reached, then repeat steps 8, 9 and 10; or until the target Reynolds number is reached.

The proposed algorithm is effective in producing accurate numerical results, as shown in Figures 6.6.9 and 6.6.10. However, the final mesh does still benefit from additional refinement passes, as shown in Table 6.1. In particular, the numerical solution of $\tilde{\omega}$ requires a significantly higher mesh density in order to represent all the flow features, with the solution changing rapidly close to the aerofoil boundaries. This point is reinforced when we consider laminar flows around the aerofoil, eliminating the turbulent variables, demonstrating significantly lower absolute error values in Table 6.2 than in Table 6.1. This is due to the physical variables, the velocity $\mathbf{u}$ and pressure p, not requiring the same mesh density as the turbulent variables, meaning that the error decreases at a much higher rate than it does for turbulent flows.

In Section 6.6.2, we applied the techniques developed in the previous numerical example to the 1.5 Stage Aachen Turbine test case [135, 138, 65]. This test case is representative of high Reynolds number industrial flow problems, and presents a number of challenges for DGFEM methods. The creation of the initial coarse mesh was particularly challenging,

requiring cubic approximations to the blade boundary in order to guide the adaptive refinement algorithm. The use of a discontinuity along the blade edge provided an unwanted source of numerical error, resulting in an overly dense packing of mesh elements around the trailing edge of the blade. However, through the proposed algorithm, we were able to reduce both the numerical error according to the target functional (6.5.1), as well as reduce the number of degrees of freedom of the problem. Additionally, we note some improvement in the accuracy of the physical flow parameters approximated by the DGFEM.

The effectiveness of the hybrid continuation algorithm is dependent on the quality of the initial mesh, limited in its ability to coarsen beyond the initial mesh width h. As such, further improvements to the accuracy of the numerical solution are limited by the amount of computational resources available. In order to reduce the memory demands of the DGFEM method, we consider an approach using polygonal meshes in Chapter 7.

Chapter 7

Extension to Polytopic Meshes

We now consider an extension of the previous developments to DGFEMs on computational meshes consisting of general polygonal, or polyhedral elements in two and three dimensions, respectively. From this point on, we shall simply refer to them as polytopic elements for brevity.

The design of computational meshes is clearly an important part of any fluid simulation, be it for FEMs or FVMs. The mesh must be fine enough to provide suitable resolution of the flow features, whilst representing the domain geometry to a required degree of accuracy, all whilst being coarse enough to be solved in reasonable time constraints. Traditional mesh generators create a triangulation on a domain that consists of triangular or quadrilateral elements in two dimensions, or tetrahedral, hexahedral, prismatic, or pyramidal elements in three dimensions. These are collectively referred to as standard mesh generators, and are a staple of industrial FEM/FVM solvers. These meshing techniques typically have problems in areas of the geometry with high curvature, where mesh refinement is likely to be concentrated, sometimes resulting in elements that self intersect (Figure 7.0.1). In order to avoid these problems, the mesh must either be designed with a dense packing of elements in these areas, or through the computationally expensive, isoparametric element mappings, that ensure the careful propagation of the domain geometry onto the interior elements. In an ideal situation, we would wish to avoid both these scenarios.

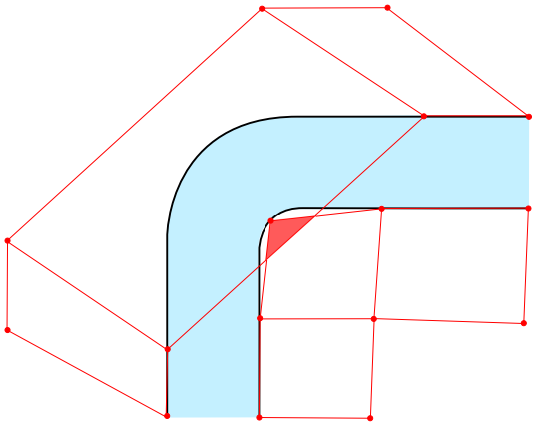

Figure 7.0.1: Self intersecting mesh around a pipe bend.

The use of polytopic elements on complicated geometries allows for a coarser mesh to be used, whilst allowing the use of higher order polynomials in the numerical solution to still achieve high accuracy. This increases the accuracy of the solution, without needlessly adding extra degrees of freedom in the form of more mesh elements. In many such situations, an extremely large number of standard elements may be required to produce a mesh, without accounting for geometrical substructures or flow features. This is very expensive computationally, and is often the case with aerodynamic models, such as those used to model the flow around aerofoils cf. [25, 80, 84]. In general, polytopic elements allow for coarser meshes to be used, and in turn, the better exploitation of DGFEMs computational flexibility, especially in areas of high geometry curvature and domains with microstructures. They also offer advantages in moving domains, such as those used in fluid-structure interaction [7], geophysical problems such as earthquake engineering [5, 7], and flows through fractured, porous media [38, 39].

A number of prominent techniques have been proposed for the design of FEMs posed on polytopic meshes. Composite Finite Elements (CFEs) were originally proposed by Hackbusch and Sauter [75, 76] for the discretisation of PDEs in complex geometries, within the conformal setting. These techniques have since been generalised to the discontinuous setting in [4, 5, 71]. CFEs are defined on general meshes that consist of polytopic elements which are generated via the agglomeration of standard elements. Another technique, much in the same vein as CFEs, is agglomerated DGFEMs [27, 28, 29]. These methods are very closely related to the DGFEM CFE developed in [4], although the CFE methodology admits more general classes of elemental shape functions. An alternative approach, supporting the use of general meshes, is the so called Hybrid High-Order method [51, 52, 53]. There are also two further approaches within the conformal setting, the Polygonal FEM and the Extended FEM [64, 129]; whereby, conformity is achieved by enriching the stan-

dard polynomial finite element spaces, similar to the original work of Babuška and Osborn [19]. Usually, the ability to handle general meshes is coupled with an increased computational workload. However, this problem was overcome with the newly proposed Virtual Element Method [1, 32, 33, 40], which navigates this difficulty, extending conforming FEMs to polytopic meshes, whilst maintaining many features associated with traditional FEMs.

In this chapter, we briefly introduce the tools required to formalise, and make mathematically rigorous, the concept of polytopic meshes in the DGFEM setting. Through the use of inverse estimates, we study the stability and convergence properties of DGFEMs on polytopic meshes, before considering the implementation of these techniques through the use of numerical examples. The key question of the discontinuity-penalisation function definition in the context of polytopic meshes, possibly containing very small spaces is of central importance in the discussion, as a poor choice can lead to either unstable or inaccurate discretisations.

7.1 Mesh Generation

We begin by defining a general class of computational mesh consisting of polytopic elements. Let $\mathcal{T}_h$ be a partitioning of the computational domain $\Omega \subset \mathbb{R}^d$, $d \in \mathbb{N}$, into disjoint open polytopic elements κ constructed in such a manner that the union of the closures of elements $\kappa \in \mathcal{T}_h$ forms a covering of the closure of Ω, that is $\bar{\Omega} = \cup_{\kappa \in \mathcal{T}_h} \bar{\kappa}$. We also define $h_\kappa := \text{diam}\,(\kappa)$ as the diameter of the element $\kappa \in \mathcal{T}_h$. One advantage of DGFEMs, is that they can naturally handle meshes containing hanging nodes. As such, we define $\mathcal{T}_h$ in such a manner, that it may contain several hanging nodes on their $(d-k)$-dimensional facets, $k = 1, 2, 3, ..., d-1$. Therefore, hanging nodes are revamped as normal nodes of polytopic elements between two faces of angle π [38]. A crucial attribute of the DGFEM described below is that it can handle elements of arbitrary angles between faces, because physical polynomial element spaces are used as opposed to mapped spaces from some reference element.

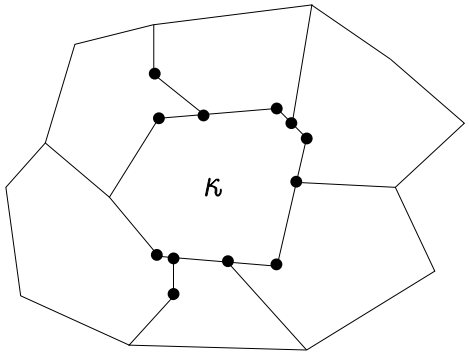

Figure 7.1.1: A polygonal element κ, along with its neighbours. Hanging nodes are denoted by a filled black circle.

In order to derive stability and approximation theorems, we proceed as in [38], considering each polygonal element as a collection of standard element shapes; in particular, as simplices. To this end, we define both element interfaces and element faces. Firstly, we define element faces as simplices in $\mathbb{R}^{d-1}$. Then, in order to allow for hanging nodes, we define the interfaces of the computational mesh $\mathcal{T}_h$ to be the intersection of $(d-1)$-dimensional facets of neighbouring elements. In the two-dimensional setting, the mesh interfaces are piecewise linear line segments. In general, an element interface consists of a set of $(d-1)$-dimensional simplices. Each element interface may be broken down into a collection of element faces, whose union forms the faces of the computation mesh $\mathcal{T}_h$. This union is denoted as $\mathbb{F}_h$ in the following analysis.

There are a number of practical solutions to the problem of generating general meshes consisting of polytopic elements. One such method, is through creating a Voronoi tessellation of the underlying geometry, [57, 131]. However, this approach also has the problem that it is not trivially extendable to three dimensional meshes. Often a far more flexible approach to producing such meshes, is to take advantage of existing mesh generators using standard elements. A geometry-conforming, fine mesh $\mathcal{T}_h^{\text{fine}}$, consisting of standard elements is generated initially. Then the polytopic elements are formed through agglomeration of the existing standard element faces, as shown in Figure 7.1.2.

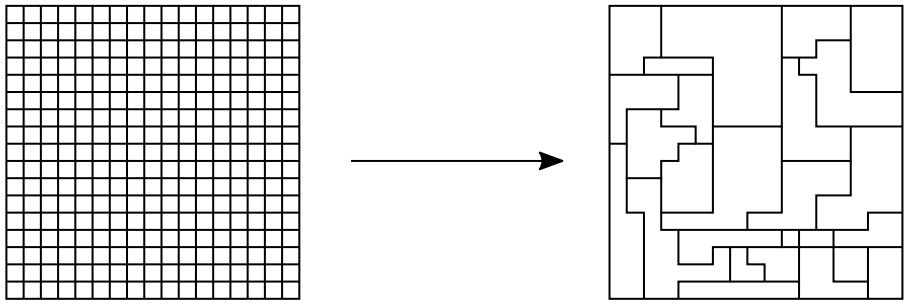

Figure 7.1.2: Agglomeration of a computational mesh $\mathcal{T}_h^{fine}$, consisting of quadrilateral elements.

Formally, there exist two key approaches to polytopic mesh generation. The first is a hierarchy of overlapping reference and logical meshes consisting of standard shaped elements, which are constructed based on successive adaptive refinement of elements that intersect the boundary $\partial\Omega$ of the computational domain Ω. Then, once a suitably fine mesh has been constructed, possibly by moving nodes onto $\partial\Omega$, a sequence of coarse geometry-conforming physical meshes, consisting of general polytopic elements, may be derived via agglomerating elements, which share the same parent within the underlying refinement tree. Examples of this approach can be found in [4, 5, 75, 76]. On the other hand, a fine mesh $\mathcal{T}_h^{\text{fine}}$ which consists of standard shaped elements may be constructed using a standard mesh generator, then, subsequently, elements can be agglomerated to form polytopes using graph partitioning algorithms. For the purposes of this study, we favour the second approach, utilising ParMETIS [96] to create a polytopic partition of the computational domain.

7.2 Inverse Estimates on Polytopic Meshes

In this section, we follow [38] in deriving new inverse inequalities on general polytopic elements. These inequalities must be robust enough to allow for element face degeneration, especially during mesh refinement, but also not result in excessively large penalisation terms in the underlying scheme. This is important as large penalisation in this context may lead to loss of accuracy and severe ill-conditioning. On the other hand, too small penalisation may not be enough to ensure the stability of the method. We begin this discussion by introducing the following inverse inequalities defined on simplices, before extending them to general polytopic elements.

Lemma 7.2.1. *Given a simplex φ in $\mathbb{R}^d$, $d = 2, 3$, we write $f \subset \partial\varphi$ to denote one of its faces. Then, for $v \in \mathcal{P}_p(\varphi)$, the following inverse inequality holds*

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv},1} p^2 \frac{|f|}{|\varphi|} \|v\|^2_{L^2(\varphi)}, \tag{7.2.1}$$

where $C_{\text{inv},1}$, is a positive constant, that is independent of v, p and h_κ.

Proof. The proof of (7.2.1) can be found in [139], whereby the precise estimate

$$\|v\|^2_{L^2(f)} \leq \frac{(p+1)(p+d)}{d} \frac{|f|}{|\varphi|} \|v\|^2_{L^2(\varphi)}, \tag{7.2.2}$$

is proven. □

Following [38], we now proceed to generalise (7.2.1) to general meshes consisting of polytopic elements. In order to do so, we must first introduce the following family of overlapping simplices associated with each face $f \subset \partial\kappa$.

Definition 7.2.2. For each element κ in the computational mesh $\mathcal{T}_h$, we define the family Ψ_b^κ of all possible d-dimensional simplices contained in κ and having at least one face in common with κ. Moreover, we write κ_b^f to denote a simplex belonging to Ψ_b^κ which shares with $\kappa \in \mathcal{T}_h$ the specific face $f \subset \partial\kappa$.

With the above definition, we may now employ (7.2.1) directly in order to deduce the corresponding inverse estimate on a general polytopic element. Therefore, given $\kappa \in T_h$, and a face $f \in \Psi_h$, such that $f \subset \partial\kappa$, consider $\kappa_b^f \in \Psi_b^\kappa$ given in definition 7.2.2. Then, for $v \in \mathcal{P}_p(\kappa)$, applying (7.2.1) on κ_b^f, we immediately deduce that

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv},1} p^2 \frac{|f|}{\left|\kappa_b^f\right|} \|v\|^2_{L^2\left(\kappa_b^f\right)} \leq C_{\text{inv},1} p^2 \frac{|f|}{\left|\kappa_b^f\right|} \|v\|^2_{L^2(\kappa)} \tag{7.2.3}$$

where $C_{\text{inv},1}$ is a positive constant, independent of v, $|f|$, $\left|\kappa_b^f\right|$, and p. Clearly, the choice of κ_b^f is not unique, and as such, we may select κ_b^f to have the largest possible measure $\left|\kappa_b^f\right|$. Hence, on the basis of (7.2.3), the following inverse inequality holds:

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv},1} p^2 \frac{|f|}{\sup_{\kappa_b^f \subset \kappa} \left|\kappa_b^f\right|} \|v\|^2_{L^2(\kappa)}. \tag{7.2.4}$$

We note that for a fixed element size h_κ, the inverse inequality (7.2.4) is sharp with respect to the polynomial degree p [139]. However, for a fixed polynomial order p, (7.2.4) lacks the sharpness with respect to $(d-k)$-dimensional facet degeneration, $k = 1, ..., d-1$. More precisely, it is not sensitive to the magnitude of the face measure relative to the measure of the polytopic element κ. To demonstrate this, we consider the two-dimensional example from [36].

Example 7.2.3. In order to demonstrate the lack of sharpness of the inverse inequality (7.2.4) with respect to one of its lower-dimensional facets degenerating, we consider the quadrilateral domain κ given by

$$\begin{aligned}\kappa := &\left\{(x,y) \in \mathbb{R}^2 : x > 0,\, y > 0,\, x + y < 1\right\} \\ &\cup \left\{\left((x,y) \in \mathbb{R}^2 : x > 0,\, y \leq 0,\, x - y < \epsilon\right)\right\},\end{aligned}$$

for some $\epsilon > 0$, see Figure 7.2.1.

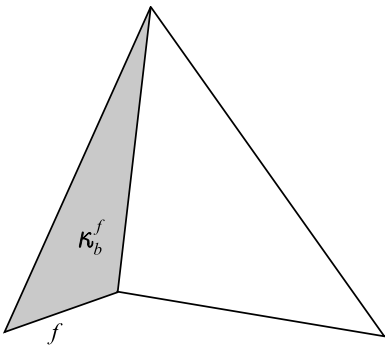

Figure 7.2.1: The quadrilateral element κ.

Given $v \in P_p(\kappa)$, let $f := \left\{(x,y) \in \mathbb{R}^2 : x - y < \epsilon\right\}$ and using (7.2.4), we have

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv},1} \frac{\sqrt{2}p^2\epsilon}{\left|\kappa_b^f\right|} \|v\|^2_{L^2(\kappa)}, \tag{7.2.5}$$

where $\kappa_b^f := \left\{(x,y) \in \mathbb{R}^2 : x > 0,\, x + \epsilon y < \epsilon,\, x - y < \epsilon\right\}$. We remark that $\left|\kappa_b^f\right| = \frac{\epsilon(1+\epsilon)}{2}$, which means that (7.2.5) becomes

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv},1} \frac{2\sqrt{2}p^2}{(1+\epsilon)} \|v\|^2_{L^2(\kappa)}.$$

Hence if we let $\epsilon \to 0$, the left-hand side $\|v\|^2_{L^2(f)} \to 0$, whereas the right-hand side $\frac{2\sqrt{2}p^2}{(1+\epsilon)} \|v\|^2_{L^2(\kappa)} \to 2\sqrt{2}p^2 \|v\|^2_{L^2(\kappa)} \neq 0$ in general.

The above suggests that the inverse inequality (7.2.4) may not be sharp with respect to element facets of degenerating measure. Therefore, if we were to employ such a bound to deduce the stability of the DGFEM approximation, it would typically lead to an excessively large penalisation term within the underlying scheme. This in turn may result in ill conditioning of the resulting system of equations and, possibly, loss of accuracy.

