Provably Convergent Anisotropic Output-Based Adaptation for Continuous Finite Element Discretizations

Hugh Alexander Carson

S.M. Massachusetts Institute of Technology (2016)
M.Eng. University of Cambridge (2014)

Submitted to the Department of Aeronautics and Astronautics in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computational Science and Engineering at the Massachusetts Institute of Technology

February 2020

Author . . . .
Department of Aeronautics and Astronautics
Thursday October 31st 2019

Certified by . . . .
David L. Darmofal
Professor of Aeronautics and Astronautics, MIT
Thesis Committee Chair

Certified by . . . .
Bernardo Cockburn
Distinguished McKnight University Professor, School of Mathematics, University of Minnesota
Thesis Committee Member

Certified by . . . .
Jaime Peraire
H.N. Slater Professor of Aeronautics and Astronautics, MIT
Thesis Committee Member

Certified by . . . .
Masayuki Yano
Assistant Professor, Institute for Aerospace Studies, University of Toronto
Thesis Committee Member

Accepted by . . . .
Sertac Karaman
Associate Professor of Aeronautics and Astronautics, MIT
Chairman, Graduate Program Committee

Accepted by . . . .
Youssef Marzouk
Professor of Aeronautics and Astronautics, MIT
Co-Director, Computational Science and Engineering

Provably Convergent Anisotropic Output-Based Adaptation for Continuous Finite Element Discretizations

Hugh Alexander Carson
Submitted to the Department of Aeronautics and Astronautics
on Thursday October 31st 2019, in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Computational Science and Engineering

Abstract

The expansion of modern computing power has seen a commensurate rise in the reliance on numerical simulations for engineering and scientific purposes. Output error estimation combined with metric-based mesh adaptivity provides a powerful means of quantifiably controlling the error in these simulations, for output quantities of interest to engineers and scientists. The Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm, developed by Yano for Discontinuous Galerkin (DG) discretization, is a highly effective method of this class.

This work begins with the extension of the MOESS algorithm to Continuous Galerkin (CG) discretization which requires fewer Degrees Of Freedom (DOF) on a given mesh compared to DG. The algorithm utilizes a vertex-based local error decomposition, and an edge-based local solve process in contrast to the element-centric construction of the original MOESS algorithm. Numerical results for linear problems in two and three dimensions demonstrate the improved DOF efficiency for CG compared to DG on adapted meshes.

A proof of convergence for the new MOESS extension is then outlined, entailing the description of an abstract metric-conforming mesh generator. The framework of the proof is rooted in optimization, and its construction enables a proof of higher-order asymptotic rate of convergence irrespective of singularities. To the author's knowledge, this is the first such proof for a Metric-based Adaptive Finite Element Method in the literature.

A three dimensional Navier Stokes simulation of a delta wing is then used to compare the new formulation to the original MOESS algorithm. The required stabilization of the CG discretization is performed using a new stabilization technique: Variational Multi-Scale with Discontinuous sub-scales (VMSD). Numerical results confirm that VMSD adapted meshes require significantly fewer DOFs to achieve a given error level when compared to DG adapted meshes; these DOF savings are shown to translate into a reduction in overall CPU time and memory usage for a given accuracy.

Thesis Committee Members:

Professor David L. Darmofal	Professor of Aeronautics and Astronautics, MIT
Professor Jaime Peraire	H.N. Slater Professor of Aeronautics and Astronautics, MIT
Professor Masayuki Yano	Assistant Professor, Institute for Aerospace Studies, University of Toronto
Professor Bernardo Cockburn	Distinguished McKnight University Professor, School of Mathematics, University of Minnesota

Thesis Readers:

Doctor Steven Allmaras	Research Scientist, MIT
Doctor Ryan Glasby	Computational Engineer, Oak Ridge National Laboratory

Acknowledgements

Thus, 5 momentous years of my life draw to a close. My time at MIT has been one of the most demanding and most enjoyable experiences of my life: from the breaking of my hubris at having never done an all-nighter before arriving (oh how that changed...), to all the hours spent exploring the US (and the rest of the world) with friends. Trying to acknowledge everyone who helped me reach this moment is a daunting task, but in common with the task of writing this thesis, an eminently rewarding one.

I would first like to thank Professor David Darmofal, my inspiring mentor throughout my time at MIT, both for the initial invitation to come to MIT and his patience in the face of my somewhat unstructured ideas over the years. Receiving edits from you late on a Saturday night or early on a Sunday morning made many a friend of mine both envy me for your engagement with my work and pity me for your high standards. The members of my committee deserve thanks: Professor Masayuki Yano both for the solid foundation that this thesis was built upon and his insight that never failed to direct me to fruitful avenues, Professor Jaime Peraire for asking the cutting questions that improved my work so much, and Bernardo Cockburn for his wealth of experience and perspective that helped to hone my approach. I am indebted to Dr. Ryan Glasby and Dr. Steven Allmaras for agreeing to be thesis readers. Additionally I thank Steve for his incredible ability in my weekly meetings to decipher my workings without even the aid of the white board. I would like to thank Mike Park for allowing me to visit NASA Langley for the summer, both for the research perspective and for all the Wednesday night sailing. The aid of Robin Courchesne-Sato, Jean Sofranas, Beth Marois and Anthony Zolnik should also be acknowledged, without whom I doubt anything in the ACDL would continue to function. In terms of funding, this thesis was also supported by: The Boeing company, technical monitor Dr. Mori Mani; NASA, technical monitor Dr. Michael Park; and Saudi Aramco, technical monitor Dr. Ali Dogru.

The numerical work of this thesis was all performed using the Solution Adaptive Numerical Simulator (SANS) research code and I'd like to thank all the members of the SANS team who made this possible: Savithru, Phil, Shun, Ben, Carmen, Cory, Steve, Arthur Huang, and Marshall Galbraith whose insistence on unit testing, though vexing at times, no doubt saved many hours in the end. I would particularly like to thank Arthur Huang with whom I have had a remarkable relationship during my time here. I have lost track of the number of times a problem intractable to one of us alone would be swiftly dispatched when we joined forces, and the stabilized method used in the final chapter of this thesis was developed in large part by him. Aside from the work, I have been blessed with great friends in the ACDL, including not once but twice with great cube mates. Firstly, thank you to Carlee, Giulia and Philippe of the DogeCube, you made the move to MIT that much easier and showed me what will remain the biggest tree I'll ever see. Secondly thank you to Max, Nisha and Cory, for the sounding board for ideas, the companionship as the lack of daylight kicked in and for the endless seam of comedy that made coming into the lab so much more enjoyable. I'd also like to thank Patrick for making me feel so welcome when I arrived, and for all the fun thereafter.

Outside of the lab, I'd like to thank Parker for all the good times, you were always there for a beer, a windsurf or to just take an overly long lunch break; developing such a friendship over my time here has been a tremendous joy. I would also like to thank my housemates: Sebastien, Spencer, Max, Maximilien, Alena and Anaëlle, for all the parties, shared shows and getting me to leave the lab during the daylight. Also, Aaron and the ice hockey guys: thanks for the good times and I acknowledge that the joy I got from those games was totally disproportionate to my skill.

Stretching back to my time at Gonville and Caius, Professor Rob Miller deserves my thanks for starting me off on the path to a PhD; it was your infectious enthusiasm that swept me up into the world of academia all those years ago. Additionally, I have him to thank for sitting me next to David Darmofal over dinner, where the idea of coming to MIT was planted in my brain. Stretching yet further back, I would like to thank Bobby, Josh and Brian; though it has been many years since I have sat in a rowing boat, the discipline and belief in hard work that you three taught me has served me well in the face of everything I have encountered since.

Finally, I would like to thank my family. Charley, Kate, Mum and Dad, without your support and encouragement none of this would have been possible. One of the hardest thing about my time here was that you were all over there.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The growth of computing has continued virtually unabated since the birth of the modern computer in the middle of the 20th century. This expansion of computation has profoundly affected engineering practice, as capabilities that once were the preserve of super computing facilities are now ubiquitous. Complex numerical simulations of Partial Differential Equations (PDEs) are now frequently employed within the typical engineering workflow. The results from these simulations may have profound impacts on engineering decisions, thus their accuracy and reliability can prove of paramount importance.

1.1 *Motivation*

The finite element method is a powerful method for the simulation of PDEs in large part because it is built upon a solid mathematical foundation[1]. This solid foundation enables the robust treatment of boundary conditions, the incorporation of higher-order basis functions and the calculation of *super convergent* functional outputs. Though simulations have increased in complexity, engineers are typically concerned with various functional outputs of the solution to a PDE as opposed to the entire solution[1]. The efficient and reliable prediction of functional outputs remains challenging, and a primary source of this difficulty is the design of suitable meshes for use with discretizations. Two immediate consequences of a poorly designed mesh are:

- Inefficiency - the mesh may concentrate computational effort in regions where it is unneeded.
- Inaccuracy - the mesh may not resolve important physical features, thereby providing erroneous output[2].

1. Ciarlet, *Basic error estimates for elliptic problems.* 1991

[1] Some typical quantities of interest include: the lift, drag and moment of an airfoil[2], species generation in a volume [3], the force applied on a domain boundary[4]

[2] Additionally, failing to resolve strong gradients can result in poor robustness in the solution process. Thus even obtaining a solution to a non-linear systems of equations may become difficult on poorly designed meshes.

Of these, inaccuracy is the more insidious issue as it can be much harder to detect. In practical settings, engineers rely on best-practice heuristics[3] to inform the design of these meshes, which may perform poorly in situations where the best practices are not applicable. Occasionally uniform mesh refinement studies are performed on these meshes to attempt to detect numerical convergence. Uniform refinement of a $3D$ simplex with N vertices and average size h, will give $8N$ nodes and an average mesh size $\frac{h}{2}$. This process of refinement could then be repeated until a computational resource budget has been depleted, with the corresponding sequence of solutions then used to modify the coarser meshes or, more often, to justify their use.

Mesh adaptation addresses this issue through an automated process of mesh modification that targets solution features of importance. The relative importance of these various features is weighed through *a posteriori* error estimation. A simplified example of a mesh adaptation process is shown in figure 1.1.

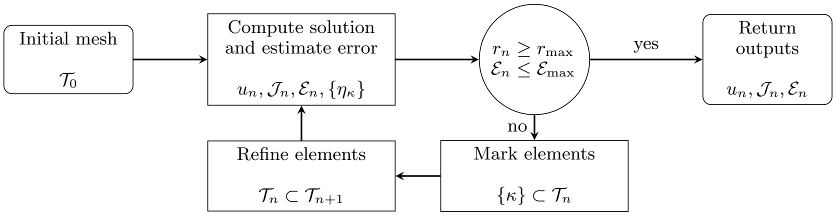

Initial mesh
$\mathcal{T}_0$
Compute solution and estimate error
$u_n, \mathcal{J}_n, \mathcal{E}_n, \{\eta_\kappa\}$
$r_n \geq r_{\max}$
$\mathcal{E}_n \leq \mathcal{E}_{\max}$
yes
Return outputs
$u_n, \mathcal{J}_n, \mathcal{E}_n$
no
Refine elements
$\mathcal{T}_n \subset \mathcal{T}_{n+1}$
Mark elements
$\{\kappa\} \subset \mathcal{T}_n$

Figure 1.1: A classical mesh adaptation process starting from initial mesh $\mathcal{T}_0$, computing solution u_n, functional output $\mathcal{J}_n$, error estimate $\mathcal{E}_n$, and localized error estimates $\{\eta_\kappa\}$, whilst observing resource limit $r_{\max}$ and error tolerance $\mathcal{E}_{\max}$.

The input for an adaptive mesh refinement process, as in figure 1.1, consists of the PDE, an initial mesh, $\mathcal{T}_0$, and the output functional of interest, $\mathcal{J}$. In the adaptation loop a solution, u_n, is computed on the mesh and the error, $\mathcal{E}_n$, estimated and localized to elements, $\{\eta_\kappa\}$. If the error satisfies a tolerance, $\mathcal{E}_{\max}$, or a resource limit, $r_{\max}$, is reached then the loop exits returning the solution, the computed output and the estimated error. If the tolerance is not satisfied and the resource limit has not been reached, then a set of elements $\{\kappa\} \subset \mathcal{T}_n$ are marked for refinement. The resulting mesh $\mathcal{T}_{n+1}$ is a *nested* refinement, by which it is meant that each element of $\mathcal{T}_{n+1}$ is either in $\mathcal{T}_n$, or comes from a subdivision of an element of $\mathcal{T}_n$.

A *metric* based mesh adaptation algorithm, depicted in figure 1.2, seeks to optimize the Riemannian metric field, $\mathcal{M}(x)$, which encodes the length of all the edges of a mesh. Starting from an initial mesh $\mathcal{T}_0$, an *implied metric* $\mathcal{M}_I(\mathcal{T}_0)$ is computed, typically using an interpolation scheme such as Affine-Invariant interpolation[5] or Log Euclidean interpolation[6]. The solution, u_n, functional output, $\mathcal{J}_n$, and estimated error, $\mathcal{E}_n$, are then computed for the mesh, and if either the tolerance, $\mathcal{E}_{\max}$, is satisfied or the resource allocation, $r_{\max}$, reached, the optimization

[3] For instance y^+ based estimates of boundary layer resolution and stationary shock wave tables.

5. Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

6. Arsigny *et al.*, *Log-Euclidean Metrics for Fast and Simple Calculus on Diffusion Tensors*. 2006

will return the solution, functional output, and estimated error. If neither the tolerance is achieved nor the resource limit is violated, then the adaptation proceeds and the effects of metric variation on the output error is modeled. These models are then optimized subject to a cost constraint, $\mathbb{C}_n$, resulting in a new *requested metric* $\mathcal{M}_{R,n}$ that ought to reduce the error. This requested metric is then passed to a metric mesher, along with the current mesh, $\mathcal{T}_n$, which generates a new mesh, $\mathcal{T}_{n+1}$, whose implied metric $\mathcal{M}_I(\mathcal{T}_{n+1})$ is closer to the requested metric $\mathcal{M}_{R,n}$.

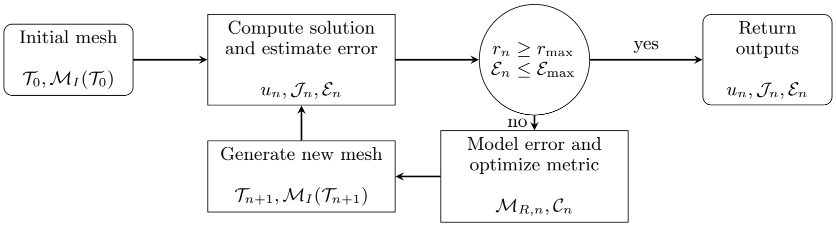

Initial mesh
$\mathcal{T}_0, \mathcal{M}_I(\mathcal{T}_0)$
Compute solution and estimate error
$u_n, \mathcal{J}_n, \mathcal{E}_n$
$r_n \geq r_{\max}$
$\mathcal{E}_n \leq \mathcal{E}_{\max}$
yes
Return outputs
$u_n, \mathcal{J}_n, \mathcal{E}_n$
no
Generate new mesh
$\mathcal{T}_{n+1}, \mathcal{M}_I(\mathcal{T}_{n+1})$
Model error and optimize metric
$\mathcal{M}_{R,n}, \mathcal{C}_n$

Figure 1.2: A metric adaptive process starting from initial mesh $\mathcal{T}_0$, computing solution u_n, functional output $\mathcal{J}_n$, and error estimate $\mathcal{E}_n$, whilst observing resource limit $r_{\max}$, and error tolerance $\mathcal{E}_{\max}$.

The Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm of Yano[7] has proven to be a highly effective metric-based mesh adaptation algorithm. The algorithm was originally devised for the Discontinuous Galerkin (DG) discretization which was particularly well-suited to the algorithm. However, the DG discretization is known to be relatively inefficient in terms of number of degrees of freedom (DOF) due to duplication on the traces of elements. MOESS contains within it a re-meshing stage which means that any two sequential meshes in the adaptation process are not related by any nested structure, this lack of structure means that the existing AFEM literature is not directly applicable.

1.2 *Background*

This sections represents a summary of the literature relating to the topics contained in this thesis.

Higher Order Discretization

Higher order discretization offers a way of increasing the accuracy of a given solution on a fixed mesh. Classical polynomial interpolation estimates are governed by the order of the approximation and the regularity of the interpolated function[8], where for $v \in H^r(\kappa)$, $\kappa \subset \mathbb{R}^d$

$$|v - \mathcal{I}v|_{H^m(\kappa)} \leq C h_\kappa^{t-m} |v|_{H^t(\kappa)}, \; t = \min(r, p+1), \tag{1.1}$$

where H^m denotes the Sobolev space whose m-th weak derivative is square integrable, $\mathcal{I}$ is the interpolation operator, h_κ is the diameter of

[7] Yano, *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*. 2012

[8] Brenner *et al.*, *The Mathematical Thoery of Finite Element Methods, Third Edition*. 2008

the element, p is the polynomial order of interpolation and C is a constant independent of h_κ and v. Thus if a solution is smooth, a higher order interpolant gives an improved asymptotic rate of convergence, however if $r < p + 1$, the rate of convergence will not increase though the constant C may decrease. The use of higher-order elements when the solution is sufficiently smooth offers a powerful way of improving the accuracy of approximation. Babuška *et al.* were the first to use higher order polynomials in the p-type Finite Element Method (FEM)[9], where instead of refining the mesh, they increased the polynomial order of approximation in elements.

Classical finite element methods consist of C^0 continuous basis functions, the resulting solution space is given by

$$\mathcal{V}_{hp}^C \equiv \{v \in C^0(\Omega) : \, v|_\kappa \in \mathbb{P}^p(\kappa), \, \forall \kappa \in \mathcal{T}\}, \tag{1.2}$$

where $\mathbb{P}^p(\kappa)$ is the space of p-th order polynomials on κ and $\mathcal{T}$ is a mesh that conforms to the domain Ω. This continuity across element boundaries means that degrees of freedom are shared between elements. The use of $\mathcal{V}_{hp}^C$ for both test and trial spaces results in the Continuous Galerkin (CG) discretization.

An alternative to CG discretization is Discontinuous Galerkin (DG) discretization, originally devised by Reed and Hill for the neutron transport equation[10]. The solution space for a typical DG discretization is given by

$$\mathcal{V}_{hp}^D \equiv \{v \in L^2(\Omega) : \, v|_\kappa \in \mathbb{P}^p(\kappa), \, \forall \kappa \in \mathcal{T}\}; \tag{1.3}$$

relaxing the requirement of continuity means that $\mathcal{V}_{hp}^D \not\subset C^0$.

The DG space is an enrichment of the continuous space, meaning on a given mesh $\|u - \pi_D u\|_{L^2(\Omega)} \leq \|u - \pi_C u\|_{L^2(\Omega)}$ where π_D and π_C are the L^2 projectors onto the corresponding spaces. The enrichment of the DG space with respect to the CG space comes from the duplication of degrees of freedom on the traces of elements as shown in figure 1.3.

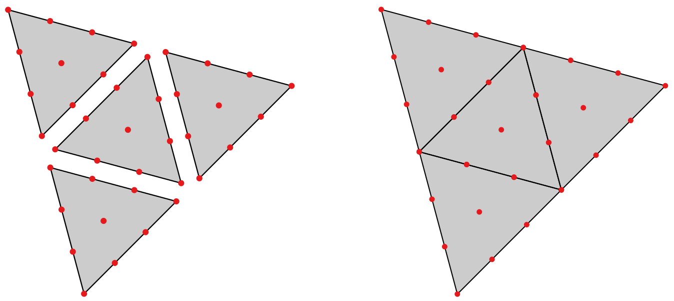

(a) DG **(b)** CG

Figure 1.3: DOF stencils for the Discontinuous Galerkin and Continuous Galerkin discretizations on Triangles.

This duplication of degrees of freedom on the traces of elements has the advantage that it enables the use of Riemann solvers and analogous jump-based methods for advective flux stabilization. Conversely,

[9] Babuska *et al.*, *The p-version of the finite element method*. 1981

[10] Reed *et al.*, *Triangular mesh methods for the neutron transport equation*. 1973

this duplication can result in a significant increase in terms of number of degrees of freedom for a given mesh, especially as the physical dimension increases. Cockburn *et al.* summarized the development of the DG discretization through the 20th century[11]. Though advective stabilization occurs naturally for the DG discretization through upwinding, additional stabilization is required for diffusive terms[12].

DG discretizations have proved popular within the Computational Fluid Dynamics (CFD) community due to their higher-order accuracy, compact stencil, elemental local conservation, stability on hyperbolic problems, and their clear relationship to upwind finite volume methods which are common in CFD[13]. DG discretization is also able to handle hanging node refinement, making it well suited to adaptive mesh refinement, and local polynomial enrichment, making it well suited to hp adaptation. However the duplication of DOFs on element traces is a significant downside to the method, which in large part motivated the development of the Hybridizable Discontinuous Galerkin (HDG) discretizations[14–16]. HDG introduces Lagrange multipliers defined on the trace of an element which weakly enforce continuity of the flux. These Lagrange multipliers decouple the interior degrees of freedom from each other, allowing for their elimination from the global solve. The interior degrees of freedom can then be calculated from these trace variables using local solvers[16]. This hybridization technique significantly reduces the size of the resulting global system, and the trace variables also enable the post-processing of the solution to increase the polynomial order of convergence[15,17,18].

HDG discretization, though drastically reducing the number of globally coupled degrees of freedom compared to DG, still suffer from duplication of degrees of freedom on the skeleton of a mesh (the trace of all the traces of a mesh). The removal of this duplication motivated the development of the Embedded Discontinuous Galerkin (EDG) discretization[19,20], which modifies the space of HDG unknowns by coupling the duplicated DOFs at the intersection of element traces[4]. This reduction in the number of globally coupled degrees of freedom comes at the cost of the local post-processing procedure and reducing the convergence rate of the numerical flux from $p+1$ to p. The resulting global EDG system exhibits significant similarities to the stabilized SUPG discretization[21], in addition to having the same number of globally coupled degrees of freedom[22].