Therefore, we proceed by deriving an alternative inverse inequality under suitable mesh assumptions. We begin by noting that since $f \subset \partial\kappa_b^f$, we have

$$\|v\|^2_{L^2(f)} \leq |f| \, \|v\|^2_{L^\infty\left(\kappa_b^f\right)} . \tag{7.2.6}$$

In order to bound the right-hand side of (7.2.6), we need to introduce some additional requirements on the elements $\kappa \in \mathcal{T}_h$. These are based on the following result which represents the generalisation of Lemma 3.7 in [69] (see [38] for details).

Lemma 7.2.4. *Let K be a shape-regular simplex in $\mathbb{R}^d$, $d = 2, 3$. Then, for each $v \in \mathcal{P}_p(K)$, there exists a simplex $\hat{\kappa} \subset K$, having the same shape as K and faces parallel to the faces of K, with $\operatorname{dist}(\partial\hat{\kappa}, \partial K) > \frac{C_{as}\operatorname{diam}(K)}{p^2}$, where C_{as} is a positive constant, independent of v, K, and p, such that*

$$\|v\|^2_{L^2(\hat{\kappa})} \geq \frac{1}{2} \|v\|^2_{L^2(K)} \tag{7.2.7}$$

(We note that $\operatorname{dist}(\partial\hat{\kappa}, \partial K)$ denotes the Haussdorff distance between $\partial\hat{\kappa}$ and ∂K. That is to say, given two sets X and Y in $\mathbb{R}^d$, $d \geq 1$, we define the Hausdorff distance between X and Y as $\operatorname{dist}(X, Y) := \sup_{x\in X} \inf_{y\in Y} |x - y|$).

Proof. We omit the proof here for conciseness, but it can be found in [38]. □

As such, we proceed by recalling the following definition, before defining the inverse inequality on polytopic elements that allows for facet degeneration.

Definition 7.2.5. An element $\kappa \in \mathcal{T}_h$ is said to be p-coverable with respect to $p \in \mathbb{N}$, if there exists a set of m_κ overlapping shape regular simplices K_i, $i = 1, ..., m_\kappa$, $m_\kappa \in \mathbb{N}$, such that

$$\operatorname{dist}(\kappa, \partial K_i) < C_{as} \frac{\operatorname{diam}(K_i)}{p^2} \text{ and } |K_i| \geq c_{as} |\kappa| ,$$

for all $i = 1, ..., m_\kappa$, where C_{as} and c_{as} are positive constants, independent of κ and $\mathcal{T}_h$.

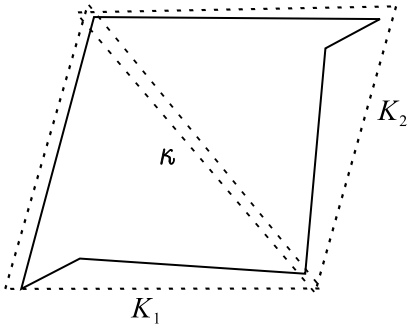

Figure 7.2.2: Illustration of Definition 7.2.5. This polygon is p-coverable for p sufficiently small, but not all p.

Following [36], in Figure 7.2.2 we present a polytopic element κ in $\mathbb{R}^2$ which may be covered by two triangles K_1 and K_2 with $m_\kappa = 2$. We note that Definition 7.2.5 admits very general polytopic elements $\kappa \in \mathcal{T}_h$ which may contain $(d-k)$-dimensional facets, $k = 1, ..., d-1$, whose measure is arbitrarily small, relative to the measure of κ itself. We note that the element in Figure 7.2.1 is p-coverable when $\epsilon < \frac{C_{as}}{p^2}$ for some constant $C_{as} > 0$.

Combining (7.2.4), (7.2.6), Lemma 7.2.4 and Definition 7.2.5, we may now present the following inverse inequality defined for general polytopic elements which directly accounts for elemental facet degeneration.

Lemma 7.2.6. *Let $\kappa \in \mathcal{T}_h$, $f \subset \partial\kappa$ denote one of its faces. Then, for each $v \in \mathcal{P}_p(\kappa)$, the following inverse inequality holds*

$$\|v\|^2_{L^2(f)} \leq C_{\text{inv}} p^2 \frac{|f|}{|\kappa|} \|v\|^2_{L^2(\kappa)}, \tag{7.2.8}$$

where

$$C_{\text{inv}} := \begin{cases} C_{\text{inv},4} \min\left\{ \frac{|\kappa|}{\sup_{\kappa_b^f \subset \kappa} \left|\kappa_b^f\right|}, p^{2(d-1)} \right\} & \text{if } \kappa \text{ is } p-\text{coverable}, \\ C_{\text{inv},1} \frac{|\kappa|}{\sup_{\kappa_b^f \subset \kappa} \left|\kappa_b^f\right|}, & \text{otherwise}, \end{cases} \tag{7.2.9}$$

with $\kappa_b^f \in \Psi_b^\kappa$, and $C_{\text{inv},1}$ and $C_{\text{inv},4}$ positive constants which are independent of $\frac{|\kappa|}{\sup_{\kappa_b^f \subset \kappa} \left|\kappa_b^f\right|}$, $|f|$, p, and v.

Proof. The proof is available in Lemma 11 in [38]. □

We remark, that (7.2.8) is sensitive with respect to $(d-k)$-dimensional facet degeneration, $k = 1, ..., d-1$. Recalling figure (7.2.1), the left- and right-hand sides of (7.2.8) degenerate at the same rate as $\epsilon \to 0$, for a fixed p.

7.3 The Discontinuity-Penalisation Function

For the partitioning of a domain Ω, we define $\mathscr{F}_h$ as the set of open $(d-1)$-dimensional element faces associated with $\mathcal{T}_h$, and we write $\mathscr{F}_h = \mathscr{F}_h^I \bigcup \mathscr{F}_h^B$, where $\mathscr{F}_h^I$ denotes the set of all element faces $f \in \mathscr{F}_h$ that are contained in Ω, and $\mathscr{F}_h^B$ is the set of boundary element faces. We decompose $\mathscr{F}_h^B = \mathscr{F}_h^- \bigcup \mathscr{F}_h^+ \bigcup \mathscr{F}_h^{\mathrm{D}} \bigcup \mathscr{F}_h^{\mathrm{N}}$, where $\mathscr{F}_h^-, \mathscr{F}_h^+, \mathscr{F}_h^{\mathrm{D}}, \mathscr{F}_h^{\mathrm{N}} \subset \mathscr{F}_h^B$ denote the subsets of boundary faces belonging to $\partial\Omega_-$, $\partial\Omega_+$, $\partial\Omega_{\mathrm{D}}$, and $\partial\Omega_{\mathrm{N}}$, respectively. Continuing the notation developed in Chapter 2, $\partial\Omega_-$ and $\partial\Omega_+$denote the inflow and outflow boundaries, whilst $\partial\Omega_{\mathrm{D}}$ and $\partial\Omega_{\mathrm{N}}$ denote the Dirichlet and Neumann boundaries, such that $\partial\Omega = \partial\Omega_- \bigcup \partial\Omega_+ \bigcup \partial\Omega_{\mathrm{D}} \bigcup \partial\Omega_{\mathrm{N}}$.

For each element $\kappa \in \mathcal{T}_h$, we define:

$$C_\kappa = \mathrm{card}\left\{ f \in \mathscr{F}_h : f \subset \partial\kappa \right\}.$$

We assume that there exists a positive constant C_f, independent of the mesh parameters, such that

$$\max_{\kappa \in \mathcal{T}_h} C_\kappa \leq C_f \tag{7.3.1}$$

We are now in a position to introduce the discontinuity-penalisation function for polytopic elements, using the inverse inequalities defined in the previous section.

Definition 7.3.1. Given that assumption (7.3.1) holds, we define the discontinuity-penalisation function $\sigma : \mathscr{F}_h^I \bigcup \mathscr{F}_h^{\mathrm{D}} \to \mathbb{R}$ by

$$\sigma(\mathbf{x}) := \begin{cases} C_\sigma \max\limits_{\kappa \in \{\kappa^+, \kappa^-\}} \left\{ C_{\mathrm{inv}} (\mu + \mu_t) \frac{p^2 |f|}{|\kappa|} \right\}, & \mathbf{x} \in f \in \mathscr{F}_h^I,\ f \subset \partial\kappa^+ \bigcap \partial\kappa^- \\ C_\sigma C_{\mathrm{inv}} (\mu + \mu_t) \frac{p^2 |f|}{|\kappa|}, & \mathbf{x} \in f \in \mathscr{F}_h^{\mathrm{D}},\ f \subset \partial\kappa \end{cases} \tag{7.3.2}$$

where μ is the molecular viscosity, μ_t is the turbulent viscosity, and C_σ is a sufficiently large positive constant [38].

7.4 Implementation

Before we proceed with some numerical experiments, we briefly discuss some of the implementation aspects of an interior penalty DGFEM for general computational meshes consisting of polytopic elements. Indeed, the construction of the elemental polynomial basis functions, as well as the quadrature rules governing the numerical integration, are not straightforward.

7.4.1 Basis Functions on Polytopes

In the usual setting, when the computational mesh $\mathcal{T}_h$ consists of standard shaped elements, the construction of the underlying finite element space is typically undertaken by mapping each element $\kappa \in \mathcal{T}_h$ to a specified reference element denoted by $\hat{\kappa}$ as in Figure 2.4.1. As such, local spaces of polynomials may be constructed on $\hat{\kappa}$ in a simple manner, subject to the enforcement of any inter-element continuity constraints. This approach is widely used in FEM software packages, even though the calculation of high-order derivatives of the computed numerical solution can be quite costly when non-affine element mappings are used.

However, DGFEMs can allow for elemental basis to be constructed within the physical element, without the need to map to a given reference element [28]. In particular, this allows them to admit general polytopic elements in a simple fashion. This attribute is of crucial importance as it allows for very general element shapes, containing even re-entrant corners, to admit optimal approximation properties. In particular, no maximum angle condition which is pertinent in finite element theory, is required.

In [28], basis functions are constructed on general meshes consisting of agglomerated elements, through the implementation of a Gram-Schmidt orthogonalisation process applied to a given set of polynomial functions defined on each $\kappa \in \mathcal{T}_h$. Alternatively, one could define polynomial spaces over a suitably chosen bounding box of the physical element κ; then construction of the element basis is achieved by restricting this space to κ. We use the latter approach for the remainder of this chapter.

Therefore, given a physical element $\kappa \in \mathcal{T}_h$, we define the Cartesian bounding box of κ, denoted by B_κ. That is to say, that the sides of the bounding box B_κ are aligned with the Cartesian axes, allowing for the simple construction of $\bar{\kappa} \subseteq \bar{B}_\kappa$, where κ is a polytopic element.

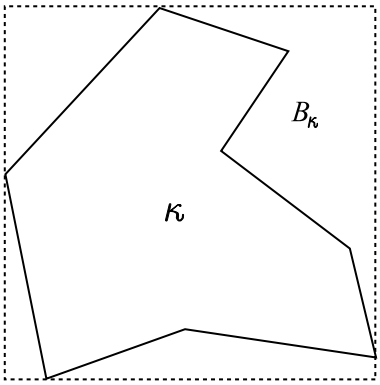

Figure 7.4.1: Bounding box B_κ of a polygonal element $\kappa \in \mathcal{T}_h$.

Further to this, we define on the bounding box B_κ, the standard polynomial space $\mathcal{P}_p(B_\kappa)$ spanned by a set of basis functions $\{\phi_{i,\kappa}\}$, $i = 1, ..., \dim(P_p(B_\kappa))$. We remark here, that the tensor-product polynomial spaces $\mathcal{Q}_p(B_\kappa)$ may also be employed, though in the absence of non-affine element mappings, the approximation order of both spaces is identical [38]. Then, writing $B_\kappa := \psi_1 \times \psi_2 \times ... \times \psi_d$, where $\psi_j := \left(x_1^j, x_2^j\right)$, $j = 1, ..., d$, and selecting $\hat{\kappa} := (-1, 1)^d$ to be the reference hypercube, the bounding box B_κ may be affinely mapped to $\hat{\kappa}$, via the mapping

$$\mathbf{x} = J_\kappa \hat{\mathbf{x}} + \mathbf{c}, \tag{7.4.1}$$

where $J_\kappa := \mathrm{diag}(h_1, ..., h_d)$, $\mathbf{c} := (m_1, ..., m_d)^T$, and $\hat{\mathbf{x}}$ is a general point in $\hat{\kappa}$. We remark that h_j, $j = 1, ..., d$, is half the length of the j^{th} side of B_κ respectively. We then follow [38], employing tensor-product Legendre polynomials, $\left\{\hat{L}_i(\hat{x})\right\}_{i=0}^{\infty}$ which are chosen to be $L^2(-1, 1)$-orthogonal, cf [124]. As such, the space of polynomials $\mathcal{P}_p(\hat{\kappa})$ of total degree p over $\hat{\kappa}$ is given by

$$\mathcal{P}_p(\hat{\kappa}) := \mathrm{span}\left\{\phi_{i,\kappa} : i = 1, ..., \dim(P_p(\hat{\kappa}))\right\}, \tag{7.4.2}$$

where $\hat{\phi}_{i,\kappa}(\hat{\mathbf{x}}) = \hat{L}_{i_1}(\hat{x}_1)\hat{L}_{i_2}(\hat{x}_2)...\hat{L}_{i_d}(\hat{x}_d)$, $i_1 + i_2 + ... + i_d \leq p$, $i_k \geq 0$, $k = 1, ..., d$, and $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_d)$. Writing $\hat{L}_{i_1}((x - m_j)/h_j)$, under the transformation (7.4.1), the space of polynomials $\mathcal{P}_p(B_\kappa)$ of total degree p over B_κ is given by

$$\mathcal{P}_p(B_\kappa) := \mathrm{span}\left\{\phi_{i,\kappa} : i = 1, ..., \dim(P_p(B_\kappa))\right\}. \tag{7.4.3}$$

It is important to note however, that the choice of bounding box B_κ is arbitrary, and that the choice of the Cartesian bounding box may be replaced with any other suitable choice, provided that the construction of the underlying polynomial basis remains simple. Other examples of bounding boxes, that may be considered, are ones for anisotropic polytopic

elements, where it may be advantageous to align the axes of the bounding box with the principle axes of the element.

7.4.2 Quadrature Rules

We also require the definition of a new quadrature rule for computational meshes consisting of polytopic elements. This is a particularly challenging task, and is discussed below, highlighting the three most prominent approaches.

7.4.2.1 Sub-Tessellation

The most general approach to the problem of quadrature over polytopic elements is to subdivide the elements into standard shapes, triangles and quadrilaterals in $\mathbb{R}^2$, hexahedra, tetrahedra, prisms and pyramids in $\mathbb{R}^3$. Then, standard quadrature rules may be applied to the sub-tessellation [36, 37, 94]. Consider an element $\kappa \in \mathcal{T}_h$, and let κ_T be a non-overlapping sub-tessellation of κ consisting of standard elements, such that $\kappa_T := \{\tau_\kappa\}$. In particular, a general hybrid sub-tessellation consisting of triangular and quadrilateral elements in $\mathbb{R}^2$, and tetrahedral, hexahedral, prismatic and pyramidal elements in $\mathbb{R}^3$ may be constructed. When making use of mesh agglomeration, the sub-tessellation is readily available; however, we may still wish to construct an alternative sub-tessellation, comprising a minimal number of sub-elements, as this improves computational efficiency. In general, however, quadrature schemes based on employing a sub-tessellation of each element often prove to be expensive, since if the cardinality of the sub-tessellation is large, then the required number of function evaluations tends to be large. This is the case when the background mesh $\mathcal{T}_h^{\text{fine}}$, which is agglomerated to form the coarse polytopic mesh $\mathcal{T}_h$, is also used to provide the sub-tessellation of each polytopic element. On the other hand, sub-tessellation may be the only option in the presence of heterogeneous PDE coefficients.

7.4.2.2 Moment Quadratures

Moment quadratures is one approach that employs a node elimination scheme, along with the least squares Newton method to try to boost the efficiency of quadrature rules based on sub-tessellation [146]. Here, we discuss the approach of [146] in greater detail.

Consider a polytopic element $\kappa \in \mathcal{T}_h$, along with a set of user defined functions $\Phi_\kappa = \{\phi_1, \phi_2, ..., \phi_n\}$, $n \geq 1$, defined over κ, as well as a quadrature rule $(\mathbf{x}_j, w_j)_{j=1}^{q_\kappa}$ on κ,

$q_\kappa \geq n$, which is assumed to integrate all functions in $\boldsymbol{\Phi}_\kappa$ exactly. Therefore, we have the following system of equations

$$\underline{\mathbf{A}}\mathbf{w} = \mathbf{I}, \tag{7.4.4}$$

where $\underline{\mathbf{A}}$ is an $n \times q_\kappa$ matrix with all entries $A_{ij} := \phi_i(\mathbf{x}_j)$, $i = 1, ..., n$, $j = 1, ..., q_\kappa$, $\mathbf{w} := (w_1, ..., w_{q_\kappa})^T$ is a vector of quadrature weights, and $\mathbf{I}$ is a vector of dimension n, with entries $I_i := \int_\kappa \phi_i \mathrm{d}\mathbf{x}$, $i = 1, ..., n$. We note that a weight function ω is included in the integral in [146], but here for simplicity, we have set $\omega \equiv 1$.

The general approach of this method is to optimise an initial quadrature rule through continuously eliminating points until the solution of (7.4.4) can no longer be determined. There are a number of different ways of selecting the initial quadrature rule $(\mathbf{x}_j, w_j)_{j=1}^{q_\kappa}$, indeed, the sub-tessellation approach discussed in the previous section may be employed. Formally, for each quadrature point and weight, the corresponding significance index s_j, $j = 1, ..., q_\kappa$ is computed via one of the following expressions as proposed in [146].

$$s_j := w_j \sum_{i=1}^{n} \phi_i^2(\mathbf{x}_j), \text{ or } s_j := \sum_{i=1}^{n} \phi_i^2(\mathbf{x}_j)$$

Then, the quadrature point and weight $(\mathbf{x}_j, w_j)$ which has the smallest significance factor is removed from the quadrature rule. The least-squares version of Newton's method is then applied to (7.4.4) to determine a new quadrature $(\mathbf{x}_j, w_j)_{j=1}^{q_\kappa - 1}$ with $(q_\kappa - 1)$ points and weights. This technique is continuously repeated until the Newton algorithm fails to converge, resulting in an optimised quadrature rule that can precisely integrate all of the functions present in the space $\boldsymbol{\Phi}_\kappa$.

An alternate approach is presented in [106] in which a set of fixed quadrature points are selected from the initial set, and (7.4.4) is solved to determine the corresponding weights.

We note that while this general approach may seem appealing, quadrature rules must be calculated and stored on an element by element basis for all elements in the polytopic mesh before calculation of the underlying matrix system can begin. We also note that whilst the optimised quadrature rule is exact for the set of functions in the space $\boldsymbol{\Phi}_\kappa$, the accuracy in terms of integrating general functions is unclear.

7.4.2.3 Integration of Homogeneous Functions

We now move to consider Lasserre's method for integrating homogeneous functions over general polytopes. The method was first introduced in [101] for convex polytopes, and

later extended to general polytopes in [45]. The underlying idea here is to utilise Stokes' Theorem, together with Euler's Homogeneous Function theorem. In particular, consider a polytopic element $\kappa \in \mathcal{T}_h$, and a sufficiently regular function f, defined over κ, we wish to evaluate

$$\int_\kappa f \, \mathrm{d}\mathbf{x}.$$

Assuming that f is a positively homogeneous function of degree q, such that $f(\lambda \mathbf{x}) = \lambda^q f(\mathbf{x})$, for $\lambda > 0$. Then, assuming that f is continuously differentiable, Euler's Homogeneous Function Theorem states that

$$q f(\mathbf{x}) = \mathbf{x} \cdot \nabla f(\mathbf{x}). \tag{7.4.5}$$

Moreover, given any vector-valued function $\mathbf{g}$ that is sufficiently smooth, Stokes' Theorem states that

$$\int_\kappa (\nabla \cdot \mathbf{g}) f \, \mathrm{d}\mathbf{x} = \int_{\partial\kappa} (\mathbf{g} \cdot \mathbf{n}_\kappa) f \, \mathrm{ds} - \int_\kappa \mathbf{g} \cdot \nabla f \, \mathrm{d}\mathbf{x}, \tag{7.4.6}$$

where $\mathbf{n}_\kappa$ denotes the unit outward normal vector to the boundary $\partial\kappa$ of κ. Then, setting $g = \mathbf{x}$ in (7.4.6) and employing (7.4.5), we deduce that

$$\int_\kappa f \, \mathrm{d}\mathbf{x} = \frac{1}{d+q} \sum_{i=1}^{n_F} \int_{F_i} (\mathbf{x} \cdot \mathbf{n}_{F_i}) f \, \mathrm{ds}, \tag{7.4.7}$$

where $\mathbf{n}_{F_i}$ denotes the restriction of the unit outward normal vector $\mathbf{n}_\kappa$ to the facet F_i, $i = 1, ..., n_F$. We note that this process can be repeated to produce a formula which involves integration on lower-dimensional facets. For example, given F_i, for some fixed i, $1 \leq i \leq n_F$, we write

$$\partial F_i = \{F_{ij} = F_i \cap F_j : F_i \cap F_j \neq \emptyset, \, i \neq j\},$$

to denote the set of $(d-2)$-dimensional facets of κ. As an example, F_{ij} is an edge of a polyhedron in $\mathbb{R}^3$ which lies on the boundary of the face F_i. Furthermore, define $\mathbf{n}_{F_{ij}}$ to be the unit normal vector to F_{ij} which lies in the plane F_i. Given a point $\mathbf{x}_i \in F_i$ and a $(d-1)$-dimensional orthonormal basis $\left\{e_j^i\right\}_{j=1}^{d-1}$ on the facet F_i, any $\mathbf{x} \in F_i$ may be written in the form

$$\mathbf{x} = \mathbf{x}_i + \sum_{k=1}^{d-1} \alpha_k e_k^i,$$

for some scalars α_k, $k = 1, ..., d-1$. Then, applying (7.4.6) to a specified facet F_i,

$1 \leq i \leq n_F$, with $g = \mathbf{x} - \mathbf{x}_i$, we deduce

$$\int_{F_i} f \, \mathrm{ds} = \frac{1}{d+q-1} \left(\sum_{F_{ij} \subset \partial F_i} \int_{F_{ij}} \left((\mathbf{x} - \mathbf{x}_i) \cdot \mathbf{n}_{F_{ij}} \right) f \, \mathrm{ds} + \int_{F_i} (\mathbf{x}_i \cdot \nabla f) \, \mathrm{ds} \right). \tag{7.4.8}$$

In $\mathbb{R}^2$, the right-hand side of (7.4.8) only involves evaluations of the integrand at the points which form the underlying face F_i, $1 \leq i \leq n_F$. Therefore, for the integration of polynomial functions, the repeated application of (7.4.8) yields an exact integration rule which only requires the evaluation of f and its partial derivatives at the vertices of $\kappa \in \mathcal{T}_h$. In $\mathbb{R}^3$, we rewrite the integral over F_{ij} to be an integral over its $(d-2)$-dimensional facets (points), and an integral involving the derivative of f over F_{ij}. Thus, for polynomial functions, the recursive application of this formula only requires the evaluation of f and its partial derivatives at the vertices of κ in order to precisely evaluate the integral of f over $\kappa \in \mathcal{T}_h$, which shows that this approach is extremely efficient. However, the disadvantage of this approach is that it can only be applied to PDE problems with slowly varying coefficients, which can be approximated well and quickly by polynomial interpolation.

Following the experimental results disclosed in [38], we exploit the integration of homogeneous functions in order to provide a quadrature rule for our DGFEM on polytopic meshes. This method offers the best balance between performance and numerical accuracy.

7.5 Numerical Experiments

In our first example we present a number of computational experiments highlighting the practical performance of exploiting polytopic meshes for turbulent flows. All polytopic meshes in this section are generated by agglomerating a fine mesh $\mathcal{T}_h^{\text{fine}}$ using the partitioning algorithm provided in the ParMETIS software package.

7.5.1 NACA 0012 Aerofoil

We consider the test case presented in Section 6.6.1, with the aim of achieving comparable numerical results for turbulent flows, while exploiting polytopic meshes to significantly reduce the number of degrees of freedom associated with the problem. As such, consider a NACA 0012 aerofoil with unit chord length $c = 1$ as shown in Figure 7.5.1. The domain $\Omega \subset \mathbb{R}^2$ is defined as the rectangle $[-2.0\ 2.0] \times [-2.0\ 4.0]$, with the aerofoil located along the central axis between $x = 0$.

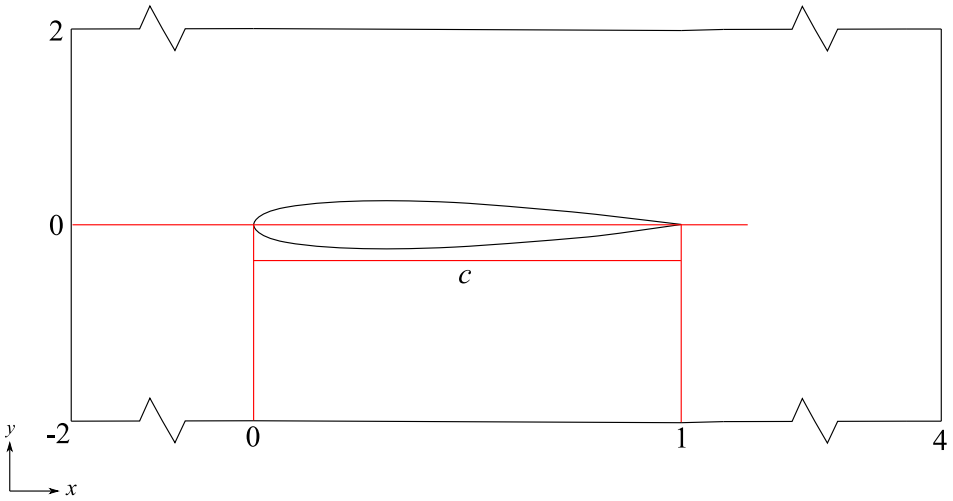

Figure 7.5.1: Schematic of the NACA 0012 aerofoil. Chord length $c = 1$, angle-of-attack $\alpha = 0°$.

We define the inlet boundary located at $x = -2.0$, to be a turbulent free stream with average velocity $u_\infty = 13.909$. The direction of the inlet velocity is adjusted to represent the differing angles of attack. A no-slip boundary condition is used for the aerofoil edges, while periodic boundaries are used for the domain outflows at $y = -2.0$ and $y = 2.0$. The main outflow at $x = 4.0$, is described by the stress-free Neumann condition.

The domain is subdivided into a fine mesh $\mathcal{T}_h^{\text{fine}}$ consisting of isotropic and anisotropic, quadrilateral elements. The quadrilateral mesh is then agglomerated into an initial polygonal mesh $\mathcal{T}_h^{\text{p-initial}}$ consisting of $1,000$ elements. This is shown in Figure 7.5.2. The 'p-' notation is used to signify a polygonal mesh.

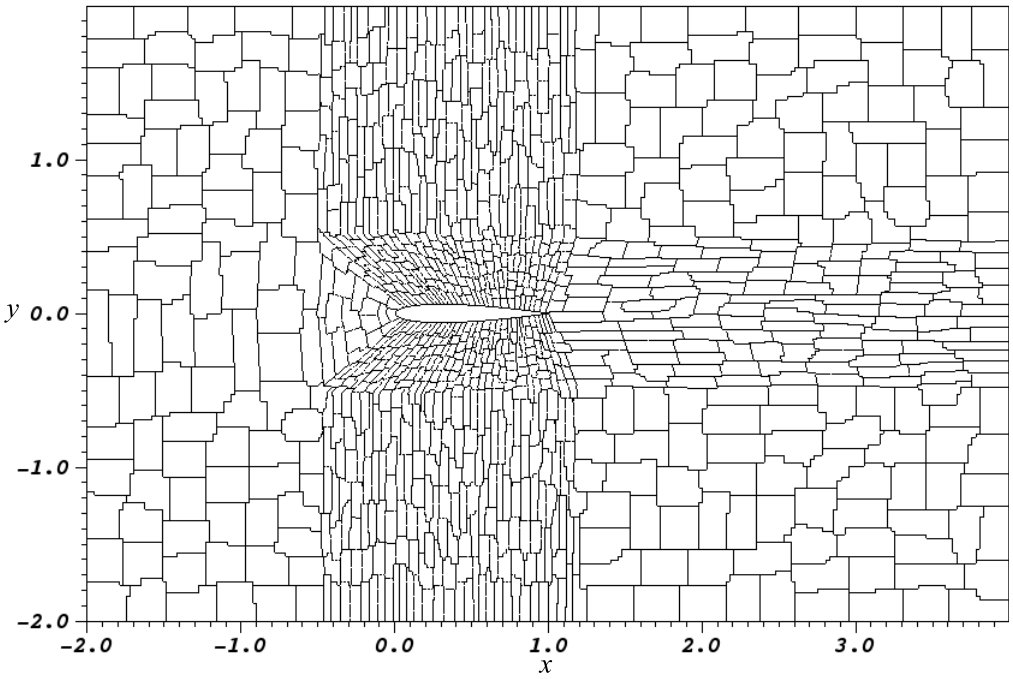

Figure 7.5.2: Initial mesh $\mathcal{T}_h^{\text{p-initial}}$, consisting of $1,000$ general polygonal elements.

In order to achieve the target Reynolds number of $Re = 23,000$, we apply the hybrid continuation algorithm outlined in Section 6.7. To make the algorithm suitable for polytopic meshes, we adjust the Reynolds number values at which we refine the mesh. The polytopic mesh allows us to mesh the same geometry with fewer elements, compared to using meshes consisting of standard elements. However, the numerical test case in Section 6.6.1 suggests that we require a far higher mesh density at the aerofoil edges compared to the rest of the domain, to account for the behaviour of the variable $\tilde{\omega}$. As such, we carry out the h-refinement in a slightly different fashion, refining/coarsening the mesh each time the Reynolds number is increased by 800 if $Re < 4,000$, and by $2,500$ if $Re > 4,000$. The coarseness of the initial mesh $\mathcal{T}_h^{\text{p-initial}}$, means that it is unlikely that we will observe areas of over refinement in the final continuation mesh, allowing for additional h-refinement passes before the flow has become fully turbulent. This will ensure that the mesh has suitable density close to the aerofoil edges to correctly represent the turbulent boundary layer. We continue the DWR approach according to the target functional (6.5.1), refining the 20% of elements with the greatest approximate absolute error, whilst coarsening the 10% of elements with the smallest error. An alternative approach would be to refine the mesh each time the Reynolds number is increased, ensuring that for a given tolerance TOL, $\sum_{\kappa \in \mathcal{T}_h} |\hat{\eta}_\kappa| \leq \text{TOL}$. However, this is unsuitable for industrial applications due to the increased computational demand compared to the approach proposed above.

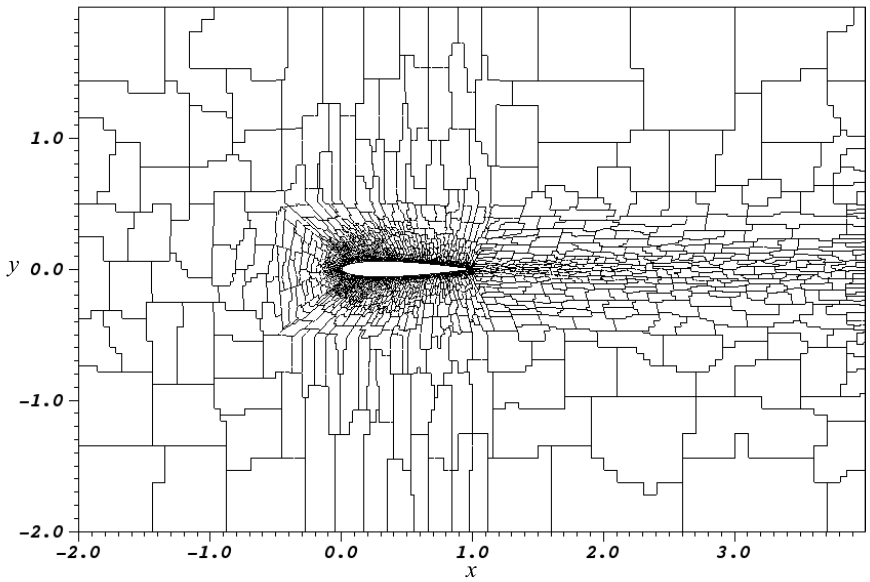

Figure 7.5.3: Final polygonal continuation mesh $\mathcal{T}_h^{\text{p-refined},0}$ after 10 passes of h-refinement. $2,112$ elements.

The final polygonal continuation mesh is shown in Figure 7.5.3. Colour plots of the non-dimensionalised velocity solutions for $Re = 23,000$ are shown in Figures 7.5.4 and 7.5.5.

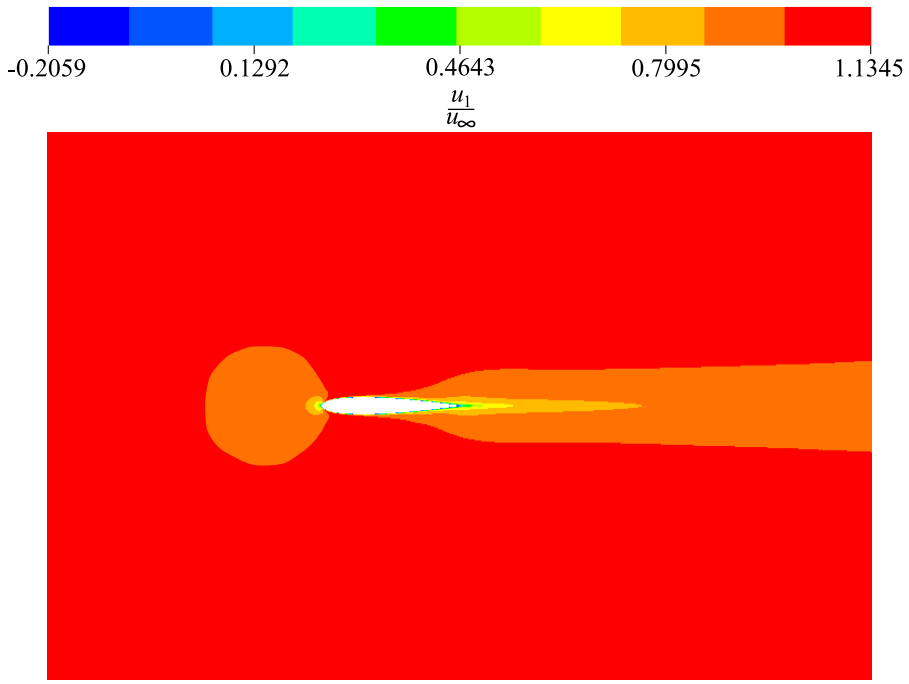

Figure 7.5.4: Colour plot of the scaled axial velocity $\frac{u_1}{u_\infty}$, computed on $\mathcal{T}_h^{\text{p-refined},0}$. $Re = 23,000$.