Stabilized Continuous Galerkin Discretization

The relative DOF efficiency of the CG discretization space means that CG represent an attractive prospect when compared to DG discretization. However, the lack of trace duplication precludes the use of inter-

12. Arnold *et al.*, *Unified analysis of discontinuous Galerkin methods for elliptic problems.* 2002

13. Hartmann *et al.*, *Error estimation and adaptive mesh refinement for aerodynamic flows.* 2009

16. Cockburn *et al.*, *Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems.* 2009

[4] For instance in two dimensions, this occurs at vertices, and in three dimensions at edges and vertices.

21. Kamenetskiy, *On the relation of the Embedded Discontinuous Galerkin method to the stabilized residual-based finite element methods.* 2016

element upwinding for advective stabilization, as such CG requires advective stabilization from other sources. Some common stabilization methods include Streamline Upwind Petrov Galerkin (SUPG)[23,24], Galerkin Least Squares (GLS)[25] and Variational Multi-Scale (VMS)[26]. Each of these methods consists of adding a weighted primal strong form residual to the residual, this term takes the form

$$\mathcal{R}^{\text{stab}}(v,u) = (\overline{\mathcal{L}}(w), \tau(\mathcal{L}u - f))_{\mathcal{T}} + \mathcal{R}(v,u) \tag{1.4}$$

$$\overline{\mathcal{L}}^{\text{SUPG}}(w) = \mathcal{F}w \tag{1.5}$$

$$\overline{\mathcal{L}}^{\text{GLS}}(w) = \mathcal{L}w \tag{1.6}$$

$$\overline{\mathcal{L}}^{\text{VMS}}(w) = -\mathcal{L}^{*}w, \tag{1.7}$$

where $(\cdot,\cdot)_{\mathcal{T}} = \sum_{\kappa}(\cdot,\cdot)_{\kappa}$ are element-wise volume integrals, τ is the intrinsic timescale or stabilization parameter, $\mathcal{F}$ is the advective flux jacobian, $\mathcal{L}$ is the strong form primal operator and $\mathcal{L}^{*}$ the adjoint strong form operator. Though SUPG and GLS can be shown to be stable, they are both adjoint inconsistent for $p > 1$ which may result in a *sub optimal* rate of convergence for functional outputs. VMS is an adjoint consistent stabilization scheme, however this adjoint consistency comes at the cost of requiring careful construction of the stabilization parameter τ for $p > 1$. Simple stability analysis shows that the negative sign on the diffusive operator in $-\mathcal{L}^{*}w$ can result in a destabilizing effect and thus τ must be carefully designed to mitigate this.

The VMS approach separates the solution into a *resolved* coarse scale and an *unresolved* fine scale, it then locally models the effect of the unresolved fine scales on the coarse scales which are solved for globally. The Residual Free Bubble (RFB) method[27] is a method of stabilization where additional elemental bubble functions are introduced which strongly satisfy the PDE, and the VMS and RFB stabilization methods have been shown to be equivalent[28].

Hughes *et al.* developed the Multiscale Discontinuous Galerkin (MDG) discretization to combine the stabilization techniques of a DG discretization with the global stencil of a CG discretization, through the modeling of the unresolved scales with a discontinuous representation[29]. However, MDG introduces global coupling of the fine scales through the flux definition thereby precluding element-wise static condensation. Sangalli took a similar approach within the framework of RFB methods, deriving the Discontinuous Residual Free Bubble method (DRFB)[30]. DRFB imposes the local boundary conditions for the fine scales weakly as opposed to essentially as in VMS, thereby utilizing an elemental discontinuous test space. In DRFB, these fine scale problems are posed purely in terms of the coarse scales and data, thus they can be element-wise statically condensed.

[23] Hughes, *A simple scheme for developing upwind finite elements*. 1978

[24] Brooks *et al.*, *Streamline upwind / Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*. 1982

[25] Hughes *et al.*, *A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations*. 1989

[26] Hughes *et al.*, *Variational Multiscale Analysis: The fine-scale Green's function, projection, optimization, localization, and stabilized methods*. 2007

[27] Brezzi *et al.*, *Choosing bubbles for advection-diffusion problems*. 1994

[28] Brezzi *et al.*, $b = \int g$. 1997

[29] Hughes *et al.*, *A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method*. 2006

[30] Sangalli, *A discontinuous residual-free bubble method for advection-diffusion problems*. 2004

A Posteriori Error Estimation

A posteriori estimation aims to provide computable estimates of numerical error in the discrete approximate solutions of Partial Differential Equations (PDEs). Through localization an *a posteriori* estimate can then be used to drive mesh adaptation. Energy-based *a posteriori* error estimates have been used in the Finite Element Method since the 1980s [31],[32], particularly within the structural analysis community given the physical relevance of the energy norm. Ainsworth and Oden provided a comprehensive summary of energy norm estimates (as well as other norms) wherein they draw attention to the use of duality arguments for analyzing the estimates in an *a posteriori* context[33].

Methods based upon localized residuals were developed for mesh adaptation by Babuška and Rheinboldt[34]. Estimates based on the solution of local Dirichlet problems and local Neumann problems were also developed by Bank and Weiser[32] and Babuška and Rheinboldt[35]. Verfürth analyzed these for elliptic partial differential showing their equivalence and used a local 'bubble function' technique to prove the efficiency of the estimates[36].

Becker *et al.* presented a DG *a posteriori* error estimate for the error in the energy norm using a Helmholtz decomposition technique[37]. Carstensen *et al.*[38] developed *a posteriori* error estimates within the unified setting of Arnold *et al.*[12].

For HDG, Cockburn and Zhang[39] derived an estimate for the Poisson equation which exploited the local post-processing of HDG to reliably and efficiently estimate the error. They made use of the data oscillation term, and the difference between the additional HDG variable, $\hat{u}_h$, and the trace of the scalar variable, $u_h|_{\partial\kappa}$. This approach was extended[40] to encompass all those methods enclosed within the unified hybridization framework[16].

Output Error Estimation

In many equations, the convergence rate of the energy norm is less than that of a functional output. In a series of papers, Babuška and Miller showed that integrated functional outputs of FEM solutions could exhibit 'super convergence', converging faster than the solution error (e.g. the L^2 norm of the error)[41–43]. Barrett and Elliott analyzed the linear convection-diffusion equation and reached similar conclusions[44].

The Dual-Weighted Residual (DWR) method, developed for FEM by Becker and Rannacher, relates the primal residual error to the error in the computed functional output by weighting it with an adjoint, or dual[45]. Define the residual of a linear PDE as

[31] Babuska *et al.*, *A posteriori error analysis of finite element solutions for one-dimensional problems.* 1981

[32] Bank *et al.*, *Some a posteriori error estimators for elliptic partial differential equations.* 1985

[33] Ainsworth *et al.*, *A posteriori error estimation in finite element analysis.* 1997

[36] Verfürth, *A posteriori error estimation and adaptive mesh-refinement techniques.* 1994

[38] Carstensen *et al.*, *A unifying theory of a posteriori error control for discontinuous Galerkin FEM.* 2009

[41] Babuška *et al.*, *The post-processing approach in the finite element method—part 1: Calculation of displacements, stresses and other higher derivatives of the displacements.* 1984

[42] Babuška *et al.*, *The post-processing approach in the finite element method- part 2: The calculation of stess intensity factors.* 1984

[43] Babuška *et al.*, *The post-processing approach in the finite element method—Part 3: A posteriori error estimates and adaptive mesh selection.* 1984

[45] Becker *et al.*, *A feed-back approach to error control in finite element methods: Basic analysis and examples.* 1996

$$u \in \mathcal{V} : \ \mathcal{R}(v,u) = a(v,u) - l(v) = 0, \qquad \forall v \in \mathcal{V}, \tag{1.8}$$

where $l : \ \mathcal{V} \to \mathbb{R}$ is a linear functional and $a : \ \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ is linear in the first argument. The corresponding discretized residual is given by

$$u_{hp} \in \mathcal{V}_{hp} : \ \mathcal{R}_h(v_{hp}, u_{hp}) = a_{hp}(v_{hp}, u_{hp}) - l_{hp}(v_{hp}) \tag{1.9a}$$

$$= 0, \qquad \forall v_{hp} \in \mathcal{V}_{hp}. \tag{1.9b}$$

For an output functional $\mathcal{J} : \ \mathcal{V} \to \mathbb{R}$, the difference between the true functional output, $\mathcal{J}(u)$, and the computed functional output, $\mathcal{J}(u_{hp})$, for a primal consistent discretization, can then be related to a dual weighted residual,

$$\mathcal{E}(u_{hp}) \equiv \mathcal{J}(u_{hp}) - \mathcal{J}(u) = \mathcal{R}_h(w - v_{hp}, u_{hp}), \ \forall v_{hp} \in \mathcal{V}_{hp}. \tag{1.10}$$

The exact adjoint w comes from solving a mean value linearized problem[46]

$$w \in \mathcal{V} + \mathcal{V}_h : \ \int_0^1 \mathcal{R}'_h[u_{hp} + s(u - u_{hp})](w, v) ds \\ - \int_0^1 \mathcal{J}'[u_{hp} + s(u - u_{hp})](v) ds = 0, \ \forall v \in \mathcal{V} + \mathcal{V}_h. \tag{1.11}$$

There are two difficulties with computing the solution to equation (1.11), the first is that $\mathcal{V}$ is infinite dimensional and the second is that the mean-value linearization requires access to the true solution u. If the PDE and output functional were linear, the latter of these would not be an issue because the linearization would be exact, however the former would still remain.

Given the lack of availability of the true solution u and the infinite dimensional space $\mathcal{V}$ used in equation (1.11), a finite dimensional system linearized about u_{hp} is used instead,

$$w_{hp} \in \mathcal{V}_{hp} : \ \mathcal{R}'_h[u_{hp}](w_{hp}, v_{hp}) - \mathcal{J}'[u_{hp}](v_{hp}) = 0, \ \forall v_{hp} \in \mathcal{V}_{hp}. \tag{1.12}$$

As a consequence of Galerkin orthogonality, see equation (1.9), it follows that weighting the residual by $w_{hp} \in \mathcal{V}_{hp}$ gives no estimate, thus an approximate adjoint $\tilde{w}$ must come from a finer space such that $\mathcal{R}_h(\tilde{w}, u_{hp}) \neq 0$. A range of techniques exist that can be used for calculating $\tilde{w}$ in the literature: smooth reconstruction from w_{hp}[47,48], mesh refinement of $\mathcal{V}_{hp}$, and the technique used in this work, polynomial refinement of $\mathcal{V}_{hp}$. If $\mathcal{V}_{hp}$ is defined using piecewise elemental polynomials of order p, the adjoint is solved in the space $\mathcal{V}_{hp'}$, where $p' = p + 1$, to give $\tilde{w} = w_{hp'}$. One advantage of this technique is that it does not require an additional mesh to be maintained; however, one

[46] Becker *et al.*, *An Optimal Control Approach to A Posteriori Error Estimation in Finite Element Methods*. 2001

[47] Zienkiewicz *et al.*, *The Superconvergent Patch Recovery and* a posteriori *error estimates. Part 1: The recovery technique*. 1992

[48] Zienkiewicz *et al.*, *The Superconvergent Patch Recovery and* a posteriori *error estimates. Part 2: Error Estimates and Adaptivity*. 1992

major disadvantage to this technique is the increased memory requirement. As a consequence, the maximum size of problem that can be run using a given memory resource is more limited by the size of the adjoint problem rather than the primal problem.

For the purposes of mesh adaptation, the DWR error estimate must be localized to facilitate decisions about where mesh refinement (and coarsening) should be performed. A typical element-wise localization for the Continuous Galerkin (CG) FEM uses a strong form residual, which introduces an element boundary contribution due to gradient jumps[34,45,49]. For DG, localization by elemental restriction of the test function has been analyzed by Yano[7] for the second viscous stabilization of Bassi and Rebay (BR2)[50].

The DWR estimate and its elemental localizations for the CG, DGBR2 and LDG-H methods were analyzed and compared for the Poisson equation by the author[51]. Therein, an estimate of the error induced by the computation of the viscous stabilization for the DGBR2 method was devised, improving the asymptotic convergence rate of the global estimate. In addition, the asymptotic rates of convergence for the LDG-H estimate, and output functional, were proven and the effect of the stabilization length scale on the output convergence rate was determined.

An alternative localization of the DWR estimate for CG was shown by Richter and Wick[49], wherein they proposed to use a partition of unity based upon the linear 'hat' functions as opposed to the elemental constant modes. A major potential advantage of this approach is that it makes use of the original residual used in the primal solve, as opposed to requiring a strong form residual evaluation, which means that it inherits Galerkin orthogonality like the DG estimates.

Implicit Error Estimation

Implicit error estimates arise from the solution of auxiliary problems which use the residual evaluation as data for small local systems of equations. The additional computational cost compared to *explicit* estimation using direct residual evaluations is rewarded in the form of constant free bounds on the error, thereby providing a certificate of accuracy for the solution. Implicit estimates that furnish bounds only exists for a smaller set of PDEs than the set of PDEs which have explicit error estimates. The key feature required of a PDE in order for a bound style estimate to be possible, is coercivity.

Implicit estimates were originally developed for energy norm estimation[52] before the importance of *equilibration* was put forward by Ladevèze and Leguillon[53]. By relaxing the continuity of the CG method and imposing element local *equilibrated flux* boundary conditions, g_κ,

[7] Yano, *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*. 2012

[50] Bassi *et al.*, *GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations*. 2000

[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015

[52] Veubeke, *Displacement and equilibrium models in the finite element method*. 1965

[53] Ladeveze *et al.*, *Error estimate procedure in the finite element method and applications*. 1983

the error, $e_h \equiv u - u_h$, can be furnished with a constant free upper bound

$$\begin{aligned} a(v, e_h) &= l(v) - a(v, u_h) && (1.13) \\ &= \sum_{\kappa} \left(l_\kappa(v) - a_\kappa(v, u_h) - \langle v, g_\kappa \rangle_{\partial\kappa} \right) && (1.14) \\ &= \sum_{\kappa} a_\kappa(v, \phi_\kappa). && (1.15) \end{aligned}$$

The g_κ can be computed locally using techniques from Ladevèze and Leguillon[53] or Ainsworth and Oden[33]. The functionals $l_\kappa(v)$ and $a_\kappa(v,u)$ are restrictions to κ of $l(v)$ and $a(v,u)$ respectively. Taking Poisson with essential boundary conditions as an example ($u \in \mathcal{V}_D : (v, f)_\Omega - (\nabla v, \nabla u)_\Omega = 0, \; \forall v \in \mathcal{V}_0$) gives

$$l_\kappa(v) = (v, f)_\kappa \qquad a_\kappa(v, u) = (\nabla v, \nabla u)_\kappa. \qquad (1.16)$$

The equilibrated fluxes, $\{g_\kappa\}$, are required to satisfy the *zero-th order equilibration conditions*[54]

$$\begin{aligned} (1, f)_\kappa - a_\kappa(1, u_h) + \langle 1, g_\kappa \rangle_{\partial\kappa} = 0 && (1.17a) \\ g_\kappa + g_{\kappa'} = 0, \text{ on } \partial\kappa \cap \partial\kappa' && (1.17b) \\ g_\kappa = g, \text{ on } \partial\kappa \cap \partial\Omega. && (1.17c) \end{aligned}$$

In words, $\{g_\kappa\}$ are single valued on $\partial\mathcal{T} \equiv \cup_{\kappa \in \mathcal{T}_h} \partial\kappa$, and balance the local forcing and bilinear form, consequently they are approximations to the true flux, for instance $g_\kappa \approx -\nabla u \cdot \hat{n}_\kappa$ for Poisson. The elemental functionals and equilibrated fluxes then used to define a system for ϕ_κ,

$$\phi_\kappa \in \mathcal{V}(\kappa) : \; a_\kappa(v, \phi_\kappa) = l_\kappa(v) - a_\kappa(v, u_h) - \langle v, g_\kappa \rangle_{\partial\kappa} \; \forall v \in \mathcal{V}(\kappa). \quad (1.18)$$

The space $\mathcal{V}(\kappa)$ is an element-wise infinite dimensional localized space, and requires a finite dimensional approximation in practice. The bounding of $|||e_h|||^2_\Omega \leq \sum_\kappa |||\phi_\kappa|||^2_\kappa$ is only guaranteed for the infinite dimensional problem. The use of *equilibrated fluxes* to decompose the global error equation into local problems is a critical aspect of this type of energy norm error estimation technique. Applying these energy bounding techniques to the adjoint and the primal together, it is possible to bound the error in a computed functional output with respect to a much finer 'truth' mesh[4,55].

These ideas were extended by the work of Sauer-Budge *et al.* using complementary energy principles to give bounds with respect to the true underlying solution for Poisson[56] and Advection-Diffusion-Reaction[57]. These provided the first true certificates of the accuracy of a computed functional output with respect to the actual true solution as opposed to a 'truth mesh' solution.

33. Ainsworth *et al.*, *A posteriori error estimation in finite element analysis*. 1997

54. Ainsworth *et al.*, *A Posteriori Error Estimation in Finite Element Analysis*. 2000

4. Patera *et al.*, *A General Lagrangian Formulation for the Computation of A-Posteriori Finite Element Bounds*. 2002

55. Peraire *et al.*, *Bounds for linear-functional outputs of coercive partial differential equations: local indicators and adaptive refinement*. 1998

56. Sauer-Budge *et al.*, *Computing bounds for linear functionals of exact weak solutions to Poisson's equation*. 2004

57. Sauer-Budge *et al.*, *Computing bounds for linear functionals of exact weak solutions to the advection-diffusion-reaction equation*. 2004

One alternative to the *equilibrated flux* approach is the 'flux-free' approach of Machiels, Maday and Patera[58] where-in the calculation of equilibrated fluxes is avoided by using a partition of unity based approach. This was extended by Pares, Diez and Huerta who noted that though the equilibrated flux approach performs well, the calculation of the fluxes in $3D$ is much more complex than $2D$[59]. In common with the *equilibrated flux* approach, this energy bound procedure can be used on the primal and adjoint to arrive at linear functional bounds, again with respect to a fine 'truth mesh' solution.

The discontinuous solution space of DG allows for the closed-form calculation of *equilibrated fluxes*. The average of the viscous flux computed with the left and right states, combined with the upwinded advective flux forms an equilibrated flux. For DG methods, $\chi_\kappa(x) = \mathbb{1}(x \in \kappa)$ is contained within $\mathcal{V}_{hp}$ and consequently equation (1.17a) is satisfied as part of the global solve. Thus the zero-th order equilibration conditions are exactly equivalent to elemental flux conservation, which is a key property of a DG discretization. Wong made use of the fact that DG fluxes are equilibrated by construction to develop functional bounds for the Local Discontinuous Galerkin discretization[60].

[60] Wong, *A-Posteriori Bounds on Linear Functionals of Coercive 2nd Order PDEs Using Discontinuous Galerkin Methods*. 2006

Anisotropic Adaptation

Physical features seen in simulations often exhibit strong directionality, for example boundary layers or shock waves. One method of anisotropic adaptive refinement is hierarchical sub-division of quadrilateral meshes using hanging nodes[61–63]. Some advantages of this approach are the simplicity of advancing from one mesh to the next and the nested structure, however the main disadvantage is that the orientation of the final mesh is dictated by the initial mesh, which can result in over refinement if flow features are not aligned with the initial mesh.

The state of the art in anisotropic adaptive meshing is the use of the Riemannian metric field, $\mathcal{M} : \Omega \to \mathrm{Sym}_d^+$, which encodes direction and size[64–66]. The space Sym_d^+ is defined as

$$\mathrm{Sym}_d^+ \equiv \{A \in \mathbb{R}^{d\times d} : A_{ij} \equiv A_{ji},\ \mathbf{v}^T A\mathbf{v} > 0,\ \forall \mathbf{v} \in \mathbb{R}^d\}. \tag{1.19}$$

[64] Loseille *et al.*, *Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error*. 2011

[65] Loseille *et al.*, *Continuous Mesh Framework Part II: Validations and Applications*. 2011

[66] Alauzet *et al.*, *A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics*. 2016

A metric field is composed of smoothly varying symmetric positive definite $d \times d$ matrices, which can be used to compute the length of a line segment, $\mathbf{b} - \mathbf{a}$, by integrating along it,

$$l_{\mathcal{M}}(\mathbf{b}-\mathbf{a}) = \int_0^1 \sqrt{(\mathbf{b}-\mathbf{a})^T \mathcal{M}(\mathbf{a}+t(\mathbf{b}-\mathbf{a}))(\mathbf{b}-\mathbf{a})}\, dt. \tag{1.20}$$

The metric field, $\mathcal{M}$, corresponding to a mesh, $\mathcal{T}$, is such that all the edges of the mesh are approximately unit length under the metric,

$$l_{\mathcal{M}}(e) \equiv l_{\mathcal{M}}(\mathbf{e}) \approx 1,\ \forall e \in \mathcal{E}(\mathcal{T}), \tag{1.21}$$

where $\mathbf{e}$ is the vector defined by e_0 and e_1, respectively the start and end nodes of the edge e. Loseille and Alauzet relaxed this requirement to *quasi-unity*, $l_{\mathcal{M}}(e) \in [\frac{1}{\sqrt{2}}, \sqrt{2}]$, $\forall e \in \mathcal{E}(\mathcal{T})$.

To generate a mesh given a metric field, we make use of a metric-mesher. Some notable metric-meshers include EPIC[67], BAMG[68] FeFlo.a[69], avro[70], refine[5] and Omega.h[71]. For a more complete list and the history of the development of the field Alauzet and Loseille provide a review[66].

One anisotropic adaptive algorithm that makes use of the Riemannian metric is the Mesh Optimization via Error Sample and Synthesis (MOESS) algorithm devised by Yano and Darmofal[72] which has had considerable success in practice[72–74].