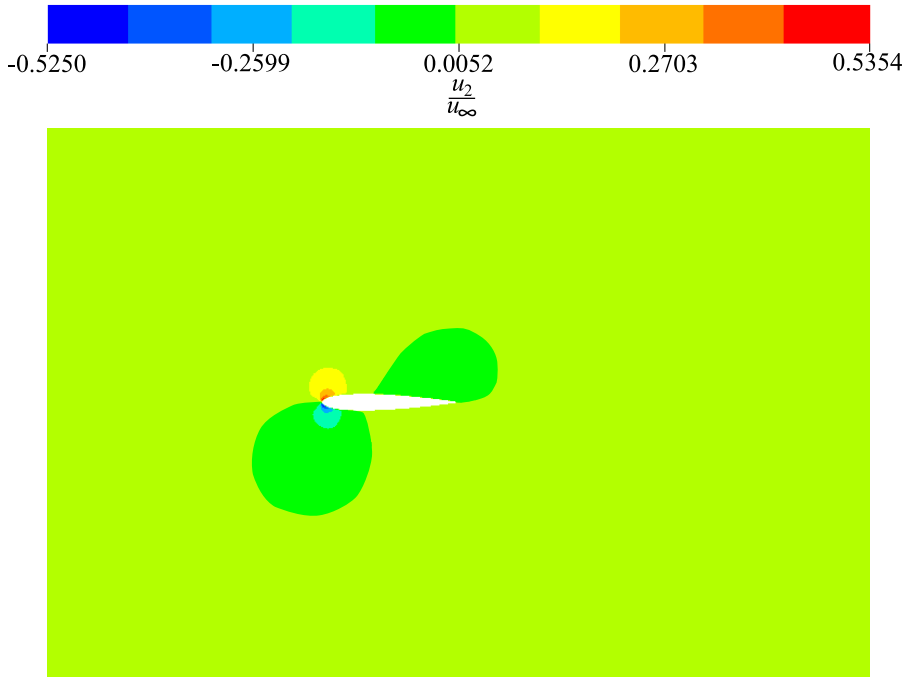

Figure 7.5.5: Colour plot of the scaled vertical velocity $\frac{u_2}{u_\infty}$, computed on $\mathcal{T}_h^{\text{p-refined},0}$. $Re = 23,000$.

Figure7.5.5 has a degree of asymmetry in the numerical solution for the vertical velocity u_2. This is highlighted by the scale of the colour plot, but is in reality much more subtle. This may be due, in part, to the asymmetric initial polygonal mesh, meaning that careful consideration should be taken when designing future polygonal meshes, mindful of the symmetries of the geometry that is being represented. As such, the simple agglomeration of standard elements may be unsuitable for turbulent incompressible flows through symmetric domains, and a tailored initial polygonal mesh preserving the geometry's symmetry may be better suited.

In spite of this, using polygonal meshes produces remarkably precise numerical solutions, while allowing a significant reduction in the degrees of freedom of the mesh. Additionally, we are able to refine the mesh further using the refinement strategy presented in Section 6.3, while maintaining fewer degrees of freedom than are required for the standard meshes presented in Section 6.6.1. These meshes are presented below in Figures 7.5.6, 7.5.7 and 7.5.8, with $217,500$, $240,000$ and $330,000$ degrees of freedom, respectively.

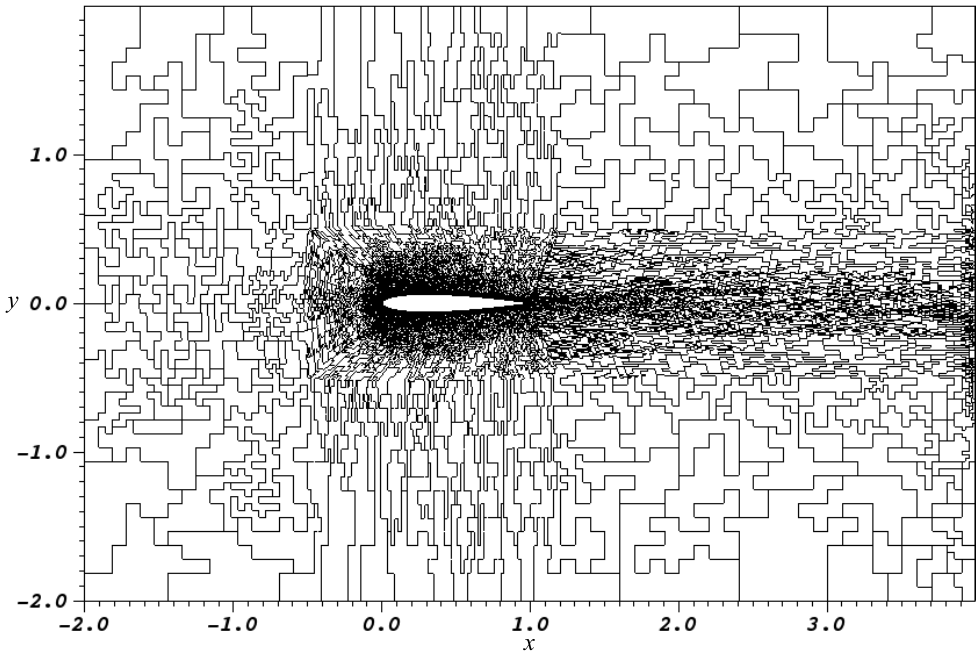

Figure 7.5.6: Polygonal mesh with $217,500$ degrees of freedom.

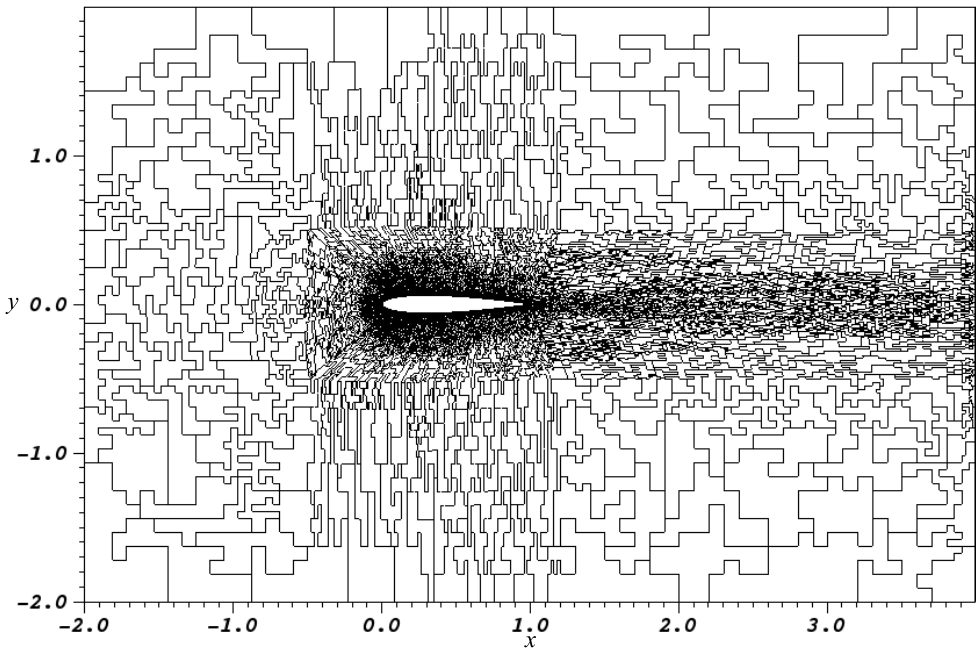

Figure 7.5.7: Polygonal mesh with $240,000$ degrees of freedom.

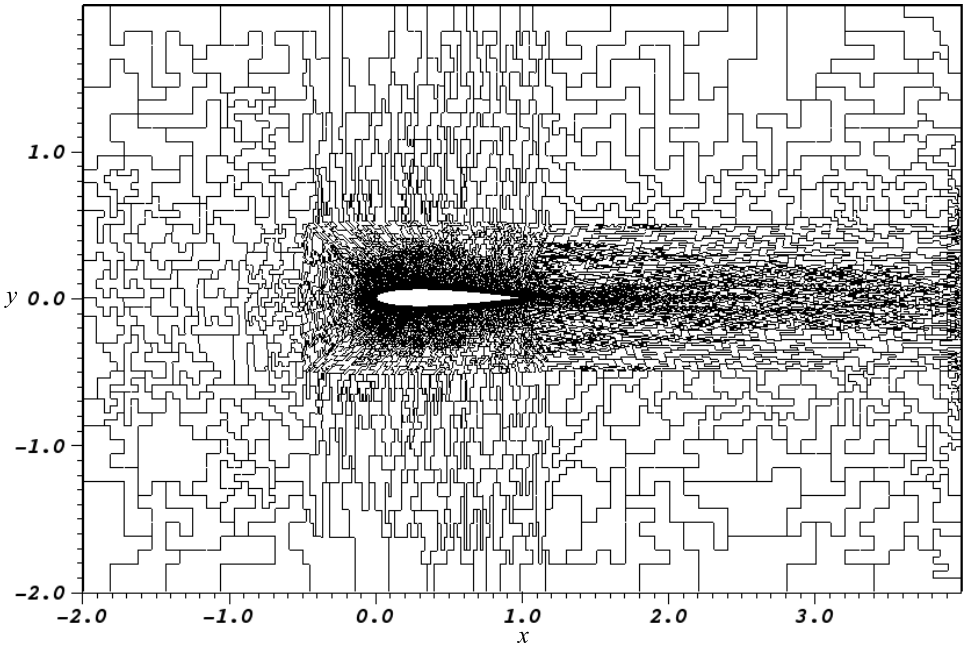

Figure 7.5.8: Polygonal mesh with $330,000$ degrees of freedom.

We use the approximation to the target functional $J(\mathbf{u}_{h,\text{fine}}) = 1553.564681$, proposed in Section 6.6.1, to compute the approximate absolute and relative error values for the polygonal DGFEM. These are summarised in Table 7.1, along with comparisons against results computed on meshes consisting of standard elements generated from $\mathcal{T}_h^{\text{initial}}$, see Figure 6.6.3, using the hybrid continuation algorithm in Section 6.7.

Degrees of Freedom (Polygonal Mesh)	Degrees of Freedom (Standard Mesh)	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$ (Polygonal Mesh)	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$ (Standard Mesh)	$\frac{\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert}{\lvert J(\mathbf{u}_{h,\text{fine}})\rvert}$ (Polygonal Mesh)
$147,930$	$958,220$	2.716821	2.516973	1.748766×10^{-3}
$217,500$	$1,096,610$	1.096843	1.138894	7.060170×10^{-4}
$240,000$	$1,202,490$	0.956189	0.936661	6.154807×10^{-4}
$330,000$	$1,385,140$	0.533870	0.554820	3.436420×10^{-4}

Table 7.1: The approximate absolute and relative error values with respect to the target functional (6.5.1).

Polygonal meshes produce remarkably precise numerical solutions, while allowing a significant reduction in the number of degrees of freedom of the problem. In particular, solutions with accuracy sufficient for practical/engineering applications can be obtained

from polygonal meshes with a relatively small number of degrees of freedom [38]. This is evident in Table 7.1, which clearly demonstrates a significant reduction in computer memory requirements using a polygonal mesh, compared to one consisting of standard shaped elements alone. Further, we are able to maintain a similar degree of computational accuracy with regards to the proposed target functional (6.5.1), mirroring the error values of the standard mesh solutions. This suggests that polygonal meshes are suitable for use on problems involving turbulent incompressible flows, and that they respond sufficiently well to DWR h-refinement. This is notable from an industrial perspective, as we have demonstrated that we are able to reduce significantly, the memory requirements of fluid computations, potentially allowing larger and more complicated problems to be tackled without additional HPC hardware upgrades. We further explore this concept in Section 7.5.2.

The adjustments made to the hybrid continuation algorithm for use with polygonal meshes, have been successful. The additional refinement passes before the flow becomes fully turbulent, are useful in ensuring a sufficiently high mesh density close to the aerofoil edges, enabling accurate represent of the numerical solution. To highlight this, we provide a close up of the polygonal mesh with $330,000$ degrees of freedom in Figure 7.5.9.

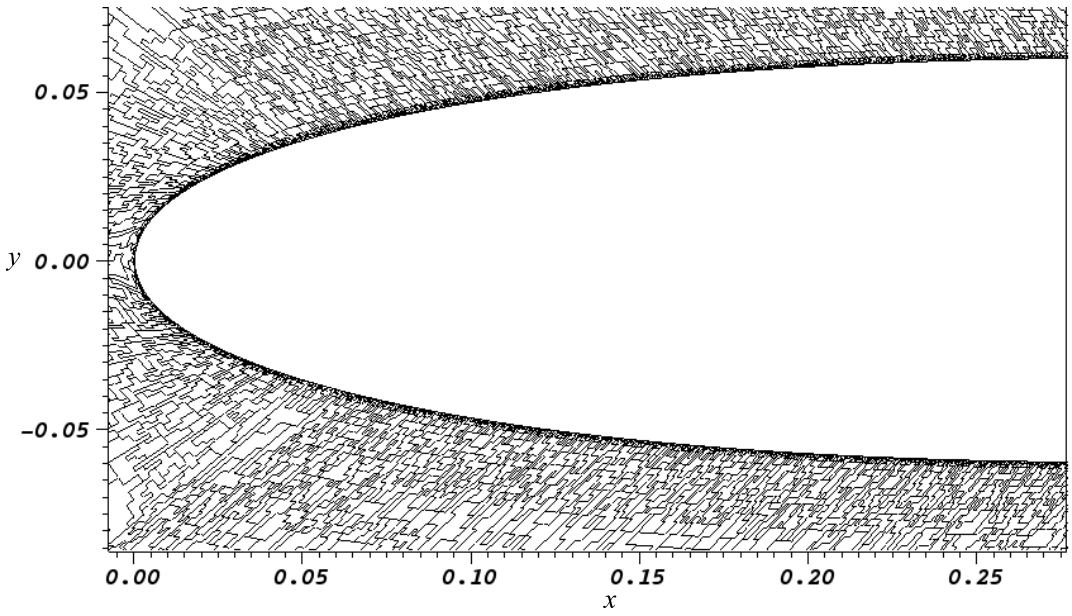

Figure 7.5.9: Close-up of polygonal mesh with $330,000$ degrees of freedom.

We now consider an industrially relevant numerical test case in Section 7.5.2, with the aim of validating the DGFEM for polygonal meshes at high Reynolds numbers.

7.5.2 Turbine Stator Cascade

In this section we present an industrially relevant test case, with the goal of reducing the computational load required to achieve a fully turbulent flow with $Re = 6.8 \times 10^5$. As such, we consider the 1.5 Stage Aachen turbine stator cascade [138] in Figure 7.5.10.

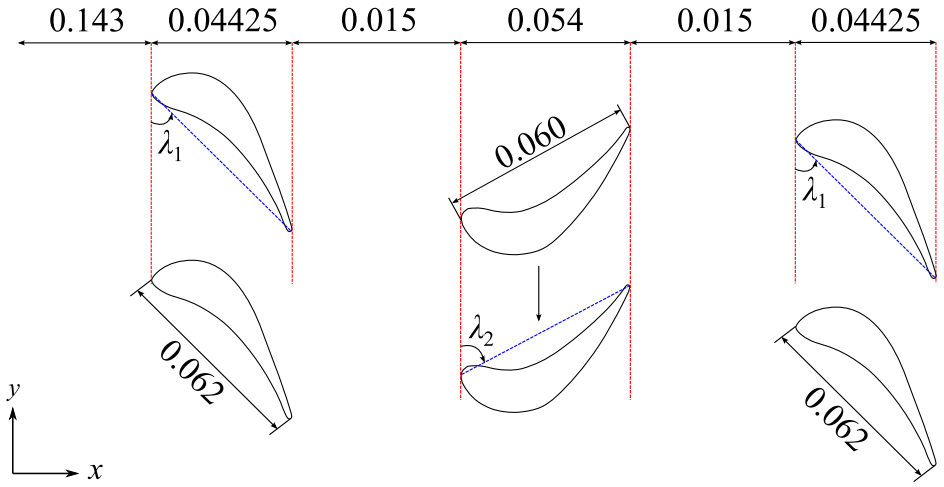

Figure 7.5.10: Schematic of the Aachen Turbine on the cascade plane [138]. All lengths are stated in meters. The angles subtend by the y-axis and the stator and rotor chords are $\lambda_1 = 45.5°$ and $\lambda_2 = 62°$ respectively.

In particular, we consider the mid-height flow between the initial stator blades, with a pitch $s_s = 0.0476\,\text{m}$. The initial stator section is designed to receive an inlet flow parallel to the horizontal axis, such that the flow is axial. The design Reynolds number, based on the chord length and exit velocity is $Re = 6.8 \times 10^5$. This is shown in Figure 7.5.11.

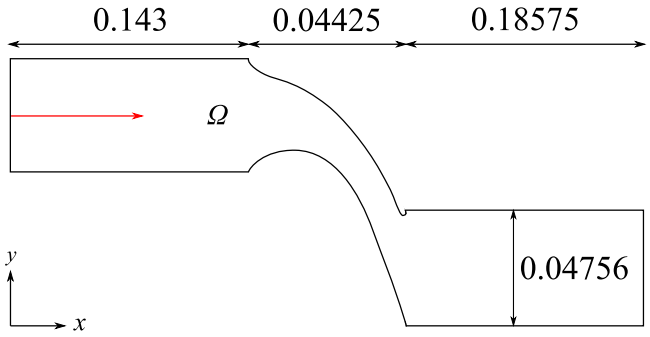

Figure 7.5.11: Stator cascade mesh geometry. All lengths are stated in metres.