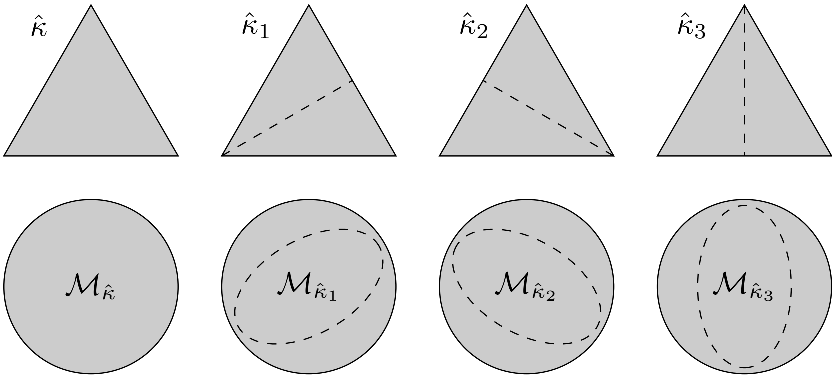

$\hat{\kappa}$
$\hat{\kappa}_1$
$\hat{\kappa}_2$
$\hat{\kappa}_3$
$\mathcal{M}_{\hat{\kappa}}$
$\mathcal{M}_{\hat{\kappa}_1}$
$\mathcal{M}_{\hat{\kappa}_2}$
$\mathcal{M}_{\hat{\kappa}_3}$

Figure 1.4: Local subdivision configurations, and their corresponding metrics in the MOESS process

MOESS is constructed around surrogate models for the sensitivity of global output error to local mesh perturbations. These mesh perturbations are encoded using the metric field as shown in figure 1.4. Local models are fit to data obtained by solving local problems based on subdivisions of individual elements of simplicial meshes.

An alternative approach to metric adaptation is through control of interpolation error measured in various L^p norms, which was placed on a theoretical foundation by the work of Loseille and Alauzet[64,65]. For a linear solution representation, this consists of constructing *a priori* error estimates for the interpolation error in terms of the solution Hessian. The Hessian is then used to size the metric along with providing orientation and anisotropy information. This approach is typically extended to systems of equations by reconstructing the Hessian of a chosen scalar function, which is often a derived quantity of the solution, such as Mach number[75].

Dolejší derived hp error estimates for interpolation error in the broken H^1-seminorm and used them to drive hp metric adaptation[76]. These error estimates are written in terms of the size and anisotropy of the metric field. Mesh adaptation can then be driven by control-

67. Michal *et al.*, *Anisotropic Mesh Adaptation through Edge Primitive Operations*. 2012

70. Caplan, *Four-dimensional Anisotropic Mesh Adaptation for Spacetime Numerical Simulations*. 2019

72. Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

64. Loseille *et al.*, *Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error*. 2011

65. Loseille *et al.*, *Continuous Mesh Framework Part II: Validations and Applications*. 2011

76. Dolejší, *Anisotropic hp-adaptive method based on interpolation error estimates in the H 1-seminorm*. 2015

---

[5] https://github.com/NASA/refine

ling these *a posteriori* error estimates, where the metric anisotropy is optimized locally before the density is optimized globally[77,78]. An additional benefit of this approach is the ability to include polynomial enrichment within the metric-adaptation framework, through the computation of a *polynomial distribution*.

Proof of Convergence for Adaptive Finite Element Methods

A mesh adaption algorithm can be said to converge if in the limit of infinite cost, the error is driven to zero. Proofs of convergence for Adaptive Finite Element Methods (AFEM) have been predominantly based around the concept of nested meshes, beginning with the treatment of the one-dimensional situation by Babuška and Vogelius[79]. Defining convergence as

$$\lim_{n\to\infty} |||u - u_n||| \to 0, \tag{1.22}$$

where $|||\cdot|||$ is the energy norm for the finite element method. They proved under mild assumptions that an AFEM based on nested refinement must converge to zero error. Their feedback approach of solving on a mesh, estimating the error, marking elements and the refining them is depicted in figure 1.1.

Dörfler extended the work of Babuška and Vogelius[79], proving monotone convergence of an AFEM for Poisson's equation in $2D$, utilizing an improved marking strategy[80]. This was done under the assumption that an initial mesh, $\mathcal{T}_H$, has sufficient fineness, μ, with respect to a user defined target error tolerance, ϵ. A mesh is said to have fineness μ if for forcing function f and boundary data g,

$$\max\left(w_1\|f - f_h\|_{L^2(\Omega)}, w_2\|hf_h\|_{L^2(\Omega)}, w_3\|g - g_h\|_{L^2(\Gamma_N)}\right) \leq \mu\epsilon, \tag{1.23}$$

where $h = \max_{\kappa(\mathcal{T}_h)} h_\kappa$ is the largest element diameter. w_1, w_2 and w_3 are positive weights introduced to balance the contribution between the 3 terms, f_h and g_h are polynomial representations of the data f and g in the basis $\mathcal{V}_h$ and Γ_N are any Neumann boundaries. If f and g can be exactly represented by $\mathcal{V}_h$, $\|f - f_h\|_{L^2(\Omega)} = 0$ and $\|g - g_h\|_{L^2(\partial\Omega)} = 0$. Equation (1.23) must hold for any nested refinement such that $\mathcal{V}_{hp}(\mathcal{T}_h) \supset \mathcal{V}(\mathcal{T}_H)_{hp}$.

A collection of elements $K \subset \mathcal{T}_H$ is then chosen such that

$$\sqrt{\sum_{\kappa\in K} \eta_\kappa^2} \geq (1-\theta)\sqrt{\sum_{\kappa\in\mathcal{T}_H} \eta_\kappa^2}, \tag{1.24}$$

for a specified $\theta \in (0,1)$, where η_κ is the standard localized strong form error indicator[54]. If the set K is then refined such that all edges of $\kappa \in K$ are split, the refined mesh $\mathcal{T}_h$ satisfies

$$\exists C \in (0,1): \ |||e_h|||_{\mathcal{T}_h} \leq C|||e_H|||_{\mathcal{T}_H} \tag{1.25}$$

[79] Babuška *et al.*, *Feedback and adaptive finite element solution of one-dimensional boundary value problems*. 1984

[80] Dörfler, *A convergent adaptive algorithm for Poisson's equation*. 1996

or the error has reached the initially specified tolerance, $|||e_h|||_{\mathcal{T}_h} \leq \epsilon$. Given the initial assumption on μ and this contraction, the solution converges in a finite number of steps.

The work of Dörfler was extended by Morin, Nochetto and Siebert who relaxed the initial assumption on the mesh fineness[81]. Their results are restricted to the second order, linear elliptic equation $-\nabla \cdot (A\nabla u) = f$ where A is piecewise constant and positive definite. They introduced the concept of data oscillation

$$\operatorname{osc}(f, \mathcal{T}_H) = \left(\sum_{\kappa \in \mathcal{T}_H} \| H(f - f_\kappa) \|^2_{L^2(\kappa)} \right)^{\frac{1}{2}} \tag{1.26}$$

where f is the forcing, H is a piecewise constant local meshsize and f_κ is the mean value of the forcing on κ. They showed that by controlling this Dörfler's assumption on mesh fineness could be relaxed; however all refinements were required to introduce an interior node to any divided element.

Binev *et al.* introduced a coarsening stage and thereby proved using nonlinear approximation theory that the optimal asymptotic accuracy is recovered for linear solutions on linear meshes[82]. Cascon *et al.* relaxed the interior node property of Morin *et al.*[81], that had been prohibitive in practice, and proved convergence for a general self-adjoint second-order elliptic PDE[83].

Carstensen *et al.* combined the literature to arrive at the axioms of adaptivity which, when satisfied, prove the convergence (and quasi-optimal convergence) of an adaptive finite element algorithm, independent of whether the PDE is linear, and the discretization used[84]. For a description of the axioms, refer to appendix D.1. This axiomatic approach was applied to output adaptation by Feischl *et al.*[85] for standard discretizations of second order linear elliptic PDEs by controlling the error in the adjoint and the primal simultaneously. The axioms of adaptivity provide an abstract framework for the proof of convergence, which in its abstraction can be applied to a wide range of estimates, PDEs and algorithms.

Contributions

The major contributions of this work are divided amongst the three chapters:

Chapter 2 Development of a MOESS algorithm for continuous discretizations. This consists of the development of a vertex defined error model, an edge focused local modeling procedure and the development of boundary conditions for the local solve domain. Demonstration of the performance of the algorithm in comparison to the

[81] Morin *et al.*, *Data oscillation and convergence of adaptive FEM*. 2000

[84] Carstensen *et al.*, *Axioms of adaptivity*. 2014

original MOESS algorithm for DG discretization using L^2 projection in two dimensions and the linear Advection Diffusion PDE in three dimensions.

Chapter 3 Development of a proof of convergence for a metric-based mesh adaptation algorithm. The framework does not rely upon a nested structure for the sequence of generated meshes and enables the proof of the optimal asymptotic order of convergence, independent of the regularity of the problem.

Chapter 4 Application of the MOESS algorithm for continuous discretizations to a three dimensional Navier Stokes simulation using a new stabilized finite element method: the Variational Multi-Scale with Discontinuous subscales (VMSD) method. Demonstration of the expected DOF efficiency of the continuous discretization and the asymptotic rates of convergence predicted by the proof of Chapter 3.

Outline

This thesis consists in large part of two self contained papers that make up Chapters 2 and 3. Chapter 2 details the extension of the MOESS algorithm to the Continuous Galerkin (CG) discretization, and compares it to the original DG algorithm for linear test problems. Chapter 3 looks at the mathematical basis of the MOESS algorithm, in particular a proof of convergence and rate of convergence for vertex defined error models of the type introduced in Chapter 2. In Chapter 4 we demonstrate the performance of the algorithm developed in Chapter 2 to a $3D$ Navier-Stokes simulation utilizing a new stabilization, achieving the theoretical performance detailed in Chapter 3. Finally, conclusions and future work are detailed in Chapter 5.

CHAPTER 2

MESH OPTIMIZATION VIA ERROR SAMPLING AND SYNTHESIS FOR CONTINUOUS GALERKIN DISCRETIZATION

Abstract

Anisotropic output-based mesh adaptivity is a powerful technique for controlling the output error of finite element simulations, particularly when used in conjunction with higher-order discretization. The Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm makes use of the *continuous mesh model* which encodes local mesh sizing and anisotropy in a Riemannian metric field, and was developed for Discontinuous Galerkin discretization. In this chapter we outline an extension of the MOESS algorithm for discretization which are defined by basis functions that are continuous across element boundaries. Error models are defined in terms of local error estimates defined at vertices, and local solutions are computed on local patches defined in terms of the edges of the mesh. The resulting algorithm displays reduced error for the same number of degrees of freedom compared to the MOESS algorithm for DG discretization, as illustrated by some numerical examples for L^2 projection and linear advection diffusion.

2.1 *Introduction*

Modern computational architectures have given rise to a drastic increase in the complexity of numerical simulations performed on a regular basis. As the complexity of these simulations has increased, the requirement to control errors within these simulations has increased commensurately. Output-based estimation enables the quantifiable control of error in measures more useful to scientific and engineering applications than typical residual norms. For instance the drag or heat flux on a boundary, or the average temperature or species consumption within a volume.

The Dual-Weighted Residual (DWR) framework developed for the Galerkin Finite Element Method (FEM) by Becker and Rannacher[45] provides a localizable estimate of the error in these quantities. The DWR estimate has been analyzed for other discretization methods by Carson *et al.*[51] and extended to other localization methods by Richter and Wick[49]. Another approach to output error estimation within FEM is the implicit estimation approach. Patera and Peraire utilize the coercivity of PDEs to bound the output with respect to the solution on a 'truth' mesh[4]. This technique was extended by Sauer-Budge *et al.* to bounding the computed output with respect to the true output[56,57].

An initial approach to anisotropic adaptive algorithms for simplices used the Hessian of the solution, which controls linear interpolation error[86,87]. For systems of equations the HEssian of a scalar derived quantity is used, for instance the Mach number for fluid dynamic simulations. This approach was codified in the work of Loseille and Alauzet[64,65] who introduce the concept of mesh-metric duality and provide interpolation error estimates. Alauzet and Loseille give a comprehensive review of the development of anisotropic adaptivity[66].

Various approaches for controlling output error have been proposed: Venditti and Darmofal constructed a Hessian of a scalar component of the solution[88]. Fidkowski and Darmofal[89] and Leicht and Hartmann[90] extended this approach to higher order discretizations and Formaggia *et al.* used the Hessian of the dual[91,92]. Though successful these methods exhibited shortcomings, with the anisotropy detection being restricted to either primal or dual features alone, or only single components of multi-variable systems.

Driving adaptive decisions using local solves has been done by Houston *et al.* [93] and Ceze and Fidkowski[61] for quad-based meshes utilizing hanging nodes. Richter[94] and Leicht and Hartmann[95] used the quad/hex structure to pose discrete optimization problems. Yano and Darmofal used a local sampling process on simplex meshes to develop the MOESS algorithm[72]. MOESS consists of building surrogate models for the effect of local mesh perturbations on the global error via local sampling, then optimizing these models before re-meshing. Fidkowski[74] proposed an alternative to the local sampling procedure of Yano and Darmofal, using projection of the higher-order adjoint between locally divided meshes to simulate local sampling.

The local sampling process of MOESS was developed particularly for the Discontinuous Galerkin (DG) discretization, though it can be naturally extended to any discretization with an elementally localiz-

[45] Becker *et al.*, *A feed-back approach to error control in finite element methods: Basic analysis and examples*. 1996

[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015

[56] Sauer-Budge *et al.*, *Computing bounds for linear functionals of exact weak solutions to Poisson's equation*. 2004

[57] Sauer-Budge *et al.*, *Computing bounds for linear functionals of exact weak solutions to the advection-diffusion-reaction equation*. 2004

[64] Loseille *et al.*, *Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error*. 2011

[65] Loseille *et al.*, *Continuous Mesh Framework Part II: Validations and Applications*. 2011

[93] Houston *et al.*, *Adaptivity and A Posteriori Error Estimation for DG Methods on Anisotropic Meshes*. 2006

[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

able estimate and discontinuous basis functions. One shortcoming of the DG discretization is the large number of degrees of freedom that come from duplication on element boundaries, which contrasts with the classical Continuous Galerkin (CG) discretization where basis functions are shared by adjacent elements. These shared degrees of freedom though efficient, prevent the application of the MOESS algorithm to this class of discretizations. MOESS has been extended to the Embedded Discontinuous Galerkin (EDG) discretization where Fidkowski distributed the error associated with the continuous features to the surrounding elements[96], thus the modeling process remains fundamentally elemental in nature.

In this chapter we present a new MOESS class algorithm that works with discretizations that share degrees of freedom amongst elements, thereby enabling the design of anisotropic output adapted meshes for CG. For simplicity of exposition we consider here only the classic *unstabilized* CG discretization, though there is no fundamental difference to the algorithm when adding stabilization.

The outline of the chapter is: Section 2.2 introduces the metric optimization framework along with the notion of mesh-metric duality and the necessary machinery for manipulation of a metric field. Section 2.3 describes the MOESS algorithm for discontinuous-type discretizations. Section 2.4 demonstrates the new MOESS algorithm for continuous-type discretizations. Section 2.5 demonstrates the new algorithm in comparison to the original discontinuous-type algorithm for L^2 error control in two dimensions. Section 2.6 outlines the vertex localized Dual-Weighted Residual output error estimate. Section 2.7 compares the algorithm to the original MOESS algorithm for a three dimensional linear advection diffusion PDE.

2.2 *Metric Optimization*

The problem of finding an optimal mesh can be abstractly written as

$$\mathcal{T}^* = \underset{\mathcal{T}\in\mathbb{T}(\Omega)}{\arg\inf}\, \mathcal{E}(\mathcal{T}) \tag{2.1a}$$

$$\text{s.t} \quad \mathbb{C}(\mathcal{T}) \leq \mathbb{C} \tag{2.1b}$$

where $\mathbb{T}(\Omega)$ is the space of conforming meshes of the domain Ω, $\mathcal{E}$ is an error functional to be minimized and $\mathbb{C}$ is a cost functional that constrains the optimization. As an example $\mathcal{E}$ might be the L^2 error computed using a discretization and $\mathbb{C}$ the number of degrees of freedom in the discretization.

Discrete optimization is relatively intractable, so in order to solve equation (2.1) we consider a continuous relaxation. This relaxation makes use of the Riemannian metric field, $\mathcal{M}(x)$, which encodes in-

[96] Fidkowski, *Output-Based Mesh Optimization for the Embedded Discontinuous Galerkin Method*. 2019

formation about a mesh via mesh-metric duality[64,65]. The metric field is a smoothly varying field of symmetric positive definite tensors that encodes distance. The distance between two points $\mathbf{a}$ and $\mathbf{b}$ is given by the expression

$$r_{\mathcal{M}}(\mathbf{a}, \mathbf{b}) = \int_0^1 \sqrt{(\mathbf{b}-\mathbf{a})^T \mathcal{M}(\mathbf{a} + t(\mathbf{b}-\mathbf{a}))(\mathbf{b}-\mathbf{a})}\, dt. \tag{2.2}$$

This also defines an alternative notation for the length of a vector, $l_{\mathcal{M}}(\mathbf{v}) \equiv r_{\mathcal{M}}(\mathbf{v}_0, \mathbf{v}_1)$, where $\mathbf{v}_0, \mathbf{v}_1$ denote the start and end of $\mathbf{v}$ respectively. The length of an edge e is then denoted by $l_{\mathcal{M}}(e) \equiv l_{\mathcal{M}}(\mathbf{e}) \equiv r_{\mathcal{M}}(\mathbf{e}_0, \mathbf{e}_1)$ where $\mathbf{e}_0, \mathbf{e}_1$ denote the start and end of edge e respectively. The metric field, $\mathcal{M}(x)$, that corresponds to a mesh, $\mathcal{T}$, is such that all the edges of the mesh are approximately unit length under the metric, $l_{\mathcal{M}}(e) \approx 1$, $\forall e \in \mathcal{E}(\mathcal{T})$, where $\mathcal{E}(\mathcal{T})$ is the set of edges of $\mathcal{T}$. Loseille and Alauzet relaxed this requirement to *quasi-unity*, $l_{\mathcal{M}}(e) \in [\frac{1}{\sqrt{2}}, \sqrt{2}]$, $\forall e \in \mathcal{E}(\mathcal{T})$[64,65].

The continuous relaxation of equation (2.1) is then written

$$\mathcal{M}^* = \operatorname*{arg\,inf}_{\mathcal{M} \in \mathbb{M}(\Omega)} \; \mathcal{E}(\mathcal{M}) \tag{2.3a}$$

$$\text{s.t} \quad \mathbb{C}(\mathcal{M}) \leq \mathbb{C} \tag{2.3b}$$

where $\mathbb{M}(\Omega)$ is the space of Riemannian metrics that conform to the domain Ω, and the functionals operate on the metric rather than the mesh. In general there is not an explicit form for $\mathcal{E}(\mathcal{M})$ nor $\mathbb{C}(\mathcal{M})$ and thus they must be modeled.

The MOESS algorithm of Yano and Darmofal[72] constructs surrogate models for $\mathcal{E}(\mathcal{M})$ and $\mathbb{C}(\mathcal{M})$ through an elemental local solve process. Data from these elemental local solves are synthesized to construct local models for how local metric perturbations affect the global error and cost functionals. These models are then optimized to compute *perturbations* to the current metric so as to improve the mesh. The elemental modeling approach employed is particular to the discontinuous discretization, and will be explained in more detail in Section 2.3.

Step Matrices

In perturbing a Riemannian metric field, it is necessary to maintain the properties of symmetry and positive definiteness, $\mathcal{M}(x) \in \text{Sym}_d^+$, $\forall x$. A first choice for a perturbation might be to add $\delta\mathcal{M}$ to give $\mathcal{M} \rightarrow \mathcal{M} + \delta\mathcal{M}$, however maintaining positive definiteness would then require placing restrictions on $\delta\mathcal{M}$ that may be difficult to enforce. In-

[64] Loseille *et al.*, *Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error*. 2011

[65] Loseille *et al.*, *Continuous Mesh Framework Part II: Validations and Applications*. 2011

[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

stead we operate on $\mathcal{M}(x)$ using a *step matrix* $S(x)$ defined by

$$S(x) = \log\left(\mathcal{M}_0^{-\frac{1}{2}}(x)\mathcal{M}(x)\mathcal{M}_0^{-\frac{1}{2}}(x)\right) \tag{2.4}$$

$$\mathcal{M}(x) = \mathcal{M}_0^{\frac{1}{2}}(x)\exp(S(x))\mathcal{M}_0^{\frac{1}{2}}(x) \tag{2.5}$$

where $S(x) \in \mathrm{Sym}_d$. Under the action of a step matrix, a metric will remain positive definite, thus the requirement of enforcing that $\mathcal{M}(x)$ remain symmetric and positive definite requires only enforcing symmetry of $S(x)$. A step matrix can be decomposed using the trace operator to give $S = s\mathcal{I} + \tilde{S}$ where $\mathrm{tr}(\tilde{S}) = 0$. Volumetric changes that change the determinant of $\mathcal{M}$ are governed by $s\mathcal{I}$, whilst $\tilde{S}$ governs shape and orientation changes that leave the determinant unchanged.