The domain Ω is subdivided using standard, non-overlapping, isotropic and anisotropic elements forming $\mathcal{T}_h^{\text{fine}}$. These elements are then agglomerated using the partitioning algorithm provided in the ParMETIS software package to form a new mesh $\mathcal{T}_h^{\text{p-initial}}$, consisting of 300 polygonal elements. In the same fashion as Section 7.5.1, additional mesh refinement passes are carried out before the flow exceeds the Reynolds number indicating that it is fully turbulent. This ensures that there is sufficient mesh density close to the blade edges in order to capture the solutions of k and ω correctly, ensuring that the nonlinear solver continues to converge as the Reynolds number increases. Therefore, h-refinement passes are carried out each time the Reynolds number is increased by 800 if $Re < 4,000$, $2,500$ if $Re < 5,000$, 20000 if $Re < 100,000$, and $100,000$ if $Re > 100,000$. The change in refinement values is made to reflect the high target Reynolds number of the flow, ensuring that computational accuracy is balanced against hardware memory requirements. At each refinement pass, we refine the 10% of elements with the largest approximate absolute error values, and the 5% of elements with the smallest error values.

Figures 7.5.12 and 7.5.13, show the final continuation mesh $\mathcal{T}_h^{\text{p-refined}}$, consisting of $4,482$ polygonal elements.

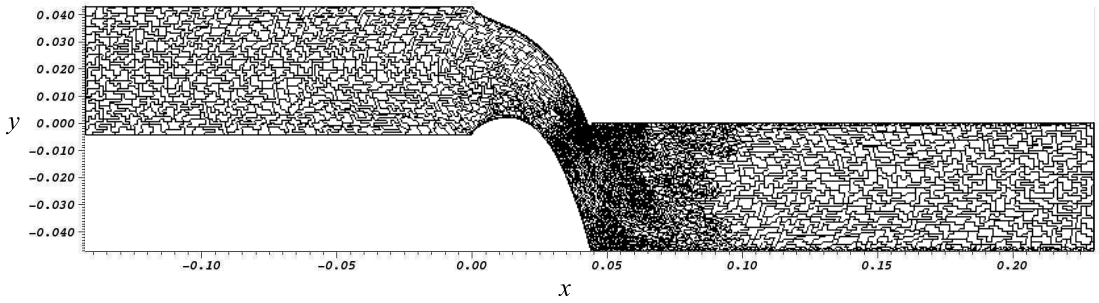

Figure 7.5.12: Final mesh $\mathcal{T}_h^{\text{p-refined}}$ consisting of $4,482$ polygonal elements.

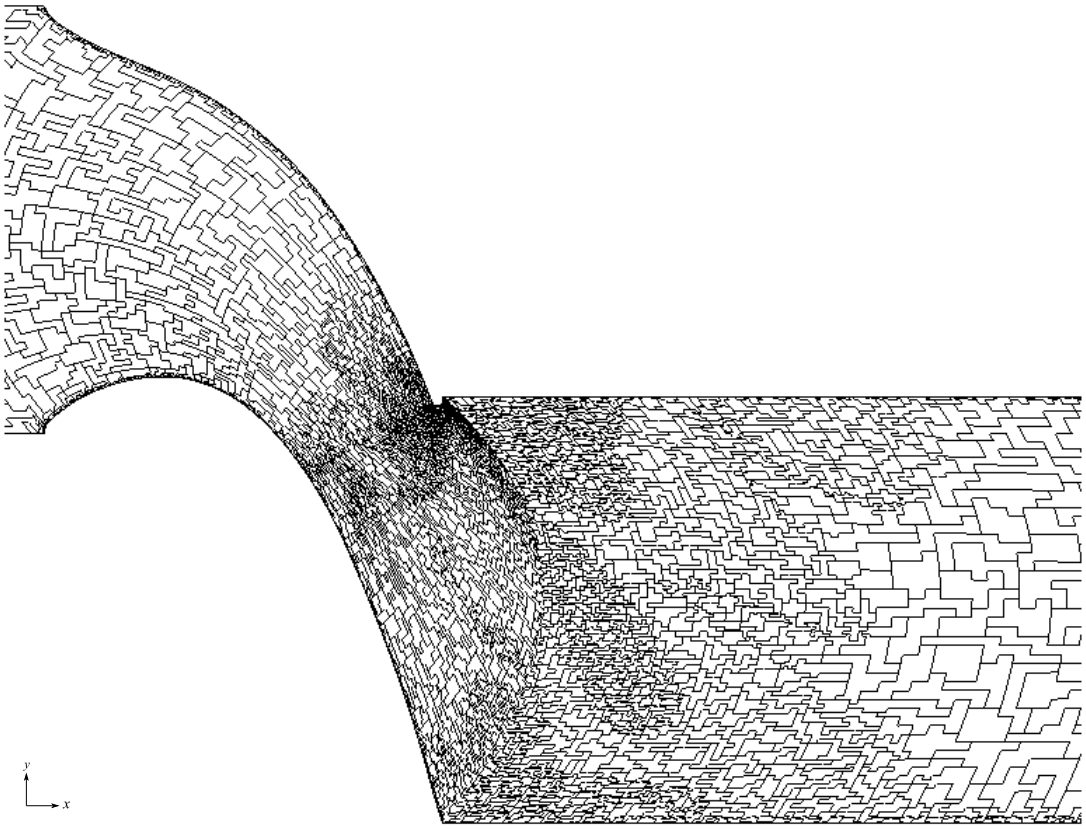

Figure 7.5.13: Close-up of the outflow of $\mathcal{T}_h^{\text{p-refined}}$.

The purpose of this numerical experiment is to demonstrate the computational efficiency of employing an agglomerated polygonal mesh over one consisting solely of standard shaped elements. We are able to achieve more accurate numerical results to those in Section 5.4.1, but with far fewer computational resources, measured in the form of degrees of freedom. In Figures 7.5.14 and 7.5.15, we present the velocity solutions with units m/s, and the static pressure p in Figure 7.5.16 with units pa.

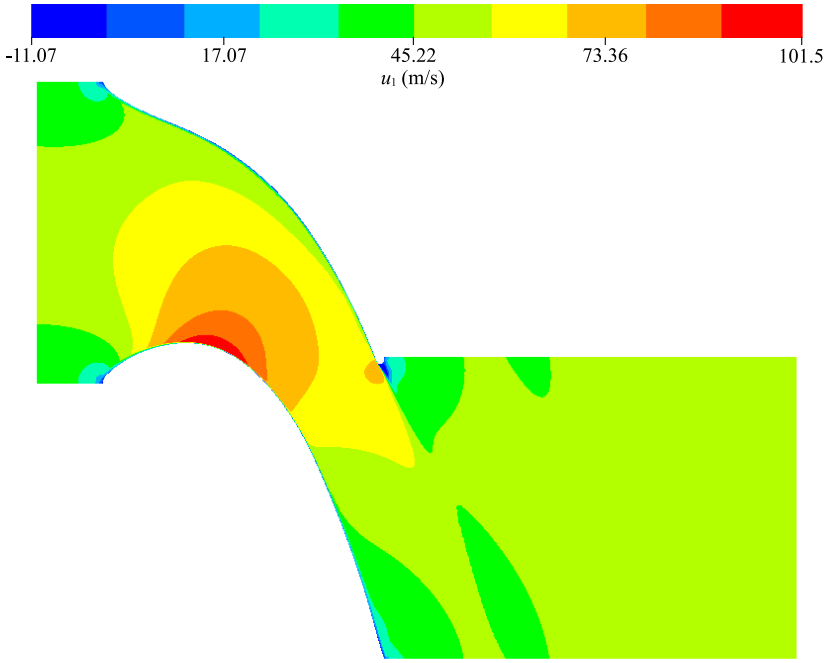

Figure 7.5.14: Colour plot of the axial velocity u_1 (m/s), computed on $\mathcal{T}_h^{\text{p-refined}}$. $Re = 680,000$.

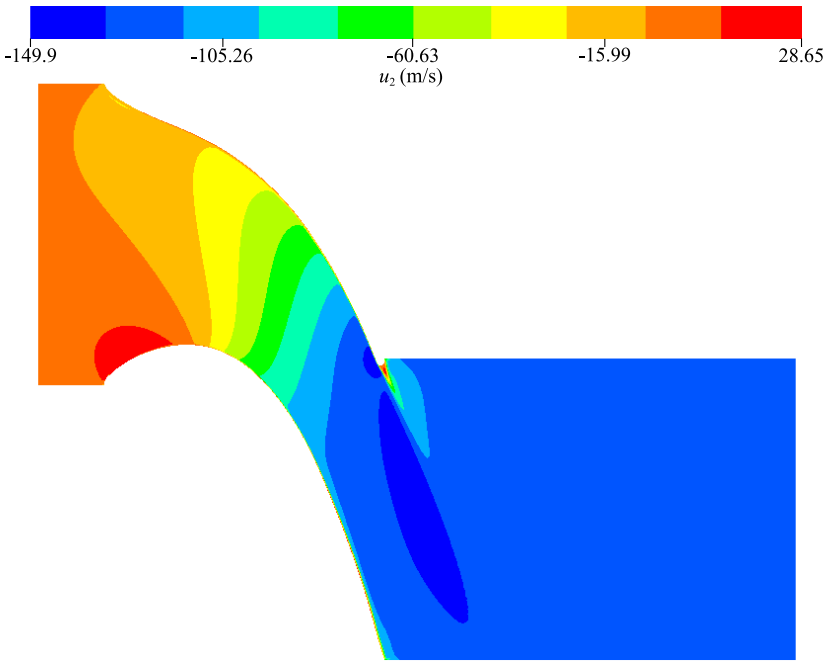

Figure 7.5.15: Colour plot of the vertical velocity u_2 (m/s), computed on $\mathcal{T}_h^{\text{p-refined}}$. $Re = 680,000$.

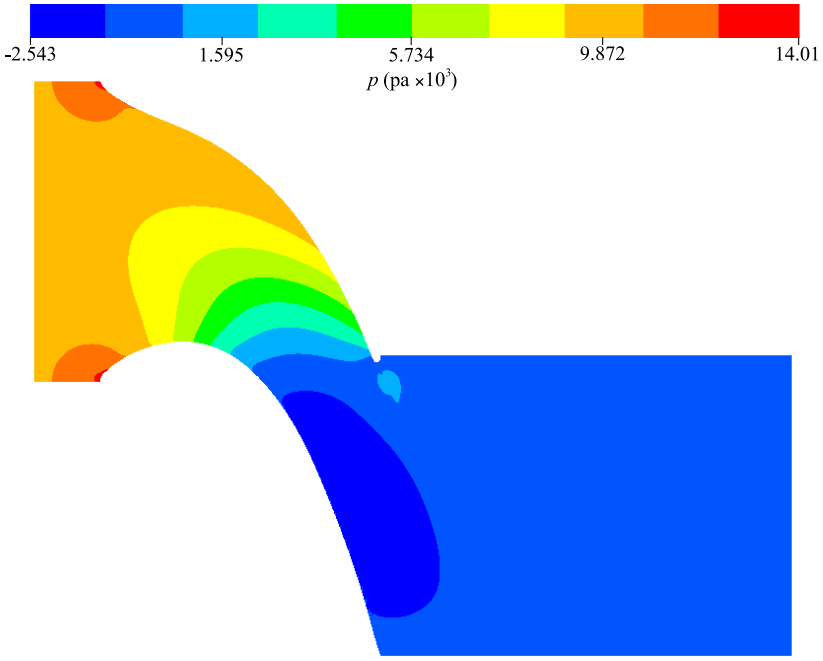

Figure 7.5.16: Colour plot of the static pressure p (pa), computed on $\mathcal{T}_h^{\text{p-refined}}$. $Re = 680,000$.

Due to the reduced computational demand of polygonal meshes, we are able to consider further h-refinement passes of the final solution shown in Figures 7.5.14,7.5.15 and 7.5.16. In particular, we carry out a single h-refinement pass, refining the 10% of elements with the largest numerical error values, creating the mesh $\mathcal{T}_h^{\text{p-refined},a}$ with $161,340$ degrees of freedom. This process is repeated on the new mesh $\mathcal{T}_h^{\text{p-refined},a}$, creating $\mathcal{T}_h^{\text{p-refined},b}$, consisting of $193,620$ degrees of freedom. A close-up of $\mathcal{T}_h^{\text{p-refined},b}$ is shown in Figure 7.5.17.

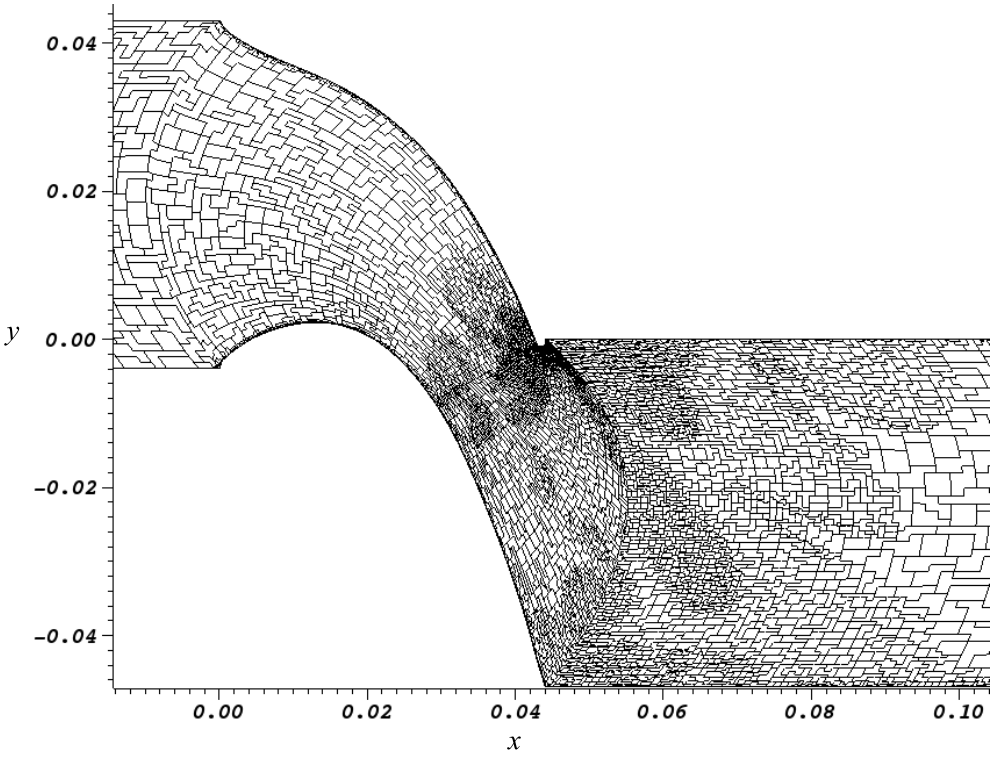

Figure 7.5.17: Polygonal mesh $\mathcal{T}_h^{\text{p-refined},b}$, with $193,620$ degrees of freedom.

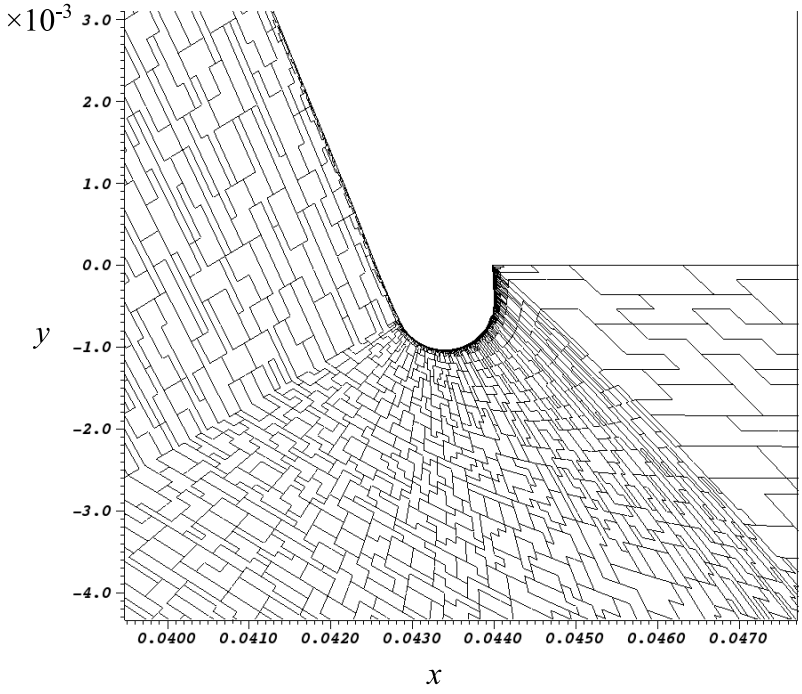

Figure 7.5.18: Close up of polygonal mesh $\mathcal{T}_h^{\text{p-refined},b}$. $193,620$ degrees of freedom.

To facilitate the comparison between standard and polygonal meshes for turbomachinery flows, we approximate the target functional $J(\mathbf{u})$ as in Section 6.6.2, with $J(\mathbf{u}_{h,\text{fine}}) = 224.218977$. Table 7.2 provides a summary of the absolute and relative error values with respect to the target functional, with solutions computed on both, standard and polygonal meshes.

Mesh Type	Degrees of Freedom	$\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert$	$\frac{\sum_{\kappa\in\mathcal{T}_h} \lvert\hat{\eta}_\kappa\rvert}{\lvert J(\mathbf{u}_{h,\text{fine}})\rvert}$
Standard Elements, Figure 5.4.3.	$1,590,000$	2.651334	1.182475×10^{-2}
Standard Elements with h-refinement, Figure 6.6.24.	$1,046,310$	1.523670	6.795455×10^{-3}
Polygonal Elements with h-refinement, $\mathcal{T}_h^{\text{p-refined}}$, Figure 7.5.12.	$134,460$	1.814689	8.0893378×10^{-3}
Polygonal Elements with h-refinement, $\mathcal{T}_h^{\text{p-refined},a}$.	$161,340$	1.294611	5.773869×10^{-3}
Polygonal Elements with h-refinement, $\mathcal{T}_h^{\text{p-refined},b}$, Figure 7.5.17.	$193,620$	1.036973	4.624823×10^{-3}

Table 7.2: Comparison of the approximate absolute error values compared to the functional of interest for standard and polygonal meshes.