Local Geometric Manipulation

The duality between mesh and metric also exists at the elemental level: any simplicial element κ has a uniquely defined elemental metric $\mathcal{M}_\kappa$ defined by

$$l_{\mathcal{M}_\kappa}(e) = \sqrt{\mathbf{e}^T\mathcal{M}_\kappa\mathbf{e}} = 1,\ \forall e \in \mathcal{E}(\kappa), \tag{2.6}$$

where $\mathbf{e}$ is the vector defined by edge e and $\mathcal{E}(\kappa)$ denotes the set of edges of κ. This element-metric pair $\{\kappa,\ \mathcal{M}_\kappa\}$ can be modified, for instance by splitting an edge of κ. The resulting pair of subdivided elements are denoted κ_j with corresponding metric $\mathcal{M}_{\kappa_j}$. The metric $\mathcal{M}_{\kappa_j}$ comes from the Affine Invariant mean[5] of the elemental metrics for the sub-elements that make up κ_j,

$$\mathcal{M}_{\kappa_j} = \underset{\mathcal{M}\in\mathrm{Sym}_d^+}{\arg\min} \sum_i \|\log(\mathcal{M}_{\kappa_{j,i}}^{-\frac{1}{2}}\mathcal{M}\mathcal{M}_{\kappa_{j,i}}^{-\frac{1}{2}})\|_F^2, \tag{2.7}$$

where $\mathcal{M}_{\kappa_{j,i}}$ denotes the i-th sub-element of κ_j. In general equation (2.7) must be solved iteratively[5], however if there are only two sub elements there exists a closed form solution

$$\mathcal{M}_{\kappa_j} = \mathcal{M}_{\kappa_{j,1}}^{\frac{1}{2}}\exp(\frac{1}{2}\log(\mathcal{M}_{\kappa_{j,1}}^{-\frac{1}{2}}\mathcal{M}_{\kappa_{j,2}}\mathcal{M}_{\kappa_{j,1}}^{-\frac{1}{2}}))\mathcal{M}_{\kappa_{j,1}}^{\frac{1}{2}}. \tag{2.8}$$

The set of elemental split configurations is denoted $\mathcal{J}$, and for each configuration, $j \in \mathcal{J}$, there is a corresponding step matrix,

$$S_{\kappa_j} = \log(\mathcal{M}_\kappa^{-\frac{1}{2}}\mathcal{M}_{\kappa_j}\mathcal{M}_\kappa^{-\frac{1}{2}}). \tag{2.9}$$

Figure 2.1 illustrates the subdivisions in two dimensions on the equilateral reference element $\hat{\kappa}$. S_{κ_j} can be computed in a stable and efficient manner using the expression $S_{\kappa_j} = Q_\kappa^T\hat{S}_jQ_\kappa$ where $\hat{S}_j = \log(\mathcal{M}_{\hat{\kappa}_j})$, $Q_\kappa = VU^T$ and $J_\kappa = U\Sigma V^T$ is the Jacobian of the transformation from $\hat{\kappa}$ to κ[72].

[5] Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

[5] Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

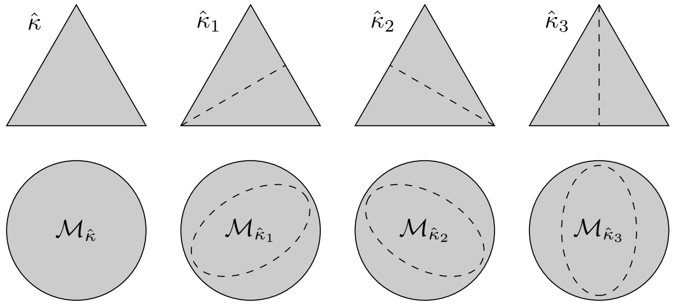

$\hat{\kappa}$
$\hat{\kappa}_1$
$\hat{\kappa}_2$
$\hat{\kappa}_3$
$\mathcal{M}_{\hat{\kappa}}$
$\mathcal{M}_{\hat{\kappa}_1}$
$\mathcal{M}_{\hat{\kappa}_2}$
$\mathcal{M}_{\hat{\kappa}_3}$

Figure 2.1: Local subdivisions of reference element $\hat{\kappa}$ into configurations $\hat{\kappa}_j$ and associated metric $\mathcal{M}_{\hat{\kappa},j}$ for $d = 2$

Implied Metric Calculation

To generate a metric conforming mesh given a metric, we make use of metric-based mesh generators, such as EPIC[67], FeFlo.a[69] and avro[70]. These metric meshers operate on a set of vertex-defined metrics $\{\mathcal{M}_v\}$. In this work, $\mathcal{M}_v$ are determined as the affine-invariant mean of the surrounding $\mathcal{M}_\kappa$,

$$\mathcal{M}_v = \underset{\mathcal{M}\in \mathrm{Sym}_d^+}{\arg\min} \sum_{\kappa(v)} \|\log\left(\mathcal{M}_\kappa^{-\frac{1}{2}}\mathcal{M}\mathcal{M}_\kappa^{-\frac{1}{2}}\right)\|_F^2, \tag{2.10}$$

where $\kappa(v)$ is the set of elements attached to v. Equation (2.10) is a non-linear optimization problem, the solution to which can be arrived at using a Newton gradient descent algorithm[5]. Given a set of vertex-defined metrics, $\{\mathcal{M}_v\}$, a perturbation can be applied using a set of vertex-defined step matrices, $\{S_v\}$. For a more complete list of metric-based mesh generators, and a description of the continuous mesh framework and the associated progress of anisotropic mesh generation we refer the reader to Alazuet and Loseille[66].

2.3 *Mesh Optimization via Error Sampling and Synthesis*

The MOESS algorithm aims to approximately solve equation (2.3) by optimizing a sum of local surrogate models. Rather than optimizing $\mathcal{M}(x)$ directly, these models are posed in terms of perturbations to the current implied metric, $\{\mathcal{M}_v\}$, in the form of a set of step matrices $\{S_v\}$. In this section we outline the formulation of these local models and the resulting optimization statement.

Cost Model

The cost functional $\mathbb{C}(\mathcal{M})$ can be localized using mesh-metric duality,

$$\mathbb{C}(\mathcal{M}) = \int_\Omega c(\mathcal{M}(x), x)dx. \tag{2.11}$$

where $c(\mathcal{M}(x), x) = c_p\sqrt{\det(\mathcal{M}(x))}$ is the *local cost kernel* and $c_p \in \mathbb{R}^+$ is a coefficient. For instance if $\mathbb{C}$ is total number of degrees of

[67] Michal *et al.*, *Anisotropic Mesh Adaptation through Edge Primitive Operations*. 2012

[69] Loseille *et al.*, *Anisotropic Adaptive Simulations in Aerodynamics*. 2010

[70] Caplan, *Four-dimensional Anisotropic Mesh Adaptation for Spacetime Numerical Simulations*. 2019

[5] Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

[66] Alauzet *et al.*, *A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics*. 2016

freedom, then c_p is the number of degrees of freedom per unit volume, measured under the metric. The value of c_p is precisely known for DG discretizations, however for continuous-type discretizations it can require modeling assumptions as described in Section 2.4.

The cost functional can be localized to an element

$$\rho_\kappa \equiv \int_\kappa c(\mathcal{M}(x);x)dx = c_p \int_\kappa \sqrt{\det(\mathcal{M}(x))}dx. \tag{2.12}$$

Define $\mathcal{M}(x)$, $\forall x \in \kappa$ as a perturbation by $S(x)$ from $\mathcal{M}_\kappa$

$$\mathcal{M}(x) = \mathcal{M}_\kappa^{\frac{1}{2}} e^{S(x)} \mathcal{M}_\kappa^{\frac{1}{2}} \tag{2.13}$$

where the step can be decomposed $S(x) = s(x)\mathcal{I} + \tilde{S}(x)$ and $\operatorname{tr}(\tilde{S}(x)) = 0$. The elemental cost is then

$$\begin{aligned} c_p \int_\kappa \sqrt{\det(\mathcal{M}(x))}dx &= \int_\kappa \sqrt{\det(\mathcal{M}_\kappa^{\frac{1}{2}} e^{s(x)\mathcal{I}+\tilde{S}(x)} \mathcal{M}_\kappa^{\frac{1}{2}})}dx && (2.14) \\ &= c_p \sqrt{\det(\mathcal{M}_\kappa)} \int_\kappa e^{\frac{s(x)d}{2}} dx && (2.15) \\ &= \rho_{\kappa_0} \frac{1}{|\kappa|} \int_\kappa e^{\frac{s(x)d}{2}} dx && (2.16) \end{aligned}$$

where $\rho_{\kappa_0} \equiv c_p \sqrt{\det \mathcal{M}_\kappa} |\kappa| = \operatorname{dof}(\hat{\kappa})$.

An approximate cost functional can then be formulated by evaluating equation (2.16). To do so requires prescribing a variation of $S(x)$ across κ. One interpolation scheme is barycentric interpolation, $S(x) \equiv \sum_v b_v(x) S_v$, where $b_v(x)$ are the barycentric coordinates for the vertices and S_v are the vertex-defined step matrices. This represents an approximation as S_v are defined relative to the vertex-defined metrics $\mathcal{M}_v$ rather than $\mathcal{M}_\kappa$. Equation (2.16) can then be evaluated using numerical quadrature. The original Yano and Darmofal work utilizes single point quadrature, whilst in this work we use $d+1$ point Gauss quadrature.

Error Locality and Local Error Modeling

To localize the error functional, we make an error locality assumption

$$\mathcal{E}(\mathcal{M}) = \int_\Omega e(\mathcal{M}(x), x)dx, \tag{2.17}$$

where $e(\mathcal{M}(x), x)$ is the *local error kernel*. This assumption is in general only valid for certain purely local functionals, such as L^2 error of an element-wise discontinuous interpolant, however it has found justification in practice[2]. We then use the set of element characteristic

[2] Yano *et al.*, *An Optimization Framework for Anisotropic Simplex Mesh Adaptation: Application to Aerodynamic Flows*. 2012

functions $\chi_\kappa(x) \equiv \mathbb{1}(x \in \kappa)$ to define the local error

$$\mathcal{E}(\mathcal{M}) = \sum_\kappa \int_\Omega \chi_\kappa e(\mathcal{M}(x), x) dx \tag{2.18a}$$

$$\epsilon_\kappa = \int_\Omega \chi_\kappa e(\mathcal{M}(x), x) dx \tag{2.18b}$$

$$\approx \int_\kappa e(\mathcal{M}_\kappa, x) dx \implies \epsilon_\kappa = \epsilon_\kappa(\mathcal{M}_\kappa), \tag{2.18c}$$

where $\mathcal{M}_\kappa$ is the elemental metric.

MOESS models the local error *indicator*, defined as $\eta_\kappa \equiv |\epsilon_\kappa|$, which avoids difficulties associated with possible cancellation effects. Given η_κ it remains to model the sensitivity to local mesh perturbations. This is done via a *local sampling* procedure that entails performing a set of edge divisions then performing an approximate solve on the resulting mesh. These *local solutions* are then used to re-compute η_κ, and a model is fit to the resulting data to give the local error indicator as a function of the step matrix, $\eta_\kappa(S_\kappa)$.

Error Indicators

The error, $\mathcal{E}$, is defined as the difference between a true output, $\mathcal{J}(u)$, and a computed output, $\mathcal{J}(u_h)$. It is problem dependent, and the means of calculating it may be discretization dependent. One example would be the square of L^2 error, for which $\mathcal{J}(u) = 0$ and $\mathcal{J}(u_h) = \int_\Omega (u - u_h)^2$. $\mathcal{E}$ could also be an estimate rather than the true error, as in the case of *a posteriori* error estimation. Given an elemental *localized error* ϵ_κ, we define a local error *indicator* $\eta_\kappa \geq |\epsilon_\kappa|$.

Local Solutions for Discontinuous Finite Element Discretization

The global solution process can be abstracted as

$$u_{hp} \in \mathcal{V}_{hp} : \quad \mathcal{R}_h(v_{hp}, u_{hp}) = 0, \quad \forall v_{hp} \in \mathcal{V}_{hp}, \tag{2.19}$$

where $\mathcal{R}_h : \mathcal{V}_{hp} \times \mathcal{V}_{hp} \to \mathbb{R}$ is the discrete residual, $\mathcal{V}_{hp}$ is a global solution space, u_{hp} is a trial function and v_{hp} is a test function. The test and trial spaces are taken to be the same, with any boundary conditions enforced weakly by the residual.

The global solution space can be interpreted as the directed sum of a finite number of elemental spaces $\mathcal{V}_{hp} \equiv \oplus_{\kappa(\mathcal{T})} \mathcal{V}_p(\kappa)$ where $\mathcal{V}_p(\kappa)$ is a p-th order polynomial space defined on κ. When an element κ is subdivided to give configuration κ_j, this defines a new global solution space $\mathcal{V}^g_{\kappa_j} \equiv \oplus_{\kappa' \in \mathcal{T} \setminus \kappa} \mathcal{V}_p(\kappa') \oplus \mathcal{V}_p(\kappa_j)$ where $\mathcal{V}_p(\kappa_j) \supset \mathcal{V}_p(\kappa)$ is the locally enriched space defined by κ_j.

The global solution computed using this enriched space is denoted $u^g_{\kappa_j}$ where we have suppressed the p subscript as we are not including

p enrichment. The resulting system is written

$$u^g_{\kappa_j} \in \mathcal{V}^g_{\kappa_j} : \quad \mathcal{R}_h(v_{\kappa_j}, u^g_{\kappa_j}) = 0, \quad \forall v_{\kappa_j} \in \mathcal{V}^g_{\kappa_j}, \tag{2.20}$$

where g denotes a global solve, and κ_j denotes the element and split-configuration.

Performing a global solve for each κ and configuration $j \in \mathcal{J}$, would be prohibitively expensive, so instead the residual is 'frozen' outside κ_j and an approximation $u_{\kappa_j} \approx u^g_{\kappa_j}$ is computed by solving the system

$$u_{\kappa_j} \in \mathcal{V}_{\kappa_j} : \ \mathcal{R}_h(v_{\kappa_j}, u_{\kappa_j}) = 0, \ \forall v_{\kappa_j} \in \mathcal{V}_{0,\kappa_j}, \tag{2.21}$$

with function spaces

$$\mathcal{V}_{\kappa_j} \equiv \{v_{\kappa_j} \in \mathcal{V}^g_{\kappa_j} : \ v_{\kappa_j}(x) = u_{hp}(x), \ \forall x \in \Omega \backslash \kappa\} \tag{2.22}$$

$$\mathcal{V}_{0,\kappa_j} \equiv \{v_{\kappa_j} \in \mathcal{V}^g_{\kappa_j} : \ v_{\kappa_j}(x) = 0, \ \forall x \in \Omega \backslash \kappa\}. \tag{2.23}$$

The choice of the elemental domain for the local solve is specific to *discontinuous discretization*, such as the Discontinuous Galerkin (DG) discretization[2]. Figure 2.2 illustrates the local solve structure on the equilateral reference element $\hat{\kappa}$ for DG. For non-linear equations we have found that a Newton-Raphson solve is sufficient to arrive at a solution because the previous global solution provides a very good initial guess for the local solve.

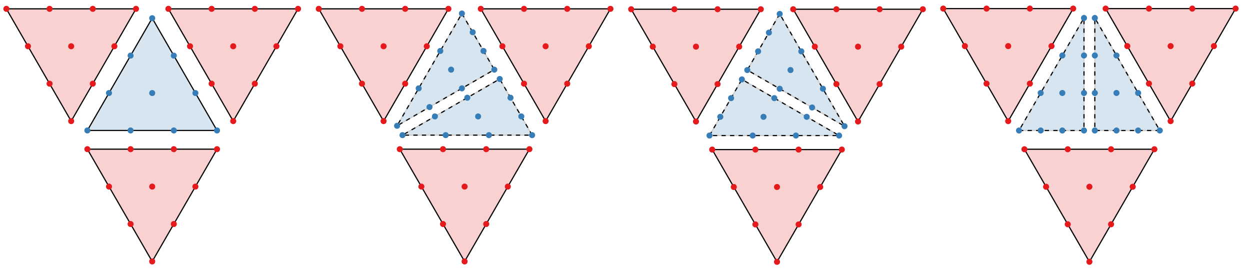

(a) $\hat{\kappa}$
(b) $\hat{\kappa}_1$
(c) $\hat{\kappa}_2$
(d) $\hat{\kappa}_3$

Figure 2.2: Local meshes used in sampling procedure for discontinuous-type MOESS for $d = 2$, $p = 3$. Red dots indicate residuals which are 'frozen' during the local solve process, whilst blue dots indicated DOFs which are calculated during the local solve

These local systems are substantially smaller than the global system, and can be computed independently of each other, exploiting parallelism. As with the error localization, this local solve is an assumption because generally $u_{\kappa_j} \neq u^g_{\kappa_j}$. The quality of the local solve is a function of the *localizability* of the error: for an elliptic equation the Green's function decays rapidly and thus there is formal justification. However, for a hyperbolic equation the interaction with *pollution* error is more complex.

Error Synthesis

A local solution u_{κ_j} for configuration κ_j is used to compute a local error indicator η_{κ_j}. This is repeated for each split configuration of an

[2] Yano *et al.*, *An Optimization Framework for Anisotropic Simplex Mesh Adaptation: Application to Aerodynamic Flows*. 2012

element to give a set $\{\eta_{\kappa_j}\}_{j\in\mathcal{J}}$, which are paired with the corresponding $\{\mathcal{M}_{\kappa_j}\}_{j\in\mathcal{J}}$. To allow for cancellation effects, when calculating η_κ the absolute value operator should be applied after the summation over $\kappa' \in \kappa_j$. For instance if $\eta_\kappa = |\epsilon_\kappa|$, then $\eta_{\kappa_j} = |\sum_{\kappa' \in \kappa_j} \epsilon_{\kappa'}|$.

A log transform is then applied to the pairs to give $\{f_{\kappa_j}, S_{\kappa_j}\}_{j\in\mathcal{J}}$ where $f_{\kappa_j} = -|\log \frac{\eta_{\kappa_j}}{\eta_\kappa}|$. These pairs are then used to fit a linear model in log space,

$$f_\kappa(S_\kappa) = \operatorname{tr}\left(R_\kappa S_\kappa\right) \tag{2.24}$$

where the *rate matrix*, $R_\kappa \in \mathrm{Sym}_d$, comes from a least squares regression

$$R_\kappa = \underset{X\in \mathrm{Sym}_d}{\arg\min} \sum_{j\in\mathcal{J}} \left(f_{\kappa_j} - \operatorname{tr}\left(X S_{\kappa_j}\right)\right)^2. \tag{2.25}$$

Additional constraints are also required of R_κ: $\exists c < 0 : c \geq r_\kappa = \operatorname{tr}(R_\kappa)$ and $\exists C_R < \infty :\ \|R_\kappa\|_F \leq C_R$. These are required to prove the convergence of a MOESS style algorithm, see Chapter 3. Given the particular values for c and C_R need not be calculated, they just need exist, it is possible to guarantee their existence via limits on the data f_{κ_j}. In particular

$$\sum_j f_{\kappa_j} < 0 \implies \exists c < 0 :\ c \geq \operatorname{tr}\left(R_\kappa\right) \tag{2.26}$$

$$|\sum_j f_{\kappa_j}| < \infty \implies \exists C_R < \infty :\ \|R_\kappa\|_F \leq C_R, \tag{2.27}$$

for the proof see Appendix A.1. Thus if $f_{\kappa_j} \in [\beta^-, \beta^+] \subset \mathbb{R}_{<0}$ then c and C_R exist. The particular values of $\beta^\pm$ can come from *a priori* analysis of the convergence rates for η_κ predicted by the isotropic model[72].

If $\mathcal{J}$ is the set of edge splits, demonstrated for two dimensions in figure 2.1, then equation (2.25) is an interpolation. Taking the exponential of equation (2.24) and introducing a regularization gives the MOESS local error model used in this work

$$\eta_\kappa(\{S_v\}) = \eta_\kappa \exp\left(\operatorname{tr}\left(R_\kappa S_\kappa\right) + \frac{\alpha \|R_\kappa\|_F}{2(d+1)} \sum_{v(\kappa)} \|S_v\|_F^2\right) \tag{2.28}$$

where $d+1 = |v(\kappa)|$ and $\alpha \leq \frac{1}{\sqrt{d}\log(2)}$ provides a regularizing effect which is required to guarantee convergence of the mesh optimization process, see Chapter 3[1].

Optimization Statement

The constructed local error and cost models are combined to give the Elemental MOESS optimization statement

[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

[1] The quadratic term results in a minima for the function, without the application of constraints. The inclusion of $\|R_\kappa\|_F$ in the coefficient ensures that distance of this minima from 0 does not grow with R_κ.

Definition 1 (Elemental MOESS optimization statement)**.**

$$\{S_v^*\} = \underset{\{S_v\} \in Sym_d}{\arg\inf} \sum_{\kappa(\mathcal{T})} \eta_\kappa e^{tr(R_\kappa S_\kappa) + \frac{\alpha \|R_\kappa\|_F}{2(d+1)} \sum_{v(\kappa)} \|S_v\|_F^2} \tag{2.29a}$$

$$s.t \quad 0 \geq \sum_{\kappa(\mathcal{T})} \rho_\kappa(\{S_v\}) - \mathbb{C} \tag{2.29b}$$

$$0 \geq C_s - tr\,(S_v), \qquad \forall v \in v(\mathcal{T}) \tag{2.29c}$$

where $S_\kappa \equiv \frac{1}{d+1} \sum_{v(\kappa)} S_v$, $\rho_\kappa(\{S_v\})$ *is the cost model for* κ, $\mathbb{C}$ *is a specified cost request, and* $C_s \in \mathbb{R}_{<0}$ *is a limit on vertex coarsening.*

This formulation of the optimization statement is different from that originally proposed by Yano and Darmofal[72], as it does not include the constraints on $|(S_v)_{i,j}|$. These have been replaced by the quadratic regularization within the local error model which limits step changes that reduce the error, and the lower bound on the trace that restricts coarsening, for more details see Chapter 3.

Definition 1 is a non-linear convex optimization problem, and thus provided there is a feasible strictly interior point then a solution exists and is unique. In this work we use the Method of Moving Asymptotes (MMA) algorithm[97], implemented within NLOPT[98], which for a convex problem is guaranteed to converge to the global minima. The optimization is solved for a new set $\{S_v\}$ from which a new metric request $\{\mathcal{M}_v\}$ is computed and then passed to the mesher. The solution is then transferred to the new mesh and a global solve performed. This process is repeated for a fixed number of iterations, or until the resulting error satisfies a prescribed tolerance.

2.4 *MOESS for Continuous Discretization*

The MOESS algorithm as outlined in Section 2.3 is generally discretization agnostic with one major exception: the local solve process. In order to differentiate this from the approach described in this section, we denote the preceding approach *discontinuous-type MOESS* or D-MOESS. This contrasts with the approach outlined in this section, *continuous-type MOESS* or C-MOESS.

To localize the error functional we use the linear 'hat' functions $\phi_v(x_{v'}) = \delta_{v,v'}$, $\forall v, v' \in V(\mathcal{T})$ as a partition of unity, motivated by the work of Richter and Wick[49]. Applying the partition of unity gives

$$\mathcal{E}(\mathcal{M}) = \sum_v \int_\Omega \phi_v e(\mathcal{M}(x), x) dx \tag{2.30a}$$

$$\epsilon_v = \int_\Omega \phi_v e(\mathcal{M}(x), x) dx \tag{2.30b}$$

$$\approx \int_\Omega \phi_v e(\mathcal{M}_v, x) dx \implies \epsilon_v = \epsilon_v(\mathcal{M}_v), \tag{2.30c}$$

[97] Svanberg, *A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations*. 2002

[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015

where $\mathcal{M}_v$ is the vertex-defined metric of equation (2.10). This abstraction in terms of localization using a partition of unity provides a possible extension to higher-order error metric models with higher order partition of unity functions.

Equation (2.30) results in a vertex defined error model we define as

$$\eta_v(S_v) = \eta_v \exp\left(\operatorname{tr}(R_v S_v) + \frac{\alpha \|R_v\|_F}{2} \|S_v\|_F^2\right), \tag{2.31}$$

where the *rate matrix*, R_v, is associated to a *vertex* error rather than an elemental error. This R_v comes from a least squares regression

$$R_v = \underset{X \in \mathrm{Sym}_d}{\arg\min} \sum_{e \in \mathcal{E}(v)} \left(f_{v_e} - \operatorname{tr}(X S_{v_e})\right)^2, \tag{2.32}$$

where $\mathcal{E}(v)$ is the set of edges that are attached to or opposite v. $\mathcal{E}(v)$ is the set of edges for which η_v will be re-computed during the local solve process. In order to guarantee convergence of the overall metric adaptation algorithm we additionally require that $\exists c \in \mathbb{R} : \; 0 > c \geq r_v$ and $\exists C_R < \infty : \; \|R_v\|_F \leq C_R$. In common with the elemental rate matrices R_κ, this can be done by imposing $f_{v_e} \in [\beta^-, \beta^+] \subset \mathbb{R}_{<0}$, see Appendix A.2.

Local Solutions for Continuous Finite Element Discretization

The computation of R_v requires log-error step pairs $\{f_{v_e}, S_{v_e}\}_{e \in \mathcal{E}(v)}$. In this section we outline the local solve process through which η_{v_e}, and subsequently f_{v_e} can be computed. The local modeling process will focus on edges of the mesh as opposed to elements. A local patch is constructed by collecting those elements attached to an edge e, giving the set $\kappa(e)$. These are then combined with the elements that complete the support of the first order basis functions for $\kappa(e)$, denoted $\kappa'(e) \equiv \{\kappa \in \mathcal{T} \backslash \kappa(e) : \; \exists \phi_i \in \{\phi_j : \; \operatorname{supp} \phi_j \cap \kappa(e) \neq \emptyset\}, \; \operatorname{supp} \phi_i \cap \kappa \neq \emptyset\}$. The patch region then consists of the union of the inner region $\kappa(e)$ and the outer region $\kappa'(e)$.

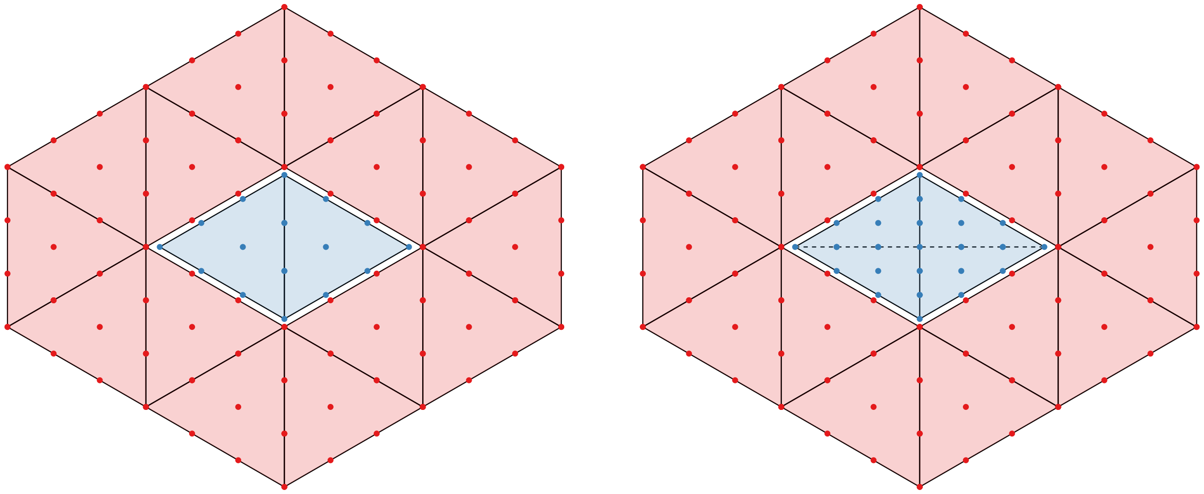
Figure 2.3 shows an example local patch in two dimensions illustrating $\kappa(e)$ and $\kappa'(e)$. It also shows the 'freezing' implied by equation (2.33), with the solution in $\kappa'(e)$ remaining fixed and the residuals in $\kappa'(e)$ being mapped to the residuals in $\kappa(e)$.

There are two main desired criteria for a local solve process with solution u_e,

1. The local problem be well-posed.

2. If e is unsplit then a local solve should return $u_e = u_{hp}$

To achieve this, we define a local problem using a jump penalization based on the previous global solution



(a) Unsplit **(b)** Split

Figure 2.3: Local meshes used in sampling procedure for continuous-type MOESS for $d = 2,\ p = 3$. Red dots indicate residuals which are 'frozen', whilst blue dots indicate DOFs which are calculated during the local solve. The blue region denotes $\kappa(e)$ whilst the red region denotes $\kappa'(e)$.

$$\mathcal{R}_{\kappa(e)}(v_e, u_e) + \langle v_e, g_{\kappa(e)} \rangle_{\partial\kappa(e)} + \langle N(v_e), u_e - u_{hp} \rangle_{\partial\kappa(e)} = 0, \qquad \forall v_e \in \mathcal{V}_e \tag{2.33}$$

where $\langle \cdot, \cdot \rangle_{\partial\kappa(e)}$ is a trace integral, $\mathcal{R}_{\kappa(e)}$ is the residual computed on $\kappa(e)$, $N(v_e) > 0$ is a Nitsche penalty function to be specified and $g_{\kappa(e)}$ is a boundary flux computed from u_{hp}. The space $\mathcal{V}_e$ is defined as

$$\mathcal{V}_e \equiv \{v_e \in C^0(\kappa(e)) : \ v_e|_\kappa \in \mathbb{P}^p(\kappa), \ \forall \kappa \in \kappa(e)\}. \tag{2.34}$$

The boundary flux $g_{\kappa(e)}$ is defined by the system

$$\mathcal{R}_{\kappa'(e)}(\bar{v}_{hp}, u_{hp}) - \langle \bar{v}_{hp}, g_{\kappa(e)} \rangle_{\partial\kappa(e)} = 0, \ \forall \bar{v}_{hp} \in \mathcal{V}_{hp}(\kappa'(e)). \tag{2.35}$$

The next step is rather than to solve for $g_{\kappa(e)}$, we rely on the fact that the traces of $\bar{\mathcal{V}}^0_{hp}(\kappa'(e))$ and $\mathcal{V}_e(\kappa'(e))$ on $\partial\kappa(e)$ have the same span. We can then eliminate $g_{\kappa(e)}$ from the first equation without explicitly computing it by mapping the residual from equation (2.35) into equation (2.33). This analytical elimination then only requires residual evaluations in $\kappa'(e)$. In the *a posteriori* error estimation literature these boundary conditions are referred to as *equilibrated fluxes*[53,99], and are used in the calculation of implicit error estimates. However, they generally employ an algorithm to explicitly compute g_κ, which could also be used directly in equation (2.33) instead.

Given $g_{\kappa(e)}$ are Neumann boundary conditions, without any additional boundary terms it is possible that a local problem may not have a unique solution. This is the case for the Poisson PDE which has a null-space consisting of constant modes. This null-space is removed by the insertion of the Nitsche-style penalty function $N(v_e)$. If e is

[53] Ladeveze *et al.*, *Error estimate procedure in the finite element method and applications.* 1983

[99] Ainsworth *et al.*, *A posteriori error estimators for second order elliptic systems part 2. An optimal order process for calculating self-equilibrating fluxes.* 1993

unsplit, then from the definition of $g_{\kappa(e)}$ it follows that $\mathcal{R}_{\kappa(e)}(v_e, u_e) + \langle v_e, g_{\kappa(e)}\rangle_{\partial\kappa(e)} = 0$. The local problem then degenerates into setting $\langle N(v_e), u_e - u_{hp}\rangle_{\partial\kappa(e)} = 0$, $\forall v_e \in \mathcal{V}_e$. Given the boundary data can be exactly represented using the trial basis, this then gives $u_e = u_{hp}$, thus the solution exists, is unique and for an unsplit patch corresponds to the global solution.

Local Error Modeling

The start and end vertices of a split edge are denoted v_1 and v_2, and the newly introduced vertex v^*. The introduction of the new vertex modifies the test functions within $\kappa(e)$. This can be seen by evaluating them at x_{v^*}, the location of v^*, for instance $\phi_{v_i}(x_{v^*}) = \frac{1}{2}$ and $\tilde{\phi}_{v_i}(x_{v^*}) = 0$, for $i = 1, 2$, where $\tilde{\phi}_{v_i}$ denotes the linear hat function defined by the vertex i of the split local patch. Thus for the start and end vertices, $\int_\Omega \phi_{v_i} e(\mathcal{M}(x); x) \neq \int_\Omega \tilde{\phi}_{v_i} e(\mathcal{M}(x); x)$.

The new linear hat functions $\tilde{\phi}_{v_i}$ are defined by $\phi_{v_i} \equiv \tilde{\phi}_{v_i} + \phi_{v_i}(x_{v^*})\phi_{v^*}$ where ϕ_{v^*} is the linear hat function associated with the new vertex. Thus to compute η_{v_i} after a split we evaluate

$$\eta_{v_i} = |\epsilon_{v_i} + \phi_{v_i}(x_{v^*})\epsilon_{v^*}|. \tag{2.36}$$

The only two vertices with $\phi_{v_i}(x_{v^*}) > 0$ are v_1 and v_2, where it is equal to $\frac{1}{2}$. The start and end vertices thus receive half each of ϵ_{v^*}, as is illustrated in figure 2.4.

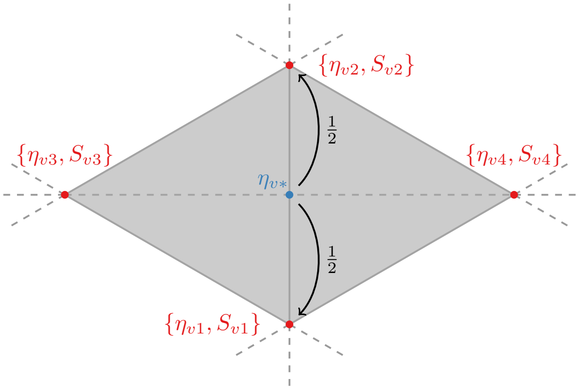


Figure 2.4: The estimate associated with new vertex v^* is equally allocated to the start and end vertices for the edge.

A local indicator η_v will be recomputed whenever an edge attached to or opposite v is split. This set of edges is denoted $\mathcal{E}(v)$, and $|\mathcal{E}(v)|$ is not known *a priori* because it is a function of the mesh topology. An equilateral tiling in two dimensions for instance, as in figure 2.3, gives $|\mathcal{E}(v)| = 12$.

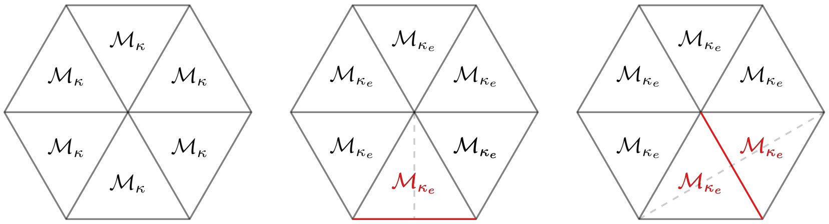
Local Geometry Modeling

The second key aspect of the construction of a MOESS local error model is the calculation of a step matrix for each split configuration. In the case of elemental step matrices, this process is non-ambiguous: $S_{\kappa_j} \equiv Q_\kappa^T \hat{S}_j Q_\kappa$ as outlined in Section 2.2. In contrast, for vertex defined step matrices it is necessary to introduce an interpolation scheme to calculate S_{v_e}, the step matrix for vertex v associated with the splitting of edge e.

First for any element attached to the edge we calculate S_{κ_e} using the same rotation of reference step matrices $\hat{S}_j$ as used to calculate S_{κ_j} in the elemental model. S_{v_e} is then calculated by taking the arithmetic mean

$$S_{v_e} = \frac{1}{|\kappa(v)|} \sum_{\kappa(v)} S_{\kappa_e}, \tag{2.37}$$

where $|\kappa(v)|$ denotes the cardinality of the set $\kappa(v)$. If an element is unmodified by the splitting of an edge, then $S_{\kappa_e} = 0$. Figure 2.5 illustrates attached and opposite edges splits for a typical two dimensional example.



(a) Unsplit **(b)** Opposite edge split **(c)** Attached edge split

Figure 2.5: Examples of opposite and attached edge splits in two dimensions. For an unsplit element $\mathcal{M}_\kappa \equiv \mathcal{M}_{\kappa_e}$, thus $S_{\kappa_e} = 0$ and red indicates those elements for which $S_{\kappa_e} \neq 0$.

Equation (2.37) represents a compromise for the sake of numerical stability. The calculation of $\mathcal{M}_{v_j}$ in order to then calculate S_v is prone to round-off error, particularly as $\mathcal{M}_v$ and $\mathcal{M}_{v_j}$ become large, which corresponds to small elements attached to v. Thus explicitly calculating the steps between $\mathcal{M}_v$ and $\mathcal{M}_{v_j}$ is numerically unstable.

Cost Model

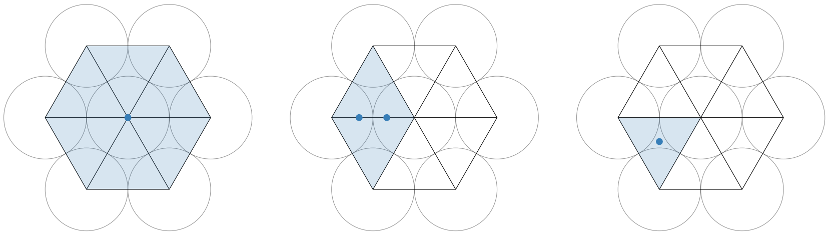
Continuous discretization presents an additional difficulty for the cost model as a consequence of the sharing of degrees of freedom across element boundaries. To resolve this we introduce *effective elemental degrees of freedom*, then outline a topology independent cost model motivated by the definition of a metric-conforming mesh. This cost model represents only an approximation to the true cost of a mesh, as it is based on an approximation of the topology for an interior vertex, and thus provides a poor approximation on boundaries. However, the number of interior vertices ought to eventually dominate the number of boundary vertices as the requested cost increases.

We define the *Effective DOF Count* (EDC) function, $d : \mathcal{V}_{hp} \to \mathbb{R}^+$, for a basis function ϕ_i as

$$d(\phi_i) = \frac{1}{|\kappa(i)|}, \tag{2.38}$$

where $\kappa(i) \equiv \{\kappa' \in \mathcal{T} : \operatorname{supp}(\phi_i) \cap \kappa' \neq \emptyset\}$. For discontinuous discretization $d(\phi_i) = 1$, $\forall i$; however, for continuous discretizations $\exists i : d(\phi_i) < 1$. In addition, the exact value of $d(\cdot)$ may vary as a function of topology, with a vertex with many attached elements having a lower value than one with fewer attached elements. Given the topology is an inherently discrete quantity, the calculated values of $d(\phi_i)$ from one mesh may have no relation to those on a successive mesh. This relationship to the discrete entity of the mesh violates the assumption made in the continuous relaxation from $\mathcal{T}$ to $\mathcal{M}$.

To address this issue we propose an *a priori* model based on the maximal packing of regular d simplices around a vertex. In two dimensions, we consider a hexagon with a vertex in the centre connected to the remaining vertices. This gives a set of 6 equilateral triangles, with the central vertex shared by 6 cells, thus a vertex basis function ϕ has $d(\phi_i) = \frac{1}{6}$, as illustrated in figure 2.6. In two dimensions an edge is a trace, thus an edge basis function has $d(\phi_i) = \frac{1}{2}$, and a cell basis function naturally has $d(\phi_i) = 1$.



(a) Vertex DOF: $d(\phi_i) = \frac{1}{6}$ **(b)** Edge DOF: $d(\phi_i) = \frac{1}{2}$ **(c)** Cell DOF: $d(\phi_i) = 1$

Figure 2.6: $d(\phi_i)$ and $\kappa(i)$ in two dimensions. The blue dots denote ϕ_i and the shaded blue region denotes the $\kappa(i)$ associated

In three dimensions a regular tetrahedral packing does not exist; however we can consider the regular convex polyhedra with a simplicial boundary, the icosahedron. Connecting each of the 20 surface triangles to the central vertex gives a set of 20 almost regular simplices (if the external edge length is 1, then the distance from a vertex to the centre is $\sin \frac{2\pi}{5} \approx 0.951$).

The number of these tetrahedra that are attached to an edge connecting a vertex of the icosahedron to the centre is 5, which gives $d(\phi_i) = \frac{1}{5}$ for edge basis functions[2]. Then in common with the two dimensional case, trace basis functions have $d(\phi_i) = \frac{1}{2}$ and cell basis functions $d(\phi_i) = 1$. These numbers and those for one dimension are summarized in table 2.1.

[2] There is an alternative value of $\frac{1}{6}$ which comes from viewing figure 2.6 as an end on view of an edge. A smaller value of $d(\phi_i)$ results in a larger requested number of elements, thus we use the value of $\frac{1}{5}$ as a more conservative estimate.

d	Vertex	Edge	Face	Volume
1	$\frac{1}{2}$	1		
2	$\frac{1}{6}$	$\frac{1}{2}$	1	
3	$\frac{1}{20}$	$\frac{1}{5}$	$\frac{1}{2}$	1

Table 2.1: Effective DOF Count for topological objects in $d \in \{1, 2, 3\}$

Table 2.1 can be used to derive formulas for the number of DOF per element, $|\mathrm{dof}(\kappa)|$, for instance in three dimensions

$$\begin{aligned}|\mathrm{dof}^{CG}(\kappa, p)| = \frac{1}{5} &+ 6\max(0, p-1)\\ &+ \max(0, (p-1)(p-2))\\ &+ \frac{1}{6}\max(0, (p-3)(p-2)(p-1))\end{aligned} \tag{2.39}$$

compared to $|\mathrm{dof}^{DG}(\kappa, p)| = \frac{(p+1)(p+2)(p+3)}{6}$. These expressions can then be used to arrive at the elemental cost model

$$\rho_\kappa(\{S_v\}) = \rho_{\kappa_0}\frac{1}{|\kappa|}\int_\kappa e^{\frac{s(x)d}{2}}dx, \tag{2.40}$$

where $\rho_{\kappa_0} \equiv c_p\sqrt{\det \mathcal{M}_\kappa}|\kappa| = \mathrm{dof}(\hat{\kappa})$.

Optimization Statement

The constructed local error and cost models are then combined to give the vertex MOESS optimization statement.

Definition 2 (Vertex MOESS optimization statement)**.**

$$\{S_v^*\} = \underset{\{S_v\}\in Sym_d}{\arg\inf} \sum_{v(\mathcal{T})} \eta_v e^{tr(R_v S_v) + \frac{\alpha\|R_v\|_F}{2}\|S_v\|_F^2} \tag{2.41a}$$

$$s.t \quad 0 \geq \sum_{\kappa\in\mathcal{T}} \rho_\kappa(\{S_v\}) - \mathbb{C} \tag{2.41b}$$

$$0 \geq C_s - tr\,(S_v), \qquad \forall v \in v(\mathcal{T}) \tag{2.41c}$$

where $\rho_\kappa(\{S_v\})$ *is the cost model for* κ, $\mathbb{C}$ *is a specified cost request, and* $C_s \in \mathbb{R}_{<0}$ *is a limit on vertex coarsening.*

2.5 L^2 Error Control

We demonstrate the ability of the C-MOESS algorithm to produce optimal meshes for the control of L^2 error, in comparison to DG/D-MOESS. The problem of L^2 error control is particularly well suited for verification as: (1) for DG discretizations the error is truly localized and (2) for D-MOESS the local solver is exact. Thus the C-MOESS algorithm can be compared to the D-MOESS algorithm in a setting in which D-MOESS should perform optimally. The metric mesher used to generate the metric conforming meshes in this section and Section 2.7 is avro[70].

The global solve can be stated as

$$u_{hp} \in \mathcal{V}_{hp} : \ (v_{hp}, u_{hp} - u)_{\mathcal{T}} = 0, \ \forall v_{hp} \in \mathcal{V}_{hp} \tag{2.42}$$

which comes from the first variation for minimizing the functional

$$\mathcal{J}(u_{hp}) = \int_{\Omega} (u_{hp} - u)^2. \tag{2.43}$$

The local error indicator is then defined as

$$\eta_i = \left| \int_{\Omega} \phi_i (u_{hp} - u)^2 \right| \tag{2.44}$$

where ϕ_i is a partition of unity function, either χ_κ for D-MOESS or ϕ_v for C-MOESS.

In L^2 projection there is not a natural choice for a boundary flux, thus we take the limit of $N(v_{e,p}) \to \infty$ in the C-MOESS local solve of equation (2.33). This corresponds to enforcing the boundary conditions on $\partial\kappa(e)$ *essentially*. This reduces the number of elements that need to be included in the computation of the residual to $\kappa(e)$, however η_v still requires evaluations within $\kappa'(e)$, though these can be precomputed.