Table 7.2 demonstrates the advantages of using polygonal meshes in combination with the hybrid continuation algorithm, compared to meshes consisting of standard elements alone. In particular, Table 7.2 shows a reduction in numerical error beyond that which is achievable on standard meshes with comparable computational hardware. The hybrid continuation algorithm, initially produced a polygonal mesh, see Figures 7.5.12 and 7.5.13, with a larger numerical error compared to the isotropic continuation mesh in Figure 6.6.24. However, the reduced memory requirements associated with polygonal meshes, allows further refinement of the solution, reducing the approximate absolute error by 32%, compared to the solution on the standard mesh in Figure 6.6.24. Additionally, the polygonal solution computed on $\mathcal{T}_h^{\text{p-refined},b}$, only requires around 20% of the computational resources compared to the solution on the standard mesh in Figure 6.6.24.

7.6 Concluding Remarks

Chapter 7 has been concerned with the development of a DGFEM suitable for turbomachinery flow problems on polygonal meshes. In particular, we investigated whether the hybrid continuation algorithm outlined in Section 6.7, could be used effectively with polygonal meshes. In Section 7.5.1, we adjusted the original algorithm, allowing additional refinement passes while the Reynolds number of the flow was below the turbulent transition value. This is to compensate for the coarse initial polygonal mesh not having enough elements close to the aerofoil/blade edges to accurately represent the solution of $\tilde{\omega}$, which increases rapidly close to no-slip boundaries. Additionally, as the flow accelerates, the turbulent boundary layer elongates and narrows along the aerofoil edge, quickly requiring a mesh resolution finer than that of the initial mesh in Figure 7.5.2. This approach worked well, effectively controlling the numerical error while significantly reducing the memory requirements of the problem. This suggested that this approach would be effective for high Reynolds number turbomachinery problems, such as those in Section 6.6.2, where the high density of the initial mesh, limited the degree to which the numerical solution could be refined.

The 1.5 Stage Aachen turbine stator cascade present in Section 7.5.2, is representative of industrial turbomachinery problem. We seeked to apply the ideas developed in Section 7.5.1, to overcome the difficulties experienced in terms of mesh design/refinement in Section 6.6.2, by allowing coarsening of the initial mesh by way of element agglomeration. Table 7.2 shows that this approach was reasonably successful, reducing the number of degrees of freedom of the final continuation solve from $1,046,310$ using standard elements, Figure 6.6.24, to $134,460$ using polygonal elements, Figure 7.5.12. The approximate absolute error value is larger on the polygonal continuation mesh, but with significantly fewer degrees of freedom. This allowed for further h-refinement passes once the target Reynolds number had been reached, which was able to produce meshes, $\mathcal{T}_h^{\text{p-refined},a}$ and $\mathcal{T}_h^{\text{p-refined},b}$, which controlled the numerical error effectively.

The combination of the continuation and DWR refinement algorithms, along with the use of polygonal computational meshes, has been successful in reducing the memory requirements of turbulent incompressible flow problems, whilst maintaining the level of numerical accuracy required by industry. In particular, the polygonal mesh elements are more effective at reducing the mesh density in areas of laminar flow, as shown in Figures 6.6.14 and 7.5.3. As such, it is the recommendation of this work, that polygonal mesh support should be added to any large scale industrial implementation of the proposed DGFEM method.

Chapter 8

Conclusions and Outlook

In this work, we developed an interior penalty Discontinuous Galerkin Finite Element Method for the discretisation of the turbulent, incompressible Navier-Stokes equations, with a novel approach that maintains stability on meshes consisting of curvilinear elements. Through careful analysis of the numerical algorithms, we have developed a methodology that incorporates adaptive refinement, to capture with a high degree of accuracy, high Reynolds number flows. In Chapter 7, we presented some of the most recent DGFEM results, employing an interior penalty method that is stable on general computational meshes consisting of polytopic elements. In particular, we extended the existing results presented in [38], further refining the continuation algorithm introduced in Chapter 6, to resolve high Reynolds number, turbulent, incompressible flows on general polytopic meshes.

We considered the interior penalty DGFEM for PDEs with non-negative characteristic form in Chapter 2, in particular, the steady-state advection-diffusion-reaction equation. This served as an introduction to DGFEMs, allowing for the derivation and analysis of the required function spaces and trace operators. A discussion concerning the stability of the method, along with the derivation of the discontinuity-penalisation function then followed. Chapter 2 provided a summary of the technical ideas and concepts that are used regularly within the literature, with the established understanding required for the discretisation of the turbulence model in Chapter 4.

In Chapter 3, we discussed the DGFEM discretisation of the incompressible Navier-Stokes equations, presenting the required function spaces and discussed the stability of the nonlinear forms. The notion of coercivity was replaced with the inf-sup stability of Babuska [17], Brezzi [35], Girault and Raviart [72], to accommodate the indefiniteness of the resulting discrete nonlinear system (3.2.3). We then considered the implementation of the numerical method, discussing preconditioning techniques designed to reduce the computational cost of the numerical scheme.

The initial numerical results presented in Section 3.5.1, 3.5.2, 3.5.3 and 3.5.4, demonstrate the suitability of the numerical solver for simplistic flow problems. Whilst these are far from the complex turbulent flows resolved in Chapters 5, 6 and 7, they did allow us to resolve some potential concerns with the solver. Firstly, when defining the boundary conditions of the domain it is not immediately clear how to resolve the point where two different boundaries join, such as the corners of the channel in Figure 3.5.1. The discretisation (3.2.2) is an integral form, and as such, may omit finitely mainly points from the computation. However, whether this property would naturally translate into the computer program written to perform the calculations, or would require additional coding was unclear. Fortunately, this issue was resolved without difficulty because of the way the quadrature rules are coded in the AptoFEM package.

Secondly, the choice of numerical flux was a further concern, since it affects significantly, the behaviour of the simulated flow. The literature suggests that the Lax-Friedrichs flux [46] is suitable for incompressible flows, which agrees with the laminar numerical results presented in Chapter 3. However, from these results, it is unclear whether the choice of flux is suitable for turbulent flows, and that further numerical experiments would be needed to determine this. In general, the numerical results presented in Chapter 3 are agreeable with real world test cases, indicating that the solver is producing industrially useful laminar results, even if the Reynolds numbers are rather low for industrial turbomachinery flows.

In Chapter 4, we extended the results of Chapter 3, introducing turbulence models for incompressible flows, with the aim of producing industrially relevant numerical results. To this end, we created a novel DGFEM discretisation of the incompressible Navier-Stokes equations, based on the interior penalty method, which includes the $k-\omega$ turbulence model. To the authors knowledge, this is the first such DGFEM for the incompressible Navier-Stokes equations with turbulence modelling. Further, we developed a continuation type algorithm to increase the Reynolds number of the simulated flow in order to produce industrially relevant results. The numerical experiments presented in Section 4.4 demonstrated the numerical scheme's ability to handle turbulent flows and produce accurate results. These simplistic initial test cases were necessary to develop the method, as it was unclear whether any of the existing approaches in the literature for compressible flows [25, 80, 82, 83], would be applicable turbulent incompressible flows.

The continuation algorithm proved most effective when the initial solve was undertaken for low Reynolds number, laminar flows. This provided a sufficiently close initial approximation for the nonlinear Newton solver, such that the turbulence model could be incorporated into the calculations. For the simple geometries seen in Section 4.4, this approach was very successful in accelerating flows through the region of Reynolds numbers associated with the laminar to turbulent transition. However, the approach of a fixed percentage increase

in Reynolds number for each successful flow simulation appears to be too strict. Instead, a methodology of relatively small increases in Reynolds number when approaching and passing through the "laminar to turbulent flow transition" should be implemented, since these are the values for which the calculated flow changes most significantly between successive solves. Then, larger increases in Reynolds number may be used once the turbulent flow is fully developed. As such, this algorithm was revisited and revised in Chapters 6 and 7 to implement these findings.

Through the careful analysis of inverse estimates, we were able to extend interior penalty DGFEMs to computational meshes consisting of anisotropic curvilinear elements in Chapter 5. This novel extension of the DGFEM allowed us to handle the industrially relevant Aachen turbine cascade test case in Section 5.4. The proposed new method for estimating the value of the discontinuity penalisation parameter is a refinement of the approach in [67, 77]. The only restriction we place on the element shape, is that each of its edges must be able to be described using only polynomials, allowing for maximum and minimum bounds to be placed on the Jacobian of the element transformation. In practice, this allows for elements to adhere exactly to blade and structure edges, with these often defined, for manufacturing purposes, with piecewise quadratic curves. Further, the relatively low, extra computational cost of the proposed implementation, compared to the standard interior penalty DGFEM, makes it ideal for use on large scale industrial flow problems.

In Chapter 6, we developed an *a posteriori* error indicator based on the dual-weighted-residual approach, in order to drive a mesh refinement algorithm. In particular, we followed Hartmann [80], applying approximations to the dual solution in order to derive a weighted error bound. After careful analysis of the errors of these approximations, we presented a suitable target functional based upon the wake pressure deficit [114]. The numerical experiments in Section 6.6 demonstrate the viability of the DWR approach for use on high Reynolds number, turbulent incompressible flows. Further, we improved the continuation algorithm proposed in Chapter 4, incorporating the findings presented there, as well as mesh refinement/coarsening to improve the overall accuracy of the numerical solution. An alternative approach would be to employ the continuation algorithm as it is proposed in Chapter 4, then refine the solution once the desired Reynolds number has been met. However, particularly in the case of high Reynolds number flows, the system of equations are extremely stiff, limiting refinement to very small changes in mesh density before a further calculation of the numerical solution is required. This affects the overall efficiency of the DGFEM solver, increasing the number of computations required to simulate flows for a given Reynolds number and target error value. As such, for industrial applications, we recommend carrying out mesh refinement passes during the continuation process in order to balance the reduction in numerical error with the increase in Reynolds number. On the other hand, as noted in Chapter 4, the flow changes most significantly when the

Reynolds number passes through the "laminar to turbulent flow transition," so the level of mesh refinement should considered carefully if the flow is not yet fully turbulent. This issue was further addressed in Chapter 7, following the introduction of polytopic meshes.

Finally, in Chapter 7, we extended our work and the knowledge in the literature to consider an interior penalty DGFEM suitable for high Reynolds number, turbulent, incompressible flows on computational meshes consisting of general polytopic elements. We introduced the construction of such meshes in both two and three dimensions, demonstrating DGFEMs flexibility to naturally handle hanging nodes. We then derived a suitable discontinuity-penalisation function to stabilise the method, through the use of inverse estimates on standard triangular and quadrilateral elements. Further to this, we discussed some approximations required for the implementation of this method, namely the choice of basis functions and the quadrature rule. We presented a number of alternative methods which are popular in the literature, before deciding upon a Cartesian bounding box approach for the basis functions, and the integration of homogeneous functions for the quadrature rule.

In Section 7.5 we presented a number of numerical experiments that extended the a posteriori error indicator results of Chapter 6 to general computational meshes consisting of polytopic elements. We made further improvements to the refinement and continuation algorithms, focusing the continuation steps close to the Reynolds numbers associated with the laminar to turbulent transition of the flow, and reducing the number of mesh refinement passes until the flow is fully turbulent. This approach has proven to be most effective, reducing the amount of unnecessary mesh refinement associated with refining laminar or transient flows before the Reynolds number could be increased to make the flow turbulent.

The use of polytopic elements reduced significantly the number of degrees of freedom required to represent the numerical solution for a given Reynolds number, as shown in Sections 7.5.1 and 7.5.2. This is most clear when we compare the meshes in Figure 7.5.2 with Figure 6.6.2. The mesh agglomeration in Figure 7.5.2 allows for far larger mesh elements to develop in areas of almost laminar flow in fewer refinement passes. Additionally, Table 7.2 shows that in using these more efficient polytopic mesh elements, we are not sacrificing the accuracy of the numerical solution.

8.1 Contributions of this Work

The contributions of the work are summarised below.

An Interior Penalty DGFEM for the Incompressible Navier-Stokes Equations with the $k-\omega$ Turbulence Model

We have discussed a number of developments and advantages of using an interior penalty DGFEM for the types of high Reynolds number flows common in industrial turbomachinery applications. The reduced stencil compared to other DGFEMs, along with the efficient manner with which we are able to incorporate isotropic, anisotropic and curvilinear elements, means that we are able to consider very general geometries without sacrificing computational efficiency.

A DGFEM for High Reynolds Number, Turbulent, Incompressible Flows on Polytopic Meshes

This work has demonstrated the effectiveness of polytopic meshes for turbomachinery flow problems. The ability to agglomerate mesh elements in near laminar areas of the domain more effectively than with standard elements, greatly reduces the number of degrees of freedom required to achieve accuracy sufficient for engineering applications.

A Continuation Type Algorithm to Find Numerical Solutions for High Reynolds Number, Turbulent, Incompressible Flows

1. Choose a target Reynolds number and decide upon the inlet conditions for the final simulated flow. Generate the initial mesh of the geometry.

2. Solve on the initial mesh with the inlet conditions chosen above, a low Reynolds number ($50 \leq Re \leq 300$) laminar flow, without calculating the turbulence model variables k and $\tilde{\omega}$. The initial estimate of the numerical solution required for the nonlinear solver should be 0 for all variables.

3. Solve on the initial mesh, with the same inlet conditions and Reynolds number, a turbulent flow using the previous laminar solve as the initial guess for the nonlinear solver. The boundary conditions for the turbulent variable $\tilde{\omega}$ should be chosen according to the channel flow results for the appropriate Reynolds number.

4. Find the Reynolds number for which the flow transitions to be fully turbulent for the chosen geometry. Select a suitable target functional, as well as, a mesh refinement percentage, $U\%$, and a coarsening percentage, $L\%$.

5. Increase the Reynolds number of the flow, adjust the boundary conditions for $\tilde{\omega}$ and solve on the previous mesh. The value by which you are able to increase the Reynolds number varies depending on how similar the previous numerical solution is to the new solution. In general, we found an increase of 10% to be suitable. However, if the nonlinear solver fails to converge, this value should be decreased and the step repeated until it does.

6. Refine the mesh. Calculate the numerical error according to the chosen target functional using the DWR method in Chapter 6. Then, refine the $U\%$ of elements with the largest error values, and coarsen the $L\%$ of elements with the smallest error values. Mesh refinement should be carried out once the flow reaches predetermined Reynolds numbers. These values are chosen such that refinement passes are more frequent when the Reynolds number is approaching the target value set out in Step 1. Additionally, we limit the total number of mesh refinement passes until the Reynolds number has increased beyond the turbulent transition value. We found 4 passes to be a reasonable balance between numerical accuracy and computational efficiency. Additional refinement passes at these lower Reynolds numbers left the final mesh over refined in areas we expect to have almost laminar flow.

7. Solve on the new refined mesh for the same flow parameters.

8. Repeat Steps 5,6 and 7 until the target Reynolds number is met. Then, continue refining the final numerical solution until a desired degree of numerical accuracy is met according to the chosen target functional.

8.2 Future Developments

This work has demonstrated the viability of interior penalty DGFEMs for industrial applications. Whilst this research has been restricted to the computation of turbulent incompressible flows on various mesh types, it would be interesting to extend this to include compressible flows. Incompressible flows are used widely in the turbomachinery industry, often for modelling flows through seals and pressure outlets. However, in order to take advantage of DGFEMs natural ability to remain stable and to handle adverse solution gradients such as those generated by shock waves in supersonic flows, one should eventually consider the compressible Navier-Stokes equations. Whilst the compressible Navier-Stokes equations are discussed at length in the literature [80], this type of problem has not yet been

considered for turbulent flows on general computational meshes consisting of polytopic or curvilinear elements.

In order to improve the continuation type algorithm detailed above, we may wish to consider anisotropic mesh refinement. Placing long, thin elements along aerofoil or blade edges will better capture boundary layer flows. These flows change and develop very quickly when the observer is moving perpendicular to a boundary wall, but change reasonably slowly when the observer is moving parallel to the boundary wall. Therefore, by exploiting the stability afforded by the curvilinear penalty parameter proposed in Chapter 5, we are able to maintain the geometry of the boundary, as well as improve the efficiency of the DGFEM solver compared to using standard isotropic refinement. This is because the value of C_γ, allows for elements to be scaled in either of the coordinate directions, allowing effective refinement of very general aerofoil geometries. The implementation of this strategy is detailed in Figures 8.2.1,8.2.2 and 8.2.3.

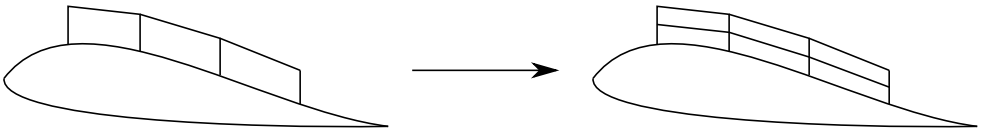

Figure 8.2.1: Anisotropic refinement parallel to the boundary wall.

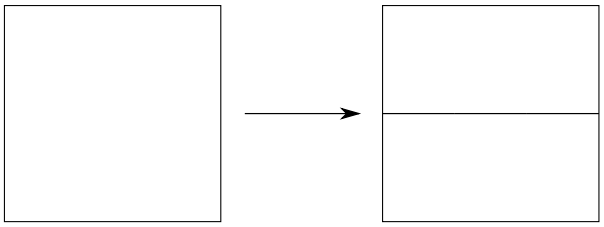

Figure 8.2.2: Anisotropic refinement of a quadrilateral element. (Type 1).