This choice of essential boundary conditions for the local-solve patch is a design decision for a given local-solve process. It reduces the required computation to perform a set of local-solves on a given mesh however, by failing to account for solution variation in $\kappa'(e)$ it may result in poorer quality local models. Poorer quality local models generally translates into a greater number of total adaptation iterations being required to arrive at an optimal mesh.

r^α-type Corner Singularity

An r^α-type corner singularity displays singular behaviour at $r = 0$, with $\alpha > 0$ determining the strength of the singularity. Yano considered the solution to Poisson's equation on an L-shaped re-entrant corner, which has exact solution

$$u = r^\alpha \sin(\alpha(\theta + \theta_0)) \tag{2.45}$$

$$r^2 = x_1^2 + x_2^2, \ \tan(\theta) = \frac{x_2}{x_1}, \tag{2.46}$$

and derived an analytical expression for the optimal mesh spacing for this function[72], given by

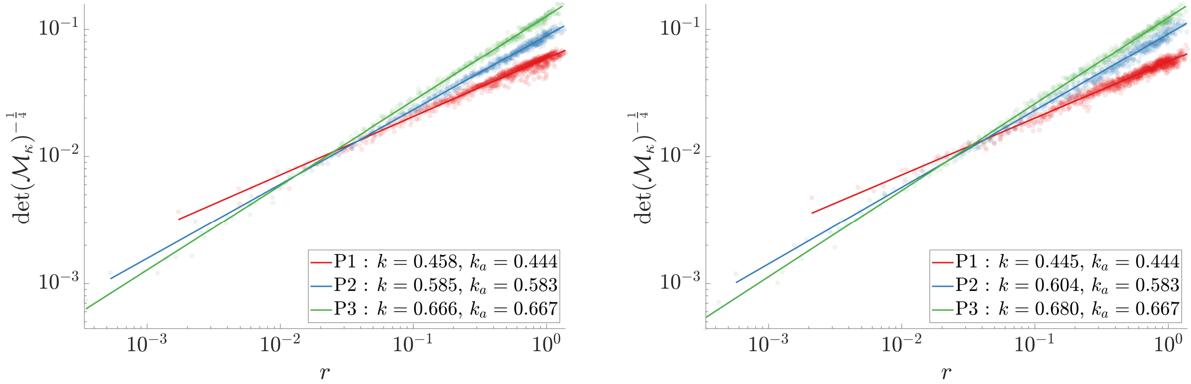
$$h = C_{\alpha p, \theta_1, \theta_2} r^{k_a}, \qquad k_a = 1 - \frac{\alpha + 1}{p + 2} \tag{2.47}$$

[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

where p is the polynomial order. This grading ought to result in an error which converges with $\mathcal{E} \sim (DOF)^{-\frac{p+1}{2}}$, see Chapter 3. The particular case we consider is $\alpha = \frac{2}{3}, \theta_0 = \frac{\pi}{2}$ and the domain is taken to be $\Omega \equiv [0,1] \times [0,1]$. The element size h is taken to be $h = \det(\mathcal{M}_\kappa)^{-\frac{1}{4}}$, which comes from noting

$$\mathcal{M}_\kappa \equiv Q_\kappa \begin{pmatrix} \frac{1}{h_1^2} & 0 \\ 0 & \frac{1}{h_2^2} \end{pmatrix} Q_\kappa^T. \tag{2.48}$$

Each adaptation is started from the same uniform 4×4 quadrilateral mesh, with each quad diagonally split into triangles and the adaptation run for 100 iterations at a fixed cost request. Figure 2.7 shows the distributions of h vs r for $p \in \{1,2,3\}$, along with the analytic mesh grading, showing good agreement between the meshes and the analytical solution. The numerical grading coefficients come from a least squares fit to the data using a linear function.



(a) D-MOESS

(b) C-MOESS

Figure 2.7: Mesh grading for the corner singularity, calculated for approximately 1333, 667 and 400 elements for $p = 1,2,3$ respectively. The dots are individual elements and the lines are from a linear regression.

Figure 2.8 shows the error versus a notional mesh scaling, $h_D \equiv (DOF)^{-\frac{1}{2}}$, where the relative DOF efficiency of C-MOESS becomes apparent when compared to D-MOESS. The results shown are the last 10 iterations out of a sequence of 50, each starting from the same initial mesh. Additionally $O(h_D^{p+1})$ convergence is observed despite the singular solution which is possible due to the graded nature of the adapted meshes.

Some adapted meshes corresponding to approximately equal number of elements for C-MOESS and D-MOESS are displayed in figure 2.9. The resulting meshes display the expected grading, with stronger grading occuring for higher polynomial degrees. A particularly striking feature is that for $p = 1$ the mesh returned by C-MOESS has a semi-structured nature. We theorize this deviation from the analysis of Yano is because the continuity introduced by the shared degrees

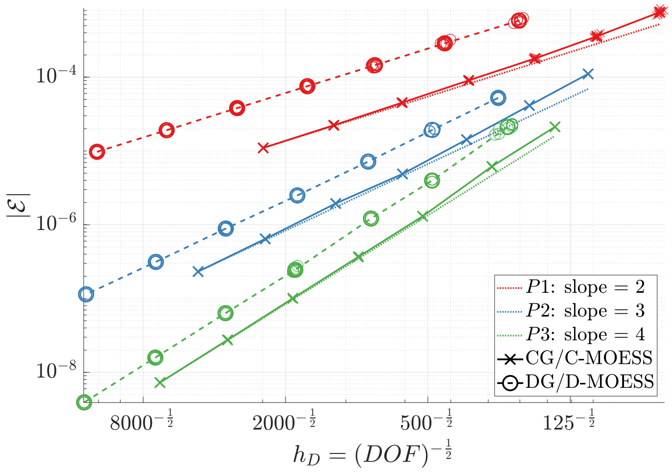


Figure 2.8: Error Convergence results for the corner singularity. Last 10 iterations are plotted, along with a curve through the mean.

of freedom violates the error locality assumption: the solution in κ is a function of $\mathcal{M}_{\kappa'}$ where $\kappa' \neq \kappa$. The effect is reduced as the polynomial order increases as the higher order degrees of freedom are shared by fewer elements.

Regularized Boundary Layer

In this section we demonstrate the anisotropic performance of the algorithm using a regularized boundary layer of the form

$$u(x_1, x_2) = e^{-\frac{x_1}{\epsilon}} + \frac{\beta}{(p+1)!} x_2^{p+1}, \tag{2.49}$$

where ϵ is the characteristic length, β is a regularization coefficient and x_1, x_2 are the coordinates perpendicular and parallel to the boundary layer. This case was shown by Yano to have an analytic solution for the optimal mesh distribution given by

$$h_1 = C_{p,\epsilon} e^{k_1^a x_1} \tag{2.50}$$

$$AR = AR_0^a e^{k_{AR}^a x_1}, \tag{2.51}$$

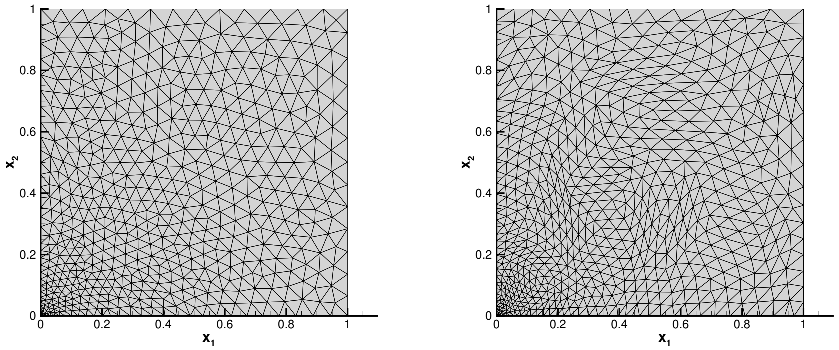
where h_1 is the spacing in the x_1 direction and $AR = \frac{h_2}{h_1}$ is the aspect ratio. The analytic solutions are given by

$$\frac{1}{k_1^a} = \epsilon \left(p + \frac{3}{2}\right) \left(1 - \frac{1}{4p^2 + 12p + 9}\right), \tag{2.52}$$

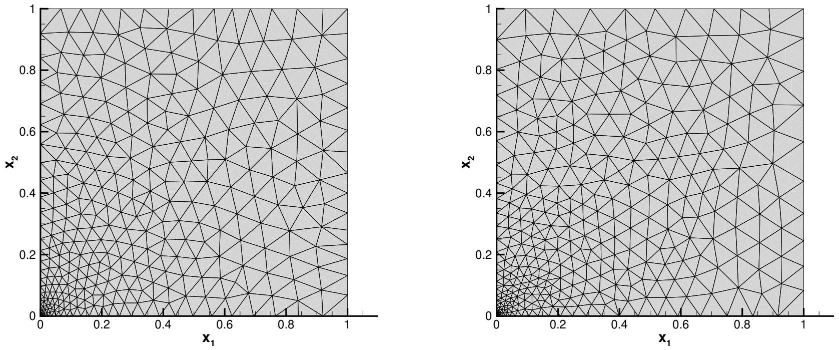
$AR_0^a = \frac{1}{\beta^{\frac{1}{p+1}} \epsilon}$ and $k_{AR}^a = -\frac{1}{\epsilon(p+1)}$[72].

We now consider the case where $\beta = 2^{p+1}$ and $\epsilon = \frac{1}{100}$, giving $AR_0^a = 50$. Figure 2.10 shows the distribution of mesh sizing in the x_1 direction, h_1, and the aspect ratio, AR, with varying x_1 location. A least squares regression is performed using the data and the resulting numerical values for AR_0, k_{AR} and k_1 are shown. The results for

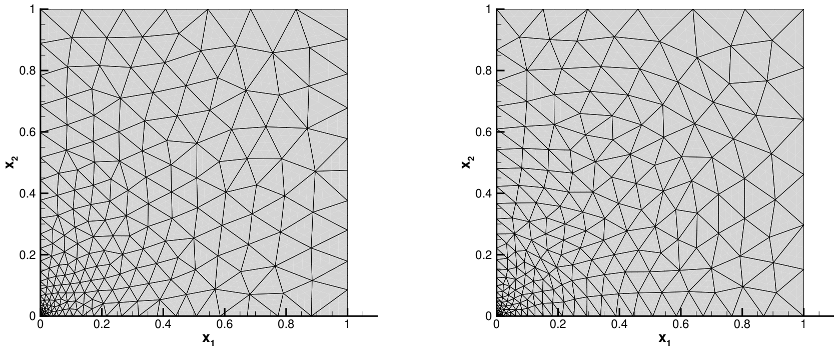
[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012



(a) D-MOESS | $p = 1$ **(b)** C-MOESS | $p = 1$

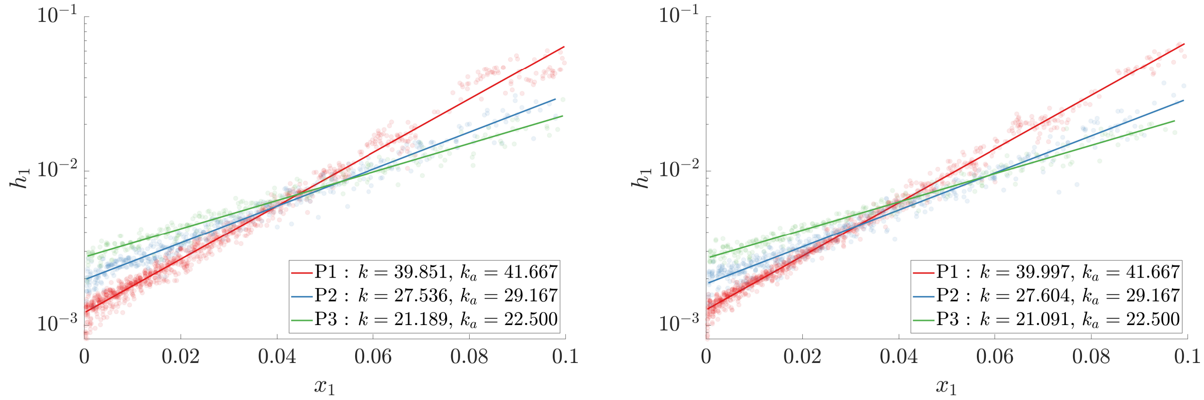


(c) D-MOESS | $p = 2$ **(d)** C-MOESS | $p = 2$



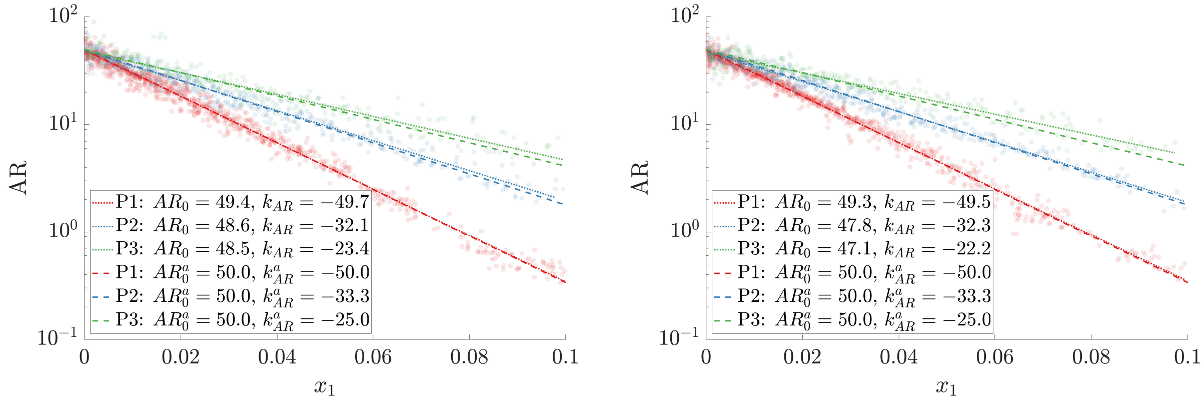
(e) D-MOESS | $p = 3$ **(f)** C-MOESS | $p = 3$

Figure 2.9: Final Meshes for the corner singularity case for approximately 1333, 667 and 400 elements for $p = 1, 2, 3$ respectively.



(a) D-MOESS - Mesh grading

(b) C-MOESS - Mesh grading



(c) D-MOESS - Aspect Ratio

(d) C-MOESS - Aspect Ratio

Figure 2.10: Mesh grading and aspect ratio for the regularized boundary layer, calculated for approximately 1333, 667 and 400 elements for $p = 1, 2, 3$ respectively. The dots are individual elements and the lines are from a linear regression.

C-MOESS are in very close agreement with the expected theoretical results, demonstrating the ability of the C-MOESS algorithm to achieve the optimal anisotropy required by the boundary layer function.

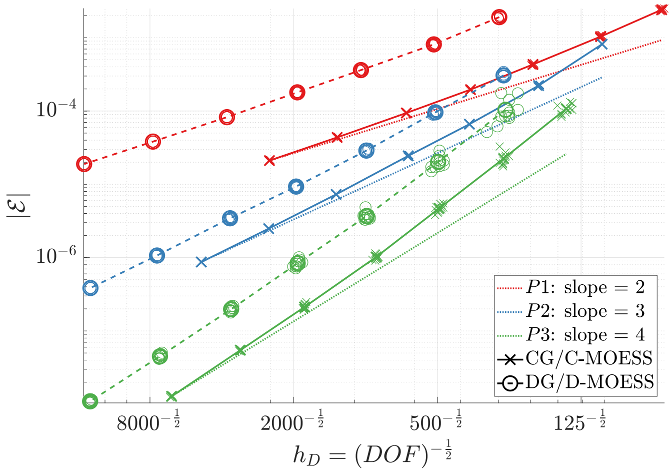


Figure 2.11: Error Convergence results for the regularized boundary layer. Last 10 iterations are plotted, along with a curve through the mean.

Figure 2.11 shows the error versus grid spacing h_D, and in common with the results for the corner singularity, C-MOESS demonstrates improved DOF efficiency when compared to the original D-MOESS algorithm. The reduction from D-MOESS to C-MOESS is largest for $p = 1$ and least for $p = 2$, this differs compared to the corner singularity case where $p = 3$ had the smallest reduction.

Figure 2.12 shows the meshes used to generate figure 2.10, demonstrating the mesh size and anisotropy grading in the x_1 direction. The $p = 2$ and $p = 3$ grids are very similar for the two algorithms, however for $p = 1$ there is a slight visual difference between the two meshes. In particular the DG/D-MOESS mesh appears slightly more regular compared to the CG/C-MOESS mesh. Similarly to the corner singularity case, we hypothesize that this is due to the non-local nature of the solution introduced by the continuity of DOFs across element interfaces. As the polynomial order increases this effect will decrease which is why for $p = 2, 3$ the resulting meshes are very similar.

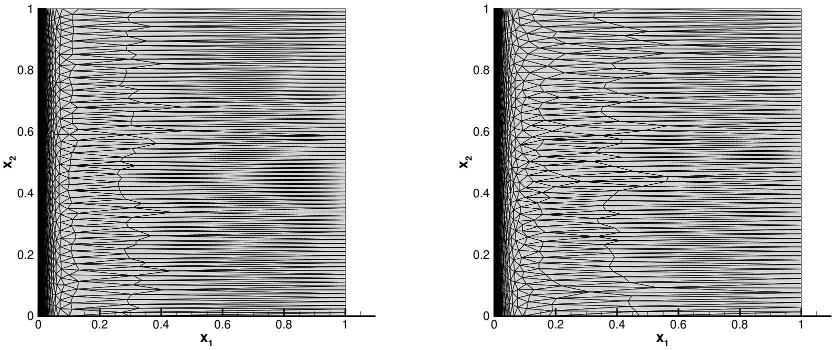
2.6 A-posteriori *output error estimation*

In this section, we demonstrate output-based adaptation using the Dual-Weighted Residual (DWR) method[45], building on the work of Richter and Wick[49]. We demonstrate it here for the unstabilized Continuous Galerkin discretization, however the method can be extended to any adjoint consistent stabilized CG method.

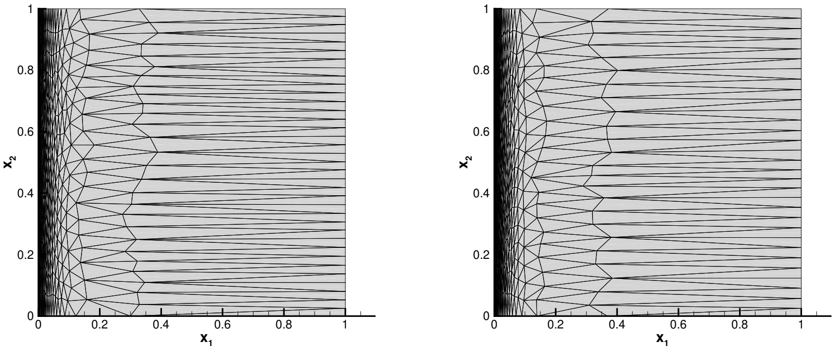
We restrict ourselves to linear output functionals of the form $\mathcal{J}(u) = \int_\Omega g(x)u + \int_{\partial\Omega} b(x)\mathbf{F}(u)\cdot\hat{n}$ and linear scalar PDEs, but the extension

[45] Becker *et al.*, *A feed-back approach to error control in finite element methods: Basic analysis and examples*. 1996

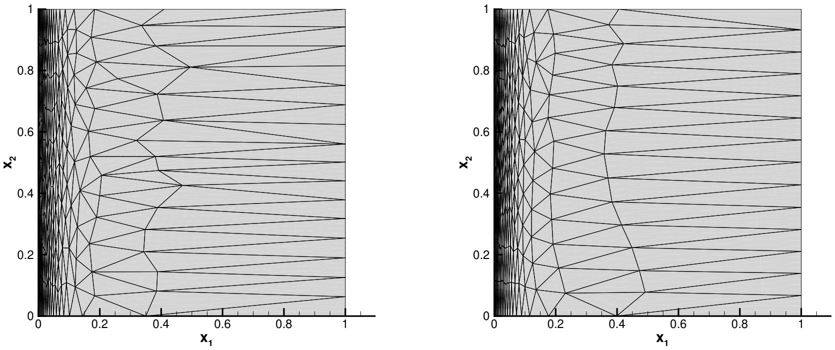
[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015



(a) D-MOESS | $p = 1$ **(b)** C-MOESS | $p = 1$



(c) D-MOESS | $p = 2$ **(d)** C-MOESS | $p = 2$



(e) D-MOESS | $p = 3$ **(f)** C-MOESS | $p = 3$

Figure 2.12: Final Meshes for the regularized boundary layer for approximately 1333, 667 and 400 elements for $p = 1, 2, 3$ respectively.

to non-linear functionals and systems of PDEs can be performed with additional technicalities[46]. The *adjoint* for a PDE is implicitly defined by the duality statement

$$\mathcal{R}(w,u) - \mathcal{J}(u) = \mathcal{R}^*(u,w) - \mathcal{J}^*(w), \tag{2.53}$$

where $\mathcal{R}^*(u,w)$ is the adjoint residual with corresponding adjoint solution, w. The primal residual can be separated into a bilinear and linear term where

$$\mathcal{R}(v,u) = a(v,u) - l(v). \tag{2.54}$$

The adjoint residual is then defined as

$$\mathcal{R}^*(v,w) = a(w,v) - \mathcal{J}(v), \tag{2.55}$$

which illustrates that the adjoint represents sensitivities of $\mathcal{J}(v)$ to residual perturbations linearized around the primal solution. Setting this residual to zero defines the adjoint solution w,

$$w \in H^1_b(\Omega) : \ \mathcal{R}^*(v,w) = 0, \ \forall v \in H^1_0(\Omega), \tag{2.56}$$

where $H^1_b(\Omega) \equiv \{v \in H^1(\Omega) : \ v(x)|_{\partial\Omega} = b\}$ and $H^1(\Omega)$ denotes the Sobolev space of functions whose first weak derivative is square integrable. We have implicitly assumed that all boundary conditions on the primal are Dirichlet. Equation (2.56) is linearized about the solution u, thus the exact adjoint PDE requires the linearization be with respect to the exact solution. This is not feasible in a discrete setting for general non-linear PDEs, as u is unavailable, however our restriction to linear PDEs and functionals means that it is valid here. For more details on the treatment of different output functionals and PDEs, we refer the reader to Becker and Rannacher[46].

Equation (2.56) has a discrete weak form

$$w_{hp} \in \mathcal{V}_{hp} : \ \mathcal{R}^*_h(v_{hp}, w_{hp}) = 0, \quad \forall v_{hp} \in \mathcal{V}_{hp}, \tag{2.57}$$

where R^*_h denotes that this is a discrete adjoint, coming from transposition of the Jacobian of the primal linear system, and the boundary conditions are enforced weakly via the residual. This can be done for an *adjoint consistent* discretization, but there are alternative approaches based around discretizing the continuous adjoint of equation (2.56) directly[100]. The standard Continuous Galerkin (CG) discretization is adjoint consistent, and this adjoint consistency gives the Functional Error Representation Formula[51]

$$\mathcal{J}(u_{hp}) - \mathcal{J}(u) = \mathcal{E} = \mathcal{R}_h(w - v_{hp}, u_{hp}), \quad \forall v_{hp} \in \mathcal{V}_{hp}, \tag{2.58}$$

which relates the error in the computed functional output, $\mathcal{J}(u_{hp})$, to the adjoint solution.

[46] Becker *et al.*, *An Optimal Control Approach to A Posteriori Error Estimation in Finite Element Methods*. 2001

Richter and Wick proposed to localize the error estimate using the partition of unity functions $\{\phi_v\}$[49],

$$\epsilon_v = \mathcal{R}_h(\phi_v(w - w_{hp}), u_{hp}), \tag{2.59}$$

where $\mathcal{R}_h(\cdot,\cdot)$ is the residual used in the global solve phase. Whilst ϵ_v is the exact localization such that $\mathcal{E} = \sum_{v \in v(\mathcal{T})} \epsilon_v$, in general the exact adjoint w is unavailable, so it must be approximated. If the dual solution came from the same space as the primal, the estimate would evaluate to zero from Galerkin orthogonality. Thus the approximation must come from a different solution space than the primal. In this work we choose the polynomial enriched space $\mathcal{V}_{hp'}$, where $p' = p + 1$, giving the higher order adjoint $w_{hp'}$. A primary advantage of this approach is that it does not require an additional mesh[45]. The requirement to explicitly subtract w_{hp} necessitates two global adjoint solves. The difference between the two adjoints, $\Delta w_{hp'} = w_{hp'} - w_{hp}$ is then transferred to the local patch[3].

2.7 Linear Advection-Diffusion (AD)

To demonstrate the performance of the algorithm we introduce the scalar linear Advection-Diffusion (AD) system in three dimensions

$$\nabla \cdot (\mathbf{a}u - \nu \nabla u) = 0, \tag{2.60}$$

where $\mathbf{a} \in \mathbb{R}^3$ is the advective velocity and $\nu \in \mathbb{R}^+$ is the viscosity. The domain of interest is defined as $\Omega \equiv [0,1] \times [0,1] \times [0,1]$, and we apply Dirichlet boundary conditions on $\partial\Omega$ taken from an exact solution

$$u(\mathbf{x}) = 1 - \prod_{i=1}^{d} \frac{1 - e^{-\frac{a_i(1-x_i)}{\nu}}}{1 - e^{-\frac{a_i}{\nu}}}. \tag{2.61}$$

As a result of this tensor product of one dimensional boundary layers, we refer to this function as the *triple boundary layer* function.

The weak form of equation (2.60) is given by

$$u \in H^1_{u_D} : \ 0 = (-\nabla v, \mathbf{a}u - \nu \nabla u)_\Omega, \quad \forall v \in H^1_0, \tag{2.62}$$

where $(v, w)_\Omega \equiv \int_\Omega vw \ dx$ is the L^2 inner product on Ω with corresponding norm $\| \cdot \|_{L^2(\Omega)}$ and $\langle v, w \rangle_\Gamma \equiv \int_\Gamma vw \ dx$ is a $d-1$ trace integral on Γ. The Continuous Galerkin (CG) discretization consists of replacing the infinite dimensional space $H^1(\Omega)$ with a finite dimensional space

$$\mathcal{V}_{hp} \equiv \{v \in C^0(\Omega) : \ v|_\kappa \in \mathbb{P}^p(\kappa), \ \forall \kappa \in \mathcal{T}\}, \tag{2.63}$$

where p denotes the polynomial order of the basis functions used in the approximation, and $\mathcal{T}$ denotes a conformal mesh of Ω, consists of

---

[3] We experimented with using the L^2 projection of $w_{hp'}$ and not explicitly subtracting off a p solution but observed a deleterious effect on the estimate and the subsequent performance of the adaptation. In addition the use of the p adjoint as an initial guess for the $p+1$ adjoint linear solve helps to reduce overall computational cost when using iterative methods.

simplicial elements κ. Integration over Ω is then replaced by integration over the finite elements $(v,w)_{\mathcal{T}} = \sum_{\kappa\in\mathcal{T}}(v,w)_\kappa$.

Dirichlet boundary conditions are implemented weakly using a Nitsche-style penalty[101]. The substitution of this space then furnishes the discretized weak form

$$\begin{aligned} u_{hp} \in \mathcal{V}_{hp} :\; 0 &= R_h(v_{hp}, u_{hp}), \quad \forall v_{hp} \in \mathcal{V}_{hp} && (2.64a)\\ R_h(v_{hp}, u_{hp}) &= (-\nabla v_{hp}, (\mathbf{a}u_{hp} - \nu\nabla u_{hp})\cdot\hat{n})_{\mathcal{T}} && (2.64b)\\ &\quad + \langle v_{hp}, \hat{\mathbf{F}}(u_{hp}, u_D) + \nu\nabla u_{hp}\cdot\hat{n}\rangle_{\partial\Omega} && (2.64c)\\ &\quad + \langle -\nu\nabla v_{hp}\cdot\hat{n} + N v_{hp}, u_{hp} - u_D\rangle_{\partial\Omega}, && (2.64d) \end{aligned}$$

where $R_h(v,w)$ is the discrete residual, $\hat{\mathbf{F}}(u_{hp}, u_D) = \frac{1}{2}\mathbf{a}\cdot\hat{n}(u_{hp}+u_D) - \frac{1}{2}|\mathbf{a}\cdot\hat{n}|(u_{hp}-u_D)$ is an upwinded advective flux and equation (2.64d) enforces dual-consistency and stability via the the Nitsche parameter, $N = \frac{8p^2\nu}{h}$[102]. The length scale, h, is given by $h = \frac{|\kappa|}{|f|}$ for face f of element κ.

The output functional considered is the boundary flux on the $x = 1$ boundary, $\mathcal{J}(u_h) = \int_{\partial\Omega_{x=1}}(\mathbf{a}u_h - \nu\nabla u_h)\cdot\hat{n}$, which gives an exact functional output

$$\mathcal{J}(u) = a_1 - \frac{a_1\exp(\frac{a_1}{\nu})\left(\exp(\frac{a_2}{\nu})(a_2-\nu)+\nu\right)\left(\exp(\frac{a_3}{\nu})(a_3-\nu)+\nu\right)}{a_2a_3(\exp(\frac{a_1}{\nu})-1)(\exp(\frac{a_2}{\nu})-1)(\exp(\frac{a_3}{\nu})-1)}. \tag{2.65}$$

The particular advection velocity and diffusivity are given by $\mathbf{a} = (\frac{8}{17}, \frac{9}{17}, \frac{12}{17})^T \in \mathbb{R}^3$ and $\nu = \frac{1}{10} \in \mathbb{R}^+$, giving a Peclet number $\mathrm{Pe} \equiv \frac{\|\mathbf{a}\|}{2L\nu} = 5$ and the exact functional $\mathcal{J}(u) \approx 0.1375831040992103$ to finite precision.

Each adaptation is started from the same uniform $3\times3\times3$ hexahedral mesh split into tetrahedra and the adaptation run for 50 iterations at a fixed cost request. Figure 2.13 shows convergence results for the sum of the error indicators $\mathcal{H} = \sum_i \eta_i$ where for CG $\eta_v = \sum_v |\mathcal{R}_h(u_{hp}, \phi_v(\psi_{h,p'} - \psi_{hp}))|$ and for DG $\eta_\kappa = |\mathcal{R}_h(u_{hp}, \psi_{h,p'}\chi_\kappa)|$. The last 10 iterations were averaged to get the trend lines and the samples are placed in a scatter plot to demonstrate the spread of the results in the averaging process. The scatter is small and thus the resulting data lie mostly on top of each other. Figure 2.13 demonstrates that C-MOESS has significantly reduced error per degree of freedom when compared with D-MOESS.

Meshes corresponding to approximately 8000, 3200 and 1600 elements for $p = 1,2,3$ respectively are shown for C-MOESS and D-MOESS in figures 2.14 and 2.15. The adaptation can be seen to focus significantly along the boundaries of the $x = 1$ face in figure 2.14 at the expense of much coarser mesh in the vicinity of the $x = 0$ face as seen in figure 2.15. The concentration of the mesh along the $x = 1$

[101] Giles *et al.*, *Algorithm developments for discrete adjoint methods*. 2003

[102] Stenberg, *On some techniques for approximating boundary conditions in the finite element method*. 1995

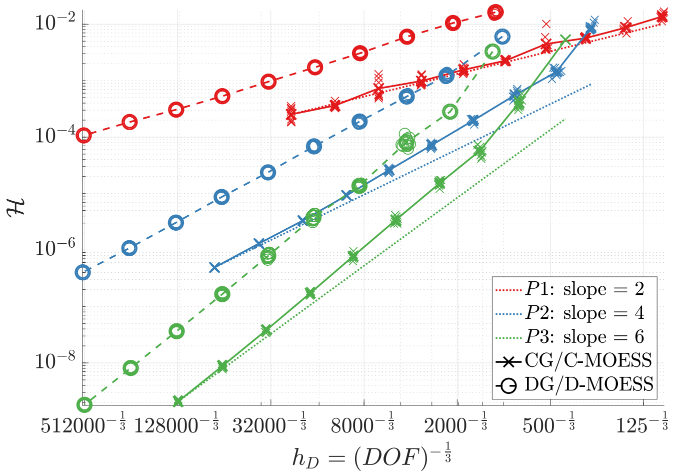
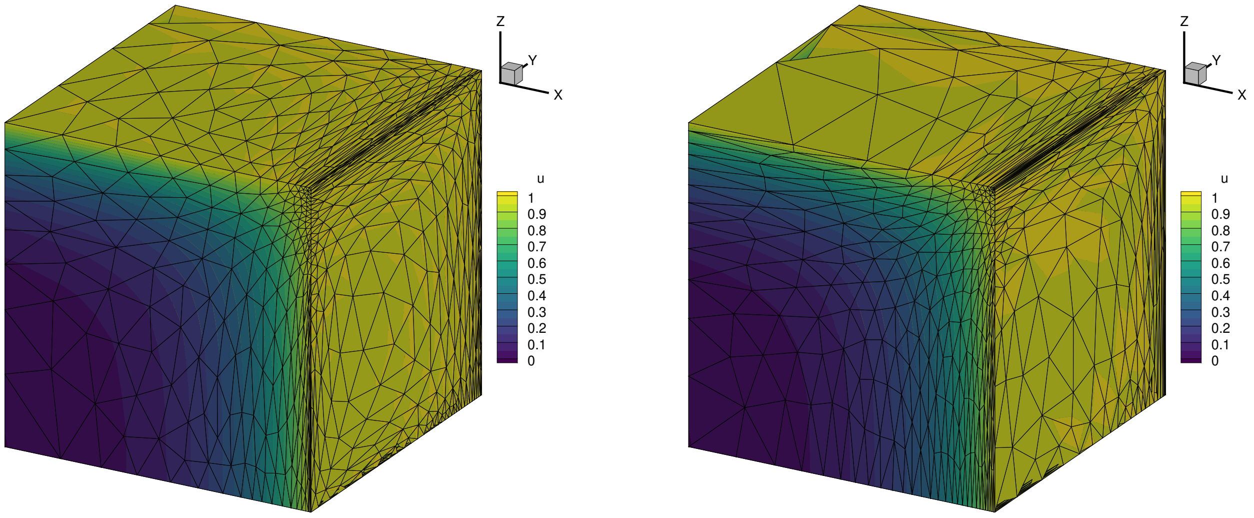


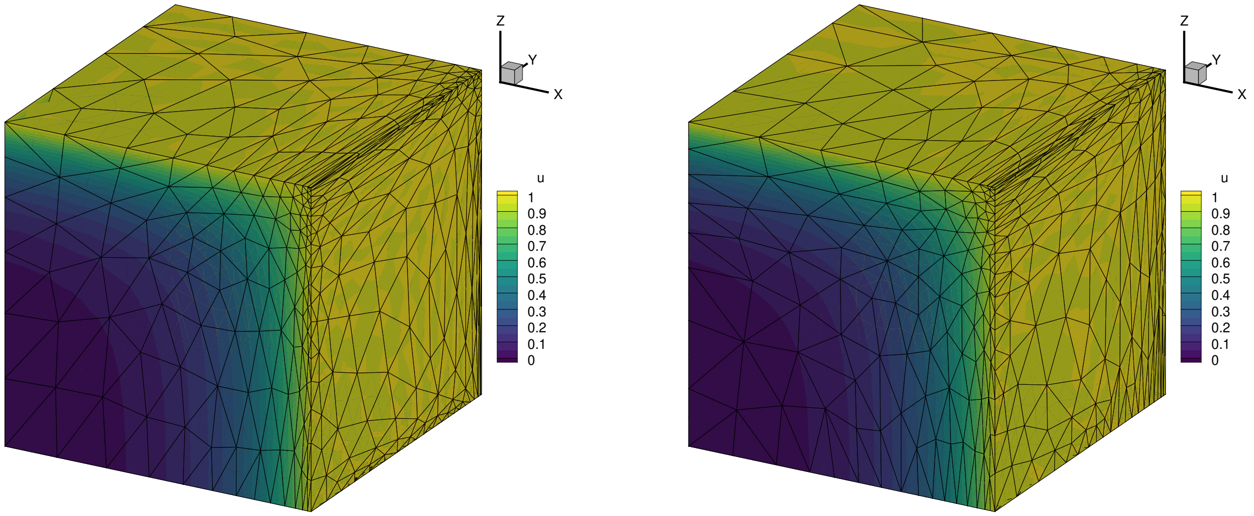
Figure 2.13: Error indicator, $\mathcal{H} = \sum_i \eta_i$, convergence results for the triple boundary layer problem. The dotted reference lines represent the expected $2p$ asymptotic rate of convergence.

face compared to the $y = 1$ and $z = 1$ faces demonstrates that the adaptation is particularly focused on those features of importance to the output functional. Adaptation driven by an error indicator for the primal residual would have focused equally on all three faces in order to resolve each of the three boundary layers.



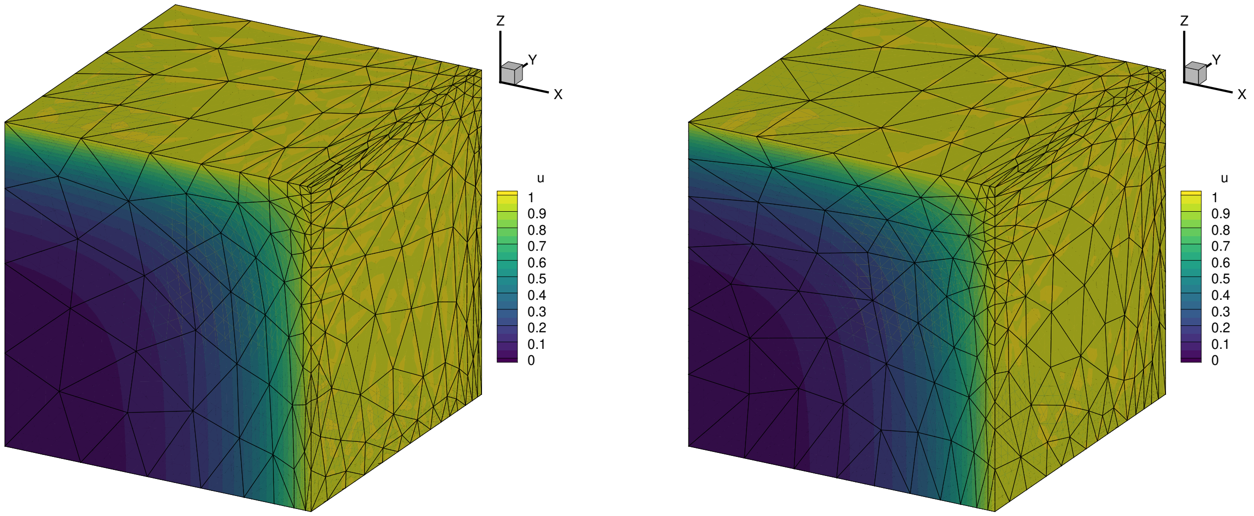
(a) DG | $p = 1$

(b) CG | $p = 1$



(c) DG | $p = 2$

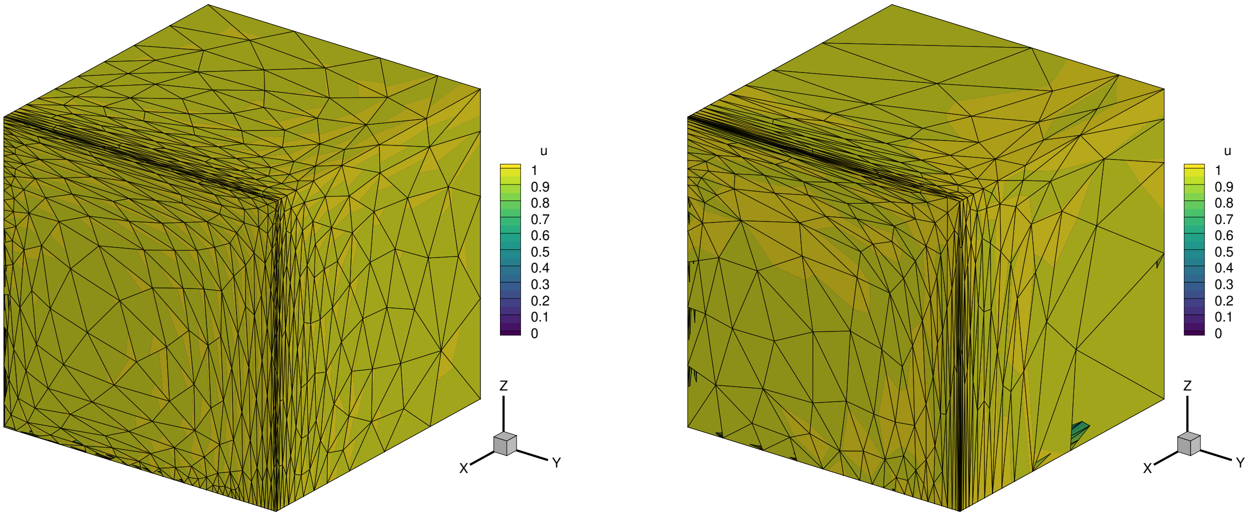
(d) CG | $p = 2$



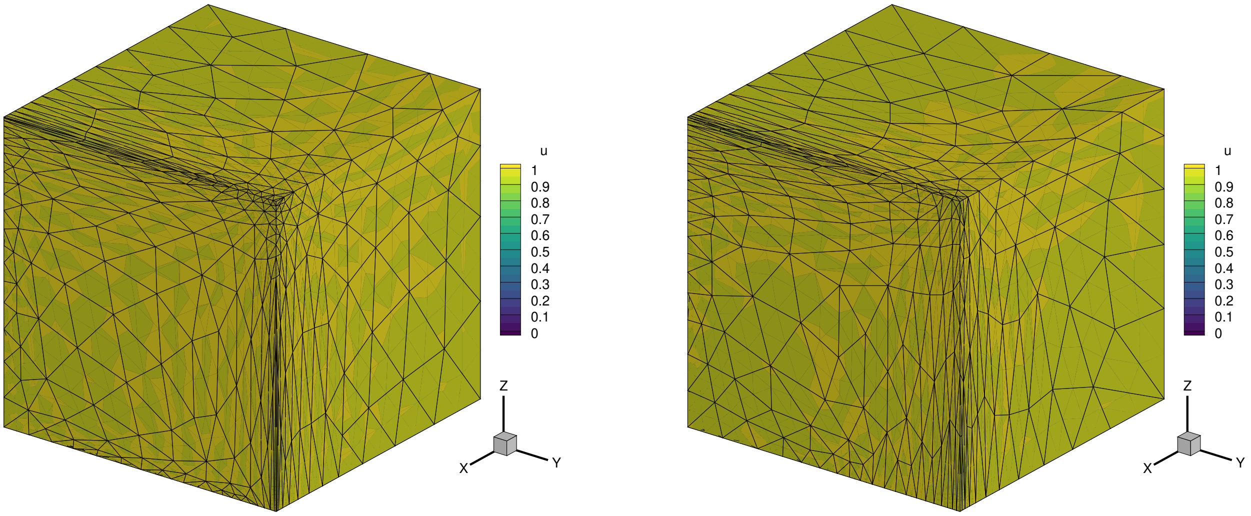
(e) DG | $p = 3$

(f) CG | $p = 3$

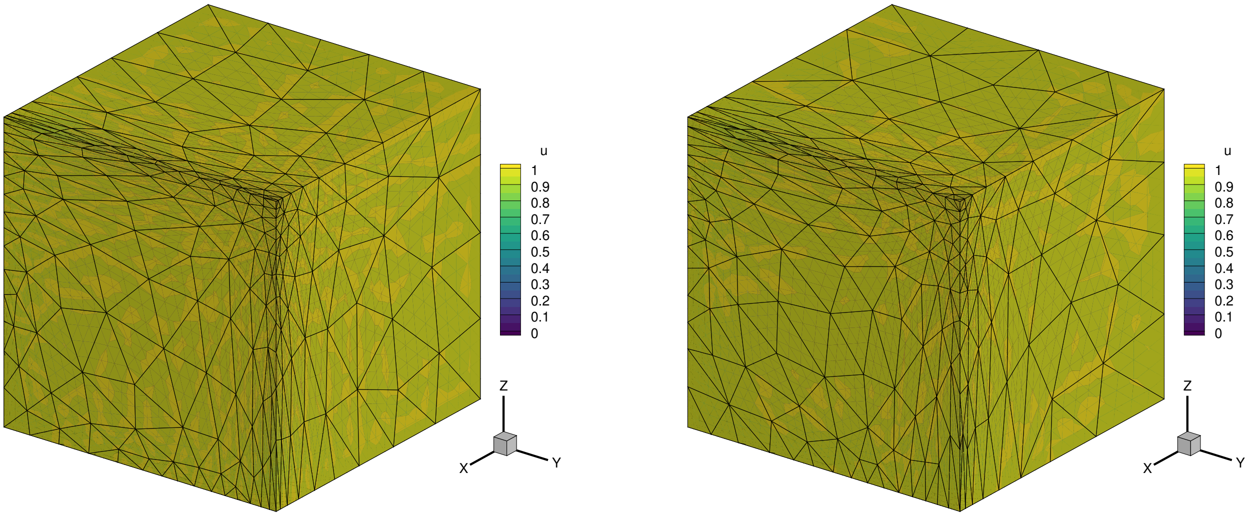
Figure 2.14: Final Meshes for Triple Boundary Layer problem for $p = 1$: $\sim$ 8000 elements, $p = 2$: $\sim$ 3200 elements and $p = 3$: $\sim$ 1600 elements.