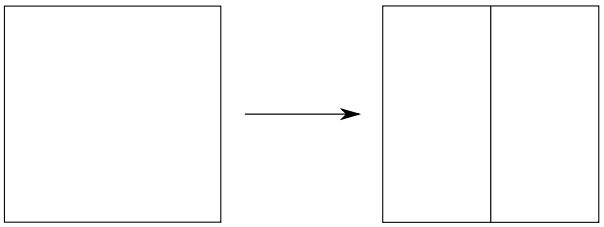

Figure 8.2.3: Anisotropic refinement of a quadrilateral element. (Type 2).

Other types of anisotropic refinement could also be considered, including weighted node placement [77]. This is particularly useful when trying to capture streamlines in a solution. For a complete review of these techniques, we refer to [77] and the references cited therein.

The numerical results presented in Chapters 5 and 6 are calculated on meshes that contain curvilinear boundary elements only. Whilst this does address the issue of current generation FVM solvers, which employ piecewise-linear polynomials to approximate what are often quadratic curves in the underlying geometry we have been able to extend this concept further into polytopic meshes. The next natural extension is to employ a mesh which consists of only curvilinear elements, directed such that they capture laminar streamlines parallel to their largest principal axis. The current discontinuity-penalisation function does allow for the construction of a stable DGFEM in this setting, but the automatic mesh refinement algorithm must be updated. A node shifting algorithm would also need to be considered in order to implement this concept, see Figure 8.2.4.

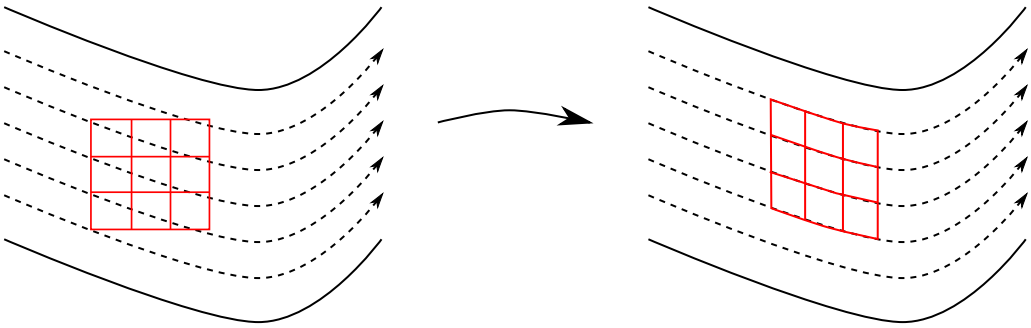

Figure 8.2.4: Mesh refinement about streamlines.

In this work, we only considered the control of the error in some target functional. We did not consider the use of norm based error estimates for Chapters 6 and 7 . It would be very informative to industry representatives to present a comparison between the two approaches, and to see how applicable the differing refinement algorithms are for high Reynolds number turbulent flows.

Another point of consideration for industrial applications is the correct method of resolving rotational frames of reference within computational meshes. A full turbine stage is traditionally modelled by three separate computational meshes as shown in Figure 8.2.5, with periodic boundary conditions analogous to those in Section 5.4.1, and mixing planes (red) introduced where the meshes join. This is to reflect the design of industrial turbomachinery, with a set of rotating turbine blades between two stages of stationary stator blades. Flow passes through the domain from left to right, with the mixing planes used

to correct the flow as it moves from a stationary to a rotational frame of reference, and then back again before exiting. However, implementation of this mixing interface in the interior penalty DGFEM for the RANS equations is unclear, in particular, the definition of the discontinuity-penalisation parameter across the interface. This would therefore prove to be a useful avenue of future research, especially for validating DGFEMs for industrial purposes.

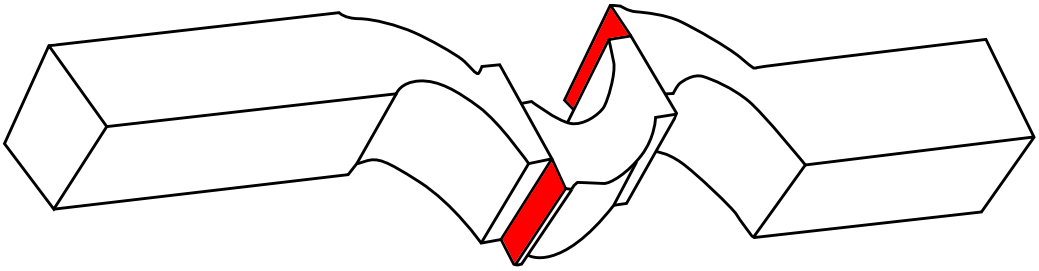

Figure 8.2.5: Three-dimensional 1.5 stage turbomachinery domain.

The approaches to mesh design and to the definition of the discontinuity-penalisation parameter could be extended further to consider other techniques for turbulence modelling. With ever-increasing computing power, especially in terms of computing clusters, Large Eddy Simulation (LES) is becoming more attractive for industrial applications. The ability to resolve the largest of the turbulent eddies, compared to RANS, which resolves none and only approximates the flow, is very appealing to industry leaders such as General Electric. This then offers the opportunity to resolve more of the flow features, dependent only on the time available to carry out the simulation. DGFEMs for LES have been considered in the literature [63], but have never been considered for polytopic meshes. This would involve the construction of a new DGFEM discretisation for the time-dependent Navier-Stokes equations, as well as considerations for new boundary conditions. We believe that this type of problem would benefit greatly from the application of polytopic meshes, with elements of arbitrary shape able to accurately capture the complicated geometry of the swirling eddies.

Through the exploration of an interior penalty DGFEM for LES, we would also need to explore the question of time step adaptivity. By their very nature, time dependent problems must be solved for each time step. However, this has a very high computational cost associated, and as such, is a worry for the industrial sector due to the time constraints they place on numerical simulations. A time step adaptive method allows for variation in the length of the time steps of the calculation, extending the length during intervals

of almost laminar flows, whilst contracting them during intervals of high turbulence. As such, adaptivity in space, as well as time, would go a considerable way to resolving the practical issues associated with LES for large-scale industrial applications.

Bibliography

[1] Ahmad, B., Alsaedi, A., Brezzi, F., Marini, L.D., Russo, A., Equivalent projectors for virtual element methods. *Comput. Math. Appl. 66(3)*, 376–391, 2013.

[2] Alfonsi, G., Reynolds-Averaged Navier-Stokes Equations for Turbulence Modeling, *Applied Mech. Reviews*, *Vol 62*, 2009.

[3] Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y., Multifrontal parallel distributed symmetric and unsymmetric solvers, *Comput. Methods in Appl. Mech. Eng.*, *Vol. 184*, 501-520, 2000.

[4] Antonietti, P.F., Giani, S., Houston, P., hp-Version composite discontinuous Galerkin methods for elliptic problems on complicated domains. *SIAM J. Sci. Comput. 35(3)*, A1417–A1439, 2013.

[5] Antonietti, P.F., Cangiani, A., Collis, J., Dong, Z., Georgoulis, E.H., Giani, S., Houston, P., Review of discontinuous Galerkin finite element methods for partial differential equations on complicated domains, *in Building Bridges: Connections and Challenges in Modern Approaches to Numerical Partial Differential Equations*, *ed. by Barrenechea, G.R., Brezzi, F., Cangiani, A., Georgoulis, E.H., Springer International Publishing, Cham*, 281–310, 2016.

[6] Antonietti, P.F., Mazzieri, I., High-order Discontinuous Galerkin methods for the elastodynamics equation on polygonal and polyhedral meshes, *Comput. Methods in Appl. Mech. Eng.*, *Vol. 342*, 414-437, 2018.

[7] Antonietti, P.F., Verani, M., Vergara, C., Zonca, S., Numerical solution of fluid-structure interaction problems by means of a high order Discontinuous Galerkin method on polygonal grids, *Fin. Ele. in Anal. and Design*, *Vol 159*, 1-14, 2019.

[8] ARCHER. EPSRC/University of Edinburgh. [Online]. Available: https://epsrc.ukri.org/research/facilities/hpc/intro/

[9] Armaly, B.F., Durst, F., Pereira, J.C.F., Schönung, B., Experimental and theorectical investigation of backward-facing step flow, *J. Fluid Mech.*, *Vol. 127*, 473-496, 1983.

[10] Arnold, D.N., An interior penalty finite element method with discontinuous elements, *SIAM J. Numer. Anal.*, 19(4):742–760, 1982.

[11] Arnold, D.N., Brezzi, F., Cockburn, B., Marini, L.D., Discontinuous Galerkin methods for elliptic problems. In Discontinuous Galerkin methods (Newport, RI, 1999), *volume 11 of Lect. Notes Comput. Sci. Eng.*, *Springer*, *Berlin*, 89–101, 2000.

[12] Arnold, D.N., Brezzi, F., Cockburn, B., Marini, L.D., Unified analysis of discontinuous Galerkin methods for elliptic problems, *SIAM J. Numer. Anal.*, 39(5):1749–1779 (electronic), 2001/02.

[13] Asghar, M.A., Liu, Y., Cui, J., Lu, L., Investigation of Unsteady Flow Interactions in a Transonic High Pressure Turbine Using Nonlinear Harmonic Method, *Energies*, 2018.

[14] Avila, K., Moxey, D., de Lozar, A., Avila, M., Barkley, D., Hof, B., The Onset of Turbulence in Pipe Flow, *Science vol. 333*, *issue 6039*, 192-196, 2011.

[15] Baas, E., Kuiper, J.H., A numerical model of heterogeneous surface strains in polymer scaffolds, *J. Biomech. 41*, 1374–1378, 2008.

[16] Baas, E., Kuiper, J.H., Yang, Y., Wood, M.A., El Haj, A.J., In vitro bone growth responds to local mechanical strain in three-dimensional polymer scaffolds, *J. Biomech. 43*, 733–739, 2010.

[17] Babuška, I., Error Bounds for Finite Element Method, *Numerische Mathematik*, *16*, 322-333, 1971.

[18] Babuška, I., The finite element method with penalty, *Math. Comp. 27*, 221–228, 1973.

[19] Babuška, I., Osborn, J.E., Generalized finite element methods: their performance and their relation to mixed methods. *SIAM J. Numer. Anal. 20(3)*, 510–536, 1983.

[20] Baker, G.A., Finite Element Methods for Elliptic Equations Using Nonconforming Elements, Mathematics of Computation, *vol. 31*, *number 137*, 45-59, 1977.

[21] Barth, T.J., Larson, M.G., *A-posteriori* error estimation for higher order Godunov finite volume methods on unstructured meshes, *in Finite volumes for complex applications III*, *Hermes Science Pub.*, *London*, 41-63, 2002.

[22] Bangerth, W., Rannacher, R., Adaptive finite element methods for differential equations, *Springer Science & Business Media*, 2003.

[23] Bass, R.F., Diffusion and Elliptic Operators, Springer, NewYork, 1997.

[24] Bassi, F., Rebay, S., Mariotti, G., Pedinotti, S., Savini, M., A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flows, *in 2nd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*, *Technologisch Instituut*, *Antwerpen*, *Belgium*, 99-108, 1997.

[25] Bassi, F., Crivellini, A., Rebay, S., Savini, M., Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and $k-\omega$ turbulence model equations, *Comp. and Fluids* 34, 507-540, 2005.

[26] Bassi, F., Botti, L., Colombo, A., Ghidoni, A., Rebay, S., Discontinuous Galerkin for turbulent flows. Adaptive high-order methods in computational fluid dynamics. *Adaptive high-order methods in computational fluid dynamics*, *Advances in Computational Fluid Dynamics*, *World Science Books*, *Ed. Wang, Z.J.*, 2011.

[27] Bassi, F., Botti, L., Colombo, A., Rebay, S., Agglomeration based discontinuous Galerkin discretization of the Euler and Navier-Stokes equations, *Comput. Fluids 61*, 77–85, 2012.

[28] Bassi, F., Botti, L., Colombo, A., Di Pietro, D.A., Tesini, P., On the flexibility of agglomeration based physical space discontinuous Galerkin discretizations, *J. Comput. Phys. 231(1)*, 45–65, 2012.

[29] Bassi, F., Botti, L., Colombo, A., Agglomeration-based physical frame dG discretizations: an attempt to be mesh free, *Math. Models Methods Appl. Sci. 24(8)*, 1495–1539, 2014.

[30] Becker, R., Rannacher, R., Weighted *a posteriori* error control in FE methods, *In et al. H. G. Bock*, *World Sci. Publ.*, *Singapore*, 1995.

[31] Becker, R., Rannacher, R., An optimal control approach to *a posteriori* error estimation in finite element methods, *In A. Iserles*, *editor, Acta Numerica 2001*, *vol 37*, *Cambridge University Press*, 1–225, 2001.

[32] Beirão da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L.D., Russo, A., Basic principles of virtual element methods, *Math. Models Methods Appl. Sci. 23(1)*, 199–214, 2013.

[33] Beirão da Veiga, L., Brezzi, F., Marini, L.D., Russo, A., Serendipity nodal VEM spaces, *Comput. Fluids 141*, 2–12, 2016.

[34] Bernardi, C., Dauge, M., Maday, Y., Polynomials in the Sobolev world, *HAL report, hal-00153795*, 2007

[35] Brezzi, F., On the Existence, Uniqueness and Approximation of Saddle-Point Problems Arising form Lagrange Multipliers, *R.A.I.R.O., 8, R2*, 129-151, 1974.

[36] Cangiani, A., Georgoulis, E.H., Houston, P., hp-Version discontinuous Galerkin methods on polygonal and polyhedral meshes, *Math. Models Methods Appl. Sci. 24(10)*, 2009–2041, 2014.

[37] Cangiani, A., Dong, Z., Georgoulis, E.H., Houston, P., hp-Version discontinuous Galerkin methods for advection–diffusion–reaction problems on polytopic meshes, *ESAIM Math. Model. Numer. Anal. 50(3)*, 699–725, 2016.

[38] Cangiani, A., Dong, Z., Georgoulis, E.H., Houston, P., hp-Version Discontinuous Galerkin Methods on Polygonal and Polyhedral Meshes, *Springer Briefs in Mathematics*, 2017

[39] Cangiani, A., Dong, Z., Georgoulis, E.H., hp–Version space-time discontinuous Galerkin methods for parabolic problems on prismatic meshes, *SIAM Journal on Scientific Computing 39(4)*, 1251–1279, 2017.

[40] Cangiani, A., Manzini, G., Sutton, O.J., Conforming and nonconforming virtual element methods for elliptic problems, *IMA J. Numer. Anal. 37(3)*, 1317–1354, 2017.

[41] Čanić, S., Keyfitz, B., A smooth solution for a Keldysh type equation, *Comm. Partial Differential Equations 21*, 319-340, 1996.

[42] Canuto, C., Quarteroni, A., Approximation results for orthogonal polynomials in Sobolev spaces. *Math. Comp. 38(157)*, 67–86, 1982.

[43] Çengel, Y.A., Cimbala, J.M., Fluid Mechanics: Fundamentals and Applications, *Mc.Graw Hill, Vol. 1*, 2004.

[44] Cherdron, W., Durst, F., Whitelaw, J.H., Assymetric Flows and Instabilities in Symmetric Ducts with Sudden Expansions, *J. Fluid Mech., vol 84*, 13-31, 1978.

[45] Chin, E.B., Lasserre, J.B., Sukumar, N., Numerical integration of homogeneous functions on convex and nonconvex polygons and polyhedra, *Comput. Mech. 56(6)*, 967–981, 2015.

[46] Cliffe, A., Hall, E.J.C., Houston, P., Adaptive Discontinuous Galerkin Methods for Eigenvalue Problems Arising in Incompressible Fluid Flows, *SIAM J. Sci. Comput., vol. 31, no. 6*, 4607-4632, 2010.

[47] Cockburn, B., Karniadakis, G.E., Shu, C.W., The development of discontinuous Galerkin methods. In Discontinuous Galerkin methods (Newport, RI, 1999), *volume 11 of Lect. Notes Comput. Sci. Eng.*, *Springer*, *Berlin*, 3-50, 2000.

[48] Costa, R., Clain, S., Loubère, R., Machado, G., Very high-order accurate finite volume scheme on curved boundaries for the two-dimensional steady-state convection-diffusion equation with Dirichlet condition, *Elsevier Applied Mathematical Modelling*, 752-767, 2018.

[49] Deardorff, J., A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers, *J. Fluid Mech.*, *vol* 41, 453-480, 1970.

[50] Di Pietro, D.A., Ern, A., Mathematical Aspects of Discontinuous Galerkin Methods, *Mathématiques & Applications (Berlin) [Mathematics & Applications]*, *vol. 69, Springer, Heidelberg*, 2012.

[51] Di Pietro, D.A., Ern, A., A hybrid high-order locking-free method for linear elasticity on general meshes, *Comput. Methods Appl. Mech. Eng. 283*, 1–21, 2015.

[52] Di Pietro, D.A., Ern, A., Hybrid high-order methods for variable-diffusion problems on general meshes, *C. R. Math. Acad. Sci. Paris 353(1)*, 31–34, 2015.

[53] Di Pietro, D.A., Ern, A., Lemaire, S., An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators, *Comput.Methods Appl. Math. 14(4)*, 461–472, 2014.

[54] Dohrmann, C.R., Bochev, P.B., A stabilized finite element method for the Stokes problem based on polynomial pressure projections, *Int. J. Numer. Meth. Fluids*, 2000.

[55] Driver, D.M., Backwards-Facing Step Measurements at Low Reynolds Number, $Re_h = 5000$, *NASA Technical Memorandum 108807*, 1994.

[56] Durst, F., Melling, A., Whitelaw, J.H., Low Reynolds Number Flow over a Plane Symmetric Sudden Expansion, *J. Fluid Mech.*, *vol 64*, 111-128, 1974.

[57] Ebeida, M.S., Mitchell, S.A., Uniform random Voronoi meshes, in Proceedings of the 20th International Meshing Roundtable, *ed. by W.R. Quadros*, *Springer*, *Berlin/Heidelberg*, 273–290, 2012.

[58] Elman, H.C., Silvester, D.J., Wathen, A.J., Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics, 2006.

[59] Eriksson, K., Estep, D., Hansbo, P., Johnson, C., Introduction to adaptive methods for differential equations, *in Acta Numerica*, *ed. by A. Iserles*, *Cambridge University Press*, *Cambridge*, 105–158, 1995.

[60] Eymard, R., Gallouët, T., Herbin, R., Handbook of Numerical Analysis, *Elsevier B.V.*, *Volume 7*, 713-1018, 2000.

[61] Feistauer, M., Kučera, V., On a robust discontinuous Galerkin technique for the solution of compressible flow. *J. Comput. Phys. 224(1)*, 208–221, 2007.

[62] Ferguson, F., Mendez, J., Dodoo-Amoo, D., Dhanasar, M., The Performance Evaluation of an Improved Finite Volume Method that Solves the Fluid Dynamic Equations, *Conference Proceedings '2018 AIAA Aerospace Sciences Meeting'*, 2018.

[63] Frère, A., Sørensen, N.N., Hillewaert, K., Chatelain, P., Winckelmans, G., Discontinuous Galerkin methodology for Large-Eddy Simulations of wind turbine airfoils, *J. Phys.*, *Conference Series 753*, 2016.

[64] Fries, T.-P., Belytschko, T., The extended/generalized finite element method: an overview of the method and its applications, *Int. J. Numer. Methods Eng. 84(3)*, 253–304, 2010.

[65] Gallus, H.E., ERCOFTAC Test Case 6 Axial Flow Turbine Stage, *Seminar and Workshop on 3D Turbomachinery flow prediction III, Les Arcs, France*, 1995.

[66] Garai, A., Diosady, L.T., Murman, S.M., Madavan, N.K., DNS of flow in a low-pressure turbine cascade using a discontinuous Galerkin spectral-element method, *Proceedings of ASME Turbo Expo 2015: Turbine Technical Conference and Exposition*, 2015.

[67] Georgoulis, E.H., Discontinuous Galerkin methods on shape-regular and anisotropic meshes, *Ph.D Thesis*, *Computing Laboratory*, *University of Oxford*, 2003.

[68] Georgoulis, E.H., Hall, E.J.C., Houston, P., Discontinuous Galerkin Methods for Advection-Diffusion-Reaction Problems on Anisotropically Refined Meshes, *SIAM Journal on Scientific Computing 30(1)*, 246-271, 2007.

[69] Georgoulis, E.H., Inverse-type estimates on hp-finite element spaces and applications, *Math. Comput. 77(261)*, 201–219, 2008.

[70] Georgoulis, E.H., Discontinuous Galerkin methods for linear problems; an introduction, *Approximation Algorithms for Complex Systems*, *Springer Proceedings in Mathematics*, *vol. 3*, *Springer-Verlag*, *Berlin*, 2011.

[71] Giani, S., Houston, P., hp-Adaptive composite discontinuous Galerkin methods for elliptic problems on complicated domains, *Numer. Methods Partial Differ. Equ. 30(4)*, 1342–1367, 2014.

[72] Girault, V., Raviart, P.A., Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms, 1986.

[73] Gong, Y., Tanner, F.X., Comparison of RANS and LES Models in the Laminar Limit for a Flow Over a Backward-Facing Step Using OpenFOAM, *Nineteenth International Multidimensional Engine Modeling Meeting at the SAE Congress*, 2009.

[74] Gui-Qiang, C., The Tricomi Equation, *The Princeton Companion to Applied Mathematics*, *III:30*, *Princeton University Press*, 2015.

[75] Hackbusch, W., Sauter, S.A., Composite finite elements for problems containing small geometric details. Part II: implementation and numerical results, *Comput. Vis. Sci. 1*, 15– 25, 1997.

[76] Hackbusch, W., Sauter, S.A., Composite finite elements for the approximation of PDEs on domains with complicated micro-structures, *Numer. Math. 75*, 447–472, 1997.

[77] Hall, E.J.C., Anisotropic Adaptive Refinement for Discontinuous Galerkin Methods, *Ph.D Thesis*, *University of Leicester*, 2007.

[78] Harriman, K., Gavaghan, D., Senior, B., Süli, E., hp-version discontinuous Galerkin methods with interior penalty for partial differential equations with nonnegative characteristic form, *in Recent advances in scientific computing and partial differential equations*, *Contemp. Math.*, *Vol. 330*, 89-119, 2003.

[79] Hartmann, R., The role of the Jacobian in the adaptive discontinuous Galerkin method for the compressible Euler equations, *in Analysis and Numerics for Conservation Laws*, *Warnecke G (ed.)*, *Springer*, *Berlin*, 301–316, 2005.

[80] Hartmann, R., Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier–Stokes equations, *Int. J. Numer. Meth. Fluids 51*, 1131–1156, 2006.

[81] Hartmann R., Houston P., Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws, *SIAM Journal on Scientic Computing 24*, 979–1004, 2002.

[82] Hartmann, R., Houston, P., Symmetric Interior Penalty DG Methods for the Compressible Navier-Stokes Equations I: Method Formulation, *Int. J. Numer. Anal. Model. 3(1)*, 1-20, 2006.

[83] Hartmann, R., Houston, P., Symmetric Interior Penalty DG Methods for the Compressible Navier-Stokes Equations II: Goal-Oriented A Posteriori Error Estimation, *Int. J. Numer. Anal. Model. 3(2)*, 141-162, 2006.

[84] Hartmann, R., Houston, P., Error estimation and adaptive mesh refinement for aerodynamic flows, *in VKI LS 2010-01: 36th CFD/ADIGMA Course on hp-Adaptive and hp-Multigrid Methods, Oct. 26–30, 2009, ed. by H. Deconinck, Von Karman Institute for Fluid Dynamics, Sint-Genesius-Rode*, 2010.

[85] Hesthaven, J.S., Warburton, T., Nodal Discontinuous Galerkin Methods. Texts in Applied Mathematics, *Vol. 54, Springer, New York*, 2008.

[86] Hirsch, C., Numerical computations of internal and external flow, *John Wiley, Vol. 2*, 1990.

[87] Houston, P., Adjoint error estimation and adaptivity for hyperbolic problems, *in Handbook of Numerical Analysis: Handbook of Numerical Methods for Hyperbolic Problems. Applied and Modern, Vol. 18, ed. by Abgrall, R., Shu, C.-W.*, 233–261, 2016.

[88] Houston, P., AptoFEM v4.0.81 User Documentation, *School of Mathematical Sciences, University of Nottingham*, 2016.

[89] Houston, P., Schötzau, D., Wihler, T.P., Energy Norm A Posteriori Error Estimation for Mixed Discontinuous Galerkin Approximations of the Stokes Problem, *J. Sci. Comput., Vol. 22*, 2005.

[90] Houston, P., Schwab, Ch., Süli, E., Stabilized hp-finite element methods for first-order hyperbolic problems. *SIAM J. Numer. Anal. 37(5)*, 1618–1643, 2000.

[91] Houston, P., Süli, E., Stabilized hp-finite element approximation of partial differential equations with non-negative characteristic form, *Computing 66* (2), 99–119, 2001.

[92] Houston, P., Schwab, Ch., Süli, E., Discontinuous hp-finite element methods for advection-diffusion-reaction problems, *SIAM J. Numer. Anal.*, 39(6):2133–2163 (electronic), 2002.

[93] Ince, N., Director at General Electric, Industry point of contact for this project, 2014-2019.

[94] Johansson, A., Larson, M.G., A high order discontinuous Galerkin Nitsche method for elliptic problems with fictitious boundary, *Numer. Math. 123(4)*, 607–628, 2013.

[95] Johnson, C., Pitkäranta, J., An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation, *Math. Comp. 46, no. 173*, 1–26, 1986.

[96] Karypis, G., Kumar, V., A fast and highly quality multilevel scheme for partitioning irregular graphs, *SIAM J. Sci. Comput. 20(1)*, 359–392, 1999.

[97] Keldyš, M.V., On certain cases of degeneration of equations of elliptic type on the boundary of a domain, *Dokl. Akad. Nauk SSSR (N.S)*, *77*, 181-183, 1951.

[98] Kim, D.-H., Yang, J.-H., Boundary Layer and Near-wake Measurements of NACA 0012 Airfoil at Low Reynolds Numbers, *47th AIAA Aerospace Sciences Meeting*, 2009.

[99] Kolmogorov, A.N., Equations of motion of an incompressible turbulent fluid, *Izv Akad Nauk SSSR Ser Phys VI No 1-2*, 56, 1942.

[100] Krause, E., Jäger, W., High Performance Computing in Science and Engineering '98, *Springer*, 1998.

[101] Lasserre, J.B., Integration on a convex polytope, *Proc. Am. Math. Soc. 126(8)*, 2433–2441, 1998.

[102] Launder B.E., Spalding, D.B., Mathematical Models of Turbulence, *Academic Press*, 1972.

[103] Launder, B.E., Spalding, D.B., The numerical computation of turbulent flows, *Computer Methods in Applied Mechanics and Engineering 3 (2)*, 269–289, 1974.

[104] LeSaint, P., Raviart, P.A., On a finite element method for solving the neutron transport equation, *in Mathematical aspects of finite elements in partial differential equations (Proc. Sympos., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1974)*, *Publication No. 33. Math. Res. Center, Univ. of WisconsinMadison, Academic Press*, *New York*, 89–123, 1974.

[105] Moukalled, F., Mangani, L., Darwish, M., The Finite Volume Method in Computational Fluid Dynamics, *Fluid Mechanics and Its Applications*, *Springer*, 2016.

[106] Mousavi, S.E., Sukumar, N., Numerical integration of polynomials and discontinuous functions on irregular convex polygons and polyhedrons, *Comput. Mech. 47(5)*, 535–554, 2011.

[107] Niederschulte, M.A., Adrian, R.J., Hanratty, T.J., Measurements of turbulent flow in a channel at low Reynolds numbers, *Exp. in Fluids 9*, 222-230, 1990.

[108] Nitsche, J., Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind, *Abh. Math. Sem. Univ. Hamburg*, 36:9–15, 1971.

[109] Oleĭnik, O.A., Radkevič, E.V., Second order equations with nonnegative characteristic form, *Plenum Press*, *New York*, *translated from the Russian by Fife*, *P.C.*, 1973.

[110] Oñate, E., Manzan, M., Stabilization Techniques for Finite Element Analysis of Convection-Diffusion Problems, *International Centre for Numerical Methods In Engineering*, *CIMNE No-183*, 2000.

[111] Patankar, S.,V., Numerical Heat Transfer and Fluid Flow, *first edition*, *CRC Press*, 1980.

[112] Peng, X., Zhimin, D., Liang, B., Qi, Z., Darcy-Stokes Streamline Simulation for the Tahe-Fractured Reservoir With Cavities, *SPE Journal*, *Vol. 14*, *Issue 3*, 543-552, 2009.

[113] Peraire, J., Persson, P.-O., High-Order Discontinuous Galerkin Methods for CFD, *World Sci. Review*, 2010.

[114] Pifer, E.A., Bramesfeld, G., Measuring Wing Profile Drag using an Integrating Wake Rake, *Technical Soaring, vol. 36 no. 3*, 2012.

[115] Prandtl, L., Bericht über Untersuchungen zur ausgebildeten Turbulenz, *Z. angew*, *Math. Mech.*, 5, 136–139, 1925.

[116] Prudhomme, S., Pascal, F., Oden, J. T., Romkes, A., Review of a priori error estimation for discontinuous Galerkin methods. Technical report, *TICAM Report 00–27*, *Texas Institute for Computational and Applied Mathematics*, 2000.

[117] Reed, W.H., Hill, T.R., Triangular mesh methods for the neutron transport equation, *Technical Report LA-UR-73-479*, *Los Alamos Scientific Laboratory*, 1973.

[118] Renda, S.M., Hartmann, R., De Bartolo, C., Wallraff, M., A high-order discontinuous Galerkin method for all-speed flows, *Int. J. Numer. Meth. Fluids*, 2014.

[119] Rivière, B., Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations, *Frontiers in Applied Mathematics*, *Society for Industrial and Applied Mathematics*, *Philadelphia*, *PA*, 2008.

[120] Rivière, B., Wheeler, M.F., Girault, V., Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems, Part I. *Computational Geosciences*, 3:337–360, 1999.

[121] Rivière, B., Wheeler, M.F., Optimal error estimates for discontinuous galerkin methods applied to linear elasticity problems, *Technical report*, *TICAM Report 00–30*, *Texas Institute for Computational and Applied Mathematics*, 2000.

[122] Rivière, B., Wheeler, M.F., Girault, V., A priori error estimates for finite element methods based on discontinuous approximation spaces for elliptic problems, *SIAM J. Numer. Anal.*, 39(3):902–931 (electronic), 2001.

[123] Saffman, P.G., The velocity of viscous vortex rings, *Stud. Appl. Math. 49*, 371-380, 1970.

[124] Schwab, Ch., p- and hp-Finite element methods: Theory and Applications in Solid and Fluid Mechanics, *Oxford University Press*, *Oxford*, 1998.

[125] Schötzau, D., Schwab, Ch., Toselli, A., Mixed hp-DGFEM for incompressible flows, *Eidgenössische Technische Hochschule*, 2002.

[126] Siebenborn, M., Schulz, V., Schmidt, S., A curved-element unstructured discontinuous Galerkin method on GPUs for the Euler equations, *Computing and Visualization in Science 15(2)*, 61–73, 2012.

[127] Smagorinsky, J., General circulation experiments with the primitive equations, i. the basic experiment, *Monthly Weather Review*, *Vol 91*, 99-164, 1963.

[128] Spalding, D.B., Heat transfer from turbulent Seperated Flows, *J. Fluid Mech. Vol. 27*, 97-109, 1967.

[129] Sukumar, N., Tabarraei, A., Conforming polygonal finite elements, *Int. J. Numer. Methods Eng. 61(12)*, 2045–2066, 2004.

[130] Süli, E., Schwab, Ch., Houston, P., hp–DGFEM for partial differential equations with nonnegative characteristic form, *in Cockburn*, *B.*, *Karniadakis*, *G.E.*, *Shu*, *C.W.*, *editors*, *Discontinuous Galerkin Methods: Theory*, *Computation and Applications*, *Lecture Notes in Computational Science and Engineering, vol. 11*, *Springer*, 221–230, 2000.

[131] Talischi, C., Paulino, G.H., Pereira, A., Menezes, I.F.M., Polymesher: a general-purpose mesh generator for polygonal elements written in Matlab, *Struct. Multidiscip. Optim. 45*, 309–328, 2012.

[132] Top500, (2018, Nov). [Online]. Available: https://www.top500.org/list/2018/11/

[133] UK Department for Business, Energy and Industrial Strategy, December 2018.

[134] Versteeg, H.K., Malalasekera, W., An Introduction to Computational Fluid Dynamics: The Finite Volume Method, *second edition*, *Pearson Education Limited*, 2007.

[135] Volmar, T., Brouillet, B., Benetschik, H., Gallus, H.E., Test Case 6: 1-1/2 Stage Axial Flow Turbine - Unsteady Computation, *ERCOFTAC Turbomachinery Seminar and Workshop*, 1998.

[136] Von Kármán, T., Mechanische Ähnlichkeit und Turbulenz, *Weidmannsche Buchh.*, 1930.

[137] Walraevens, R., Gallus, H.E., European Research Community On Flow Turbulence And Combustion. ERCOFTAC SIG on 3D Turbomachinery Flow Prediction, Test-case 6-1-1/2 Stage axial flow turbine, *ERCOFTACAPPACET Seminar and Workshop on Turbomachinery Flow Prediction VIII.*

[138] Walraevens, R., Gallus, H.E., Test Case 6: 1-1/2 Stage Axial Flow Turbine, 1997.

[139] Warburton, T., Hesthaven, J.S., On the constants in hp-finite element trace inverse inequalities, *Comput. Methods Appl. Mech. Eng. 192(25)*, 2765–2773, 2003.

[140] Wilcox, D.C., Traci, R.M., A Complete Model of Turbulence, *AIAA Paper*, 76-351, 1976.

[141] Williams, P.T., Baker, A.J., Numerical Simulations of Laminar Flow over a 3D Backward-Facing Step, *Int. J. Numer. Meth. Fluids.*, *Vol 24*, 1159-1183, 1997.

[142] Wheeler, M.F., An elliptic collocation-finite element method with interior penalties, *SIAM J. Numer. Anal.*, 15(1):152–161, 1978.

[143] Wilcox, D.C., Rubesin, M.W., Progress in Turbulence Modeling for Complex Flow Fields Including Effects of Compressibility, *NASA TP-1517*, 1980.

[144] Wilcox, D.C., Reassessment of the Scale Determining Equation for Advanced Turbulence Models, *AIAA Journal*, *Vol. 26*, *No. 11*, 1299-1310, 1988.

[145] Wilcox, D.C., Turbulence Modeling for CFD, *third edition*, 2006.

[146] Xiao, H., Gimbutas, Z., A numerical algorithm for the construction of efficient quadrature rules in two and higher dimensions, *Comput. Math. Appl. 59(2)*, 663–676, 2010.

[147] Zienkiewicz, O.C., Taylor, R.L., The finite element method, *Mc.Graw Hill*, *Vol. 2*, 1991.