(a) DG | $p = 1$ **(b)** CG | $p = 1$



(c) DG | $p = 2$ **(d)** CG | $p = 2$



(e) DG | $p = 3$ **(f)** CG | $p = 3$

Figure 2.15: Back view of final Meshes for Triple Boundary Layer problem for $p = 1$: $\sim$ 8000 elements, $p = 2$: $\sim$ 3200 elements and $p = 3$: $\sim$ 1600 elements.

CHAPTER 3

CONVERGENCE OF MESH ADAPTATION VIA METRIC OPTIMIZATION

Abstract

Adaptive Finite Element Methods (AFEM) are an increasingly common means of automatically controlling error in numerical simulations. Proofs of convergence and rate of convergence exist for AFEM, however these proofs typically rely upon a nested structure for the sequence of meshes. A Metric Adaptive Finite Element Method (MAFEM) utilizes the *continuous mesh model* and instead seeks to optimize a Riemannian metric field for a given cost, from which a mesh is generated. This meshing process results in a sequence of *non-nested* meshes. In this chapter we introduce a proof of convergence for a particular MAFEM, utilizing the optimization statement to relate the error on the sequence of meshes. In addition, we prove that such a sequence of meshes will demonstrate the optimal asymptotic rate of convergence for a given polynomial order. Finally some numerical results demonstrate the performance of the algorithm for a Poisson solution on an L-shaped domain.

3.1 *Introduction*

Mesh adaptivity is increasingly used for the efficient simulation of partial differential equations using finite element methods, the advantages being that it reduces user intervention in the meshing process and improves the quality of the final solutions. The development of mesh adaptivity can be roughly divided into two portions: *a posteriori* estimation of an error and the control of this estimate through mesh mechanics.

Two main types of *a posteriori* error estimation are residual or energy norm error, focused on solution error measured in various norms[31,32,36], and output error, focused on the error in output functionals of interest[45,49,55–57]. Reviews of error estimation are available including the textbook by Verfürth[103] and the monograph by Ainsworth and Oden[33].

103. Verfürth, *A Posteriori Error Estimation Techniques for Finite Element Methods*. 2013

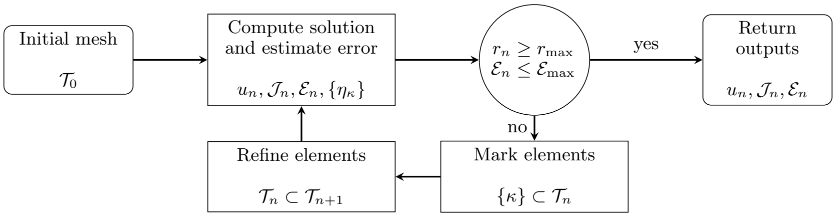
33. Ainsworth *et al.*, *A posteriori error estimation in finite element analysis*. 1997

Babuška and Vogelius[79] demonstrated the convergence of an adaptive finite element method (AFEM) in $1D$. Defining convergence as

$$\lim_{n\to\infty} |||u - u_n||| \to 0 \tag{3.1}$$

79. Babuška *et al.*, *Feedback and adaptive finite element solution of one-dimensional boundary value problems*. 1984

where $|||v|||$ is the energy norm, they proved under mild assumptions that an AFEM based on nested refinement must converge to zero error. Their feedback approach of solving on a mesh, estimating the error, marking elements and the refining them is depicted in figure 3.1. Dörfler improved on the work of Babuška and Vogelius[79], proving monotone convergence of an AFEM for Poisson's equation in $2D$, utilizing a more nuanced marking strategy[80]. This work was expanded upon by Morin, Nochetto and Siebert[81], who relaxed some of Dörfler's assumptions. A key aspect of this AFEM paradigm is that the final mesh is a 'descendent' of the original mesh as it is consists of a sequence of refinements from the starting mesh $\mathcal{T}_0$.



Initial mesh
$\mathcal{T}_0$
Compute solution and estimate error
$u_n, \mathcal{J}_n, \mathcal{E}_n, \{\eta_\kappa\}$
$r_n \geq r_{\max}$
$\mathcal{E}_n \leq \mathcal{E}_{\max}$
yes
Return outputs
$u_n, \mathcal{J}_n, \mathcal{E}_n$
no
Mark elements
$\{\kappa\} \subset \mathcal{T}_n$
Refine elements
$\mathcal{T}_n \subset \mathcal{T}_{n+1}$

Figure 3.1: A classical adaptive process starting from initial mesh $\mathcal{T}_0$, computing solution u_n, functional output $\mathcal{J}_n$, error estimate $\mathcal{E}_n$, localized error estimates $\{\eta_\kappa\}$, and resource requirement r_n.

79. Babuška *et al.*, *Feedback and adaptive finite element solution of one-dimensional boundary value problems*. 1984

80. Dörfler, *A convergent adaptive algorithm for Poisson's equation*. 1996

81. Morin *et al.*, *Data oscillation and convergence of adaptive FEM*. 2000

Binev *et al.* introduced a coarsening stage and showed that if it is possible to approximate solution u with $O(N^{-s})$ error, where N is the number of DOFs and $s > 0$ a convergence rate, then their algorithm will achieve the same asymptotic rate[82]. Cascon *et al.* relaxed the interior node property of Morin *et al.*[81], that had been prohibitive in practice, and proved convergence for a general self-adjoint second-order elliptic PDE[83].

Carstensen *et al.* combined the literature to arrive at the axioms of adaptivity which when satisfied prove the convergence (and quasi-optimal convergence) of an adaptive finite element algorithm, independent of whether the PDE is linear, and the discretization used[84]. This axiomatic approach was applied to output adaptation by Feischl *et al.*[85] for standard discretizations of second-order linear elliptic PDEs by controlling the error in the adjoint and the primal simultaneously.

84. Carstensen *et al.*, *Axioms of adaptivity*. 2014

Houston *et al.* used a marking-based adaptive approach for anisotropic adaptation[93], where they used *a posteriori* error estimation and 'local solves' to choose the optimal refinement from a discrete set of choices. This approach was also applied by Ceze and Fidkowski[61,104] where they included p refinement amongst the refinements and additionally used a merit function to balance the benefit of a refinement against the cost.

A generally key feature of the AFEM literature is the notion of marking and refining elements. These refinements ensure that, at least locally, the finite element spaces are nested, and from this a contraction map on the nested spaces can be derived. It is precisely this nested structuring that metric-based adaptation seeks to avoid in order to achieve efficiency gains through arbitrary anisotropy, unaffected by the choice of initial mesh.

An alternative interpretation of mesh adaptation is through the lens of optimization. Mesh optimization can be thought of as seeking a mesh that minimizes a specified error functional $\mathcal{E}(\mathcal{T})$ for a given maximum cost, $\mathbb{C}$. Written abstractly,

$$\mathcal{T}^* = \underset{\mathcal{T}\in\mathbb{T}(\Omega)}{\arg\inf}\, \mathcal{E}(\mathcal{T}) \tag{3.2a}$$

$$\text{s.t} \quad \mathbb{C}(\mathcal{T}) \leq \mathbb{C} \tag{3.2b}$$

where $\mathbb{T}(\Omega)$ is the space of conforming meshes of the domain Ω, $\mathcal{E}$ is an error functional to be minimized and $\mathbb{C}$ is a cost functional that constrains the optimization. As an example, $\mathcal{E}$ might be the L^2 error computed using a discretization and $\mathbb{C}$ the number of degrees of freedom in the discretization.

Equation (3.2) is a discrete optimization problem, and thus relatively intractable. We seek an approximate solution by considering a continuous relaxation. To perform this relaxation we optimize instead a Riemannian metric field, $\mathcal{M}(x)$, which is a smoothly varying field of symmetric positive definite tensors and x a point in the domain Ω. This field encodes distances between any two points of a domain through the expression

$$r_{\mathcal{M}}(\mathbf{a},\mathbf{b}) = \int_0^1 \sqrt{(\mathbf{b}-\mathbf{a})^T\mathcal{M}(\mathbf{a}+t(\mathbf{b}-\mathbf{a}))(\mathbf{b}-\mathbf{a})}\, dt. \tag{3.3}$$

This also defines an alternative notation for the length of a vector, $l_{\mathcal{M}}(\mathbf{v}) \equiv r_{\mathcal{M}}(\mathbf{v}_0,\mathbf{v}_1)$, where $\mathbf{v}_0, \mathbf{v}_1$ denote the start and end of $\mathbf{v}$ respectively.

Any mesh $\mathcal{T}$ has a corresponding metric field $\mathcal{M}(x)$. This field is such that all the edges of the mesh are approximately unit length under the metric $l_{\mathcal{M}}(e) \equiv l_{\mathcal{M}}(\mathbf{e}) \approx 1$, $\forall e \in \mathcal{E}(\mathcal{T})$, where $\mathbf{e}$ is the vector defined by edge e and $\mathcal{E}(\mathcal{T})$ denotes the set of all edges of $\mathcal{T}$. A typical requirement of a metric field is *quasi-unity*, $l_{\mathcal{M}}(e) \in [\frac{1}{\sqrt{2}}, \sqrt{2}]$, $\forall e \in \mathcal{E}(\mathcal{T})$. The connection between meshes and metric fields is referred to as mesh-metric duality[64,65].

Mesh-metric duality furnishes a continuous relaxation of equation (3.2),

$$\mathcal{M}^* = \underset{\mathcal{M}\in\mathbb{M}(\Omega)}{\arg\inf}\ \mathcal{E}(\mathcal{M}) \tag{3.4a}$$

$$\text{s.t}\quad \mathbb{C}(\mathcal{M}) \leq \mathbb{C} \tag{3.4b}$$

where $\mathbb{M}(\Omega)$ is the space of Riemannian metrics that conform to the domain Ω, and the functionals operate on the metric rather than the mesh. An explicit form rarely exists for $\mathcal{E}(\mathcal{M})$ or $\mathbb{C}(\mathcal{M})$ and thus they must be modeled. This modeling can be done with *a priori* analysis[64,65,77] or using *a posteriori* interrogation of the mesh[72,74].

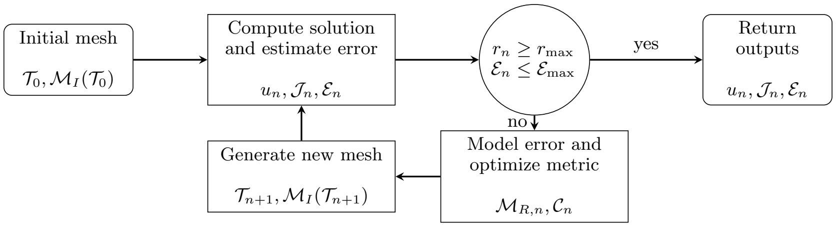

Initial mesh
$\mathcal{T}_0, \mathcal{M}_I(\mathcal{T}_0)$
Compute solution and estimate error
$u_n, \mathcal{J}_n, \mathcal{E}_n$
$r_n \geq r_{\max}$
$\mathcal{E}_n \leq \mathcal{E}_{\max}$
yes
Return outputs
$u_n, \mathcal{J}_n, \mathcal{E}_n$
no
Model error and optimize metric
$\mathcal{M}_{R,n}, \mathcal{C}_n$
Generate new mesh
$\mathcal{T}_{n+1}, \mathcal{M}_I(\mathcal{T}_{n+1})$

Figure 3.2: A metric adaptive process starting from initial mesh $\mathcal{T}_0$, computing solution u_n, functional output $\mathcal{J}_n$, error estimate $\mathcal{E}_n$, and resource requirement r_n.

Given a model for $\mathcal{E}(\mathcal{M})$ or $\mathbb{C}(\mathcal{M})$, the metric optimization is iterated at a given cost level until either a tolerance or resource limit is achieved. A typical example of this feedback loop is shown in figure 3.2, and can be seen to mirror the structure of figure 3.1. The ability to generate arbitrary element anisotropy and orientation during metric adaptation can lead to greater efficiency when compared to nested meshes which can be limited by the original mesh.

Metric optimization has found success particularly within the computational fluid dynamics community, we refer the reader to Alauzet and Loseille for a review of the field[66].

One means of proving convergence for a finite element method on a sequence of meshes is to show that the maximum element diameter approaches zero and to assert the meshes are *non-degenerate*. Non-degeneracy is defined as

$$\exists \gamma > 0 : \ \gamma \leq \frac{B_\kappa}{h_\kappa}, \quad \forall \kappa \in \mathcal{T}_n,\ \forall n \tag{3.5}$$

where B_κ and h_κ are the respective diameters of the largest *inscribed* and the smallest *circumscribing* balls of κ. This is a equivalent to a minimum angle condition[8]. The largest element diameter in a sequence of non-degenerate meshes converging to zero then ensures that the error in an interpolant defined on the mesh approaches zero[8], and for a stable finite element discretization this is sufficient.

One method of controlling the largest element diameter is by the assumption of *quasi-uniformity* which, in addition to *non-degeneracy*,

[66] Alauzet *et al.*, *A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics*. 2016

[8] Brenner *et al.*, *The Mathematical Thoery of Finite Element Methods, Third Edition*. 2008

[8] Brenner *et al.*, *The Mathematical Thoery of Finite Element Methods, Third Edition*. 2008

requires

$$\exists C : \frac{\max_{\kappa \in \mathcal{T}_n} h_\kappa}{\min_{\kappa \in \mathcal{T}_n} h_\kappa} \leq C, \ \forall n. \tag{3.6}$$

If the number of elements in a non-degenerate sequences of meshes diverges, then the smallest element diameter of the mesh must approach zero. The quasi-uniformity assumption then means that the largest element diameter must also approach zero, and thus the interpolation error approaches zero.

In Section 3.4 we introduce the concept of *quasi-stationarity* which bears similarity to quasi-uniformity. However, the coupling is introduced implicitly via an optimization process rather than explicitly via assumption.

The outline of the chapter is as follows. In Section 3.2 we introduce the mesh-metric duality, including the notion of the implied metric and an abstract definition of a metric-mesher. In Section 3.3 we demonstrate our framework for functional error modeling. Section 3.4 provides the definition of convergence we use in this chapter, which we then prove in Section 3.5 along with a rate of convergence. Finally Section 3.6 demonstrates the theoretical results of Section 3.5 for a simple test problem with geometric singularities.

3.2 *Mesh-Metric Duality*

A conforming mesh $\mathcal{T}$ of closed domain Ω is a finite set of elements κ, where all κ satisfy[1]

- κ is closed and the interiors $\mathring{\kappa}$ are non-empty and connected
- The boundary of $\partial\kappa$ of an element κ is Lipschitz-continuous
- $\Omega \equiv \cup_{\kappa \in \mathcal{T}} \kappa$
- $\mathring{\kappa}_i \cap \mathring{\kappa}_j \equiv \emptyset, \ \forall i, \ j.$
- The i-th trace of κ, denoted $\partial\kappa_i$, satisfies either $\exists \kappa' : \ \partial\kappa_i \cap \partial\kappa' \neq \emptyset$ or $\partial\kappa_i \cap \partial\Omega \neq \emptyset$.

The space of all *simplex* meshes that conform to Ω is denoted $\mathbb{T}(\Omega)$. For any given simplex $\kappa \in \mathcal{T}$, a unique, single-valued, elemental metric, $\mathcal{M}_\kappa \in \mathrm{Sym}_d^+$, exists such that

$$l_{\mathcal{M}_\kappa}(e_i) = \sqrt{\mathbf{e}_i^T \mathcal{M}_\kappa \mathbf{e}_i} = 1, \qquad \forall e_i \in \mathcal{E}(\kappa) \tag{3.7}$$

where $\mathcal{E}(\kappa)$ denotes the set of edges of κ, $\mathbf{e}_i$ denotes the vector defined by the edge e_i and $\mathrm{Sym}_d^+ \equiv \{A \in \mathbb{R}^{d\times d} : A_{ij} \equiv A_{ji}, \ \mathbf{v}^T A \mathbf{v} > 0, \ \forall \mathbf{v} \in \mathbb{R}^d\}$. This metric is given by the expression $\mathcal{M}_\kappa \equiv J_\kappa^{-T} J_\kappa^{-1}$ where J_κ is the Jacobian that maps the regular unit simplex $\hat{\kappa}$ to κ. The set of elemental metrics $\{\mathcal{M}_\kappa\}$ is element-wise discontinuous and this lack of continuity means it is not a true Riemannian metric field. To compute a smoothly-varying *implied* metric field from these elemental metrics, we use the Affine Invariant framework[5].

[5] Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

Definition 3 (Affine Invariant (AI) implied metric). *For a set of positive definite metric tensors, $\{\mathcal{M}_\kappa\}_{\kappa\in\kappa(v)} \in Sym_d^+$, where $\kappa(v)$ is the set of κ attached to vertex v, the Affine-Invariant implied metric for the vertex v is given by*

$$\mathcal{M}_v = \underset{\mathcal{M}\in Sym_d^+}{\arg\min} \frac{1}{|\kappa(v)|} \sum_{\kappa(v)} \|\log\left(\mathcal{M}_\kappa^{-\frac{1}{2}}\mathcal{M}\mathcal{M}_\kappa^{-\frac{1}{2}}\right)\|_F^2, \tag{3.8}$$

where $|\kappa(v)|$ is the cardinality of the set. This defines the implied metric

$$\mathcal{M}_I(x) = \underset{\mathcal{M}\in Sym_d^+}{\arg\min} \sum_{v(\kappa)} b_v(x)\|\log\left(\mathcal{M}_v^{-\frac{1}{2}}\mathcal{M}\mathcal{M}_v^{-\frac{1}{2}}\right)\|_F^2, \tag{3.9}$$

where $b_v(x)$ are the barycentric coordinates for the $\kappa \in \mathcal{T}$ that contains x.

Definition 3 requires the solution of a non-linear minimization problem, which can be reached using a Newton gradient descent algorithm[5].

[5] Pennec *et al.*, *A Riemannian framework for tensor computing*. 2006

Step Matrices

In perturbing a Riemannian metric field, it is necessary to maintain the properties of symmetry and positive definiteness, $\mathcal{M}(x) \in \text{Sym}_d^+, \forall x$. A first choice for a perturbation might be to add $\delta\mathcal{M}$ to give $\mathcal{M} \to \mathcal{M} + \delta\mathcal{M}$, however maintaining positive definiteness would then require placing restrictions on $\delta\mathcal{M}$ that may be difficult to enforce. An alternative is the use of *step matrices*, $S \in \text{Sym}_d$, defined by

$$S = \log\left(\mathcal{M}_0^{-\frac{1}{2}}\mathcal{M}\mathcal{M}_0^{-\frac{1}{2}}\right) \tag{3.10}$$

$$\mathcal{M} = \mathcal{M}_0^{\frac{1}{2}}\exp(S)\mathcal{M}_0^{\frac{1}{2}}. \tag{3.11}$$

A step matrix can be represented using the trace decomposition $S = s\mathcal{I} + \tilde{S}$, where $\text{tr}(\tilde{S}) = 0$. $s\mathcal{I}$ governs volumetric changes that alter the determinant of $\mathcal{M}$, whilst $\tilde{S}$ governs shape and orientation changes which do not alter the determinant. The *implied* metric field defined by $\{\mathcal{M}_v\}$ can then be perturbed by a set of vertex defined step matrices $\{S_v\}$.

An approximate elemental step matrix can be computed from the vertex step matrices using the expression $S_\kappa \equiv \frac{1}{(d+1)}\sum_{v(\kappa)} S_v$, where $(d+1) \equiv |v(\kappa)|$ is the number of vertices of simplex κ. This is an approximation as it does not relate to the $\mathcal{M}_\kappa$ of equation (3.7) exactly in most cases. However, this approximation provides a useful means of going from the vertex-defined step matrices to elemental-defined step matrices.

Metric contractive meshing

Three important metrics must be considered in discussing a mesher: the *implied metric*, $\mathcal{M}_I$, of the *starting mesh* given by Definition 3, the *requested metric*, $\mathcal{M}_R$, which is a perturbation from $\mathcal{M}_I$, and the *implied metric* of the mesh *returned* by the mesher $\mathcal{M}_{I'}$. Ideally the requested metric and implied metric of the returned mesh are the same, but in practice there can be discrepancy.

We now define an abstract notion of a metric contractive mesher

Definition 4 (β contractive metric mesher)**.** *A β contractive metric mesher $\mathcal{M} : \mathbb{T}(\Omega) \times \mathbb{M}(\Omega) \to \mathbb{T}(\Omega)$ must return a non-degenerate mesh $\mathcal{T}'$ that satisfies*

$$\sup_{x\in\Omega} \|S^{\mathcal{M}_R}_{\mathcal{M}_I}(x)\|_F > \sup_{x\in\Omega} \|S^{\mathcal{M}_R}_{\mathcal{M}_{I'}}(x)\|_F, \tag{3.12}$$

if $\mathcal{M}_I$ and $\mathcal{M}_R$ satisfy

$$\max_{v(\mathcal{T})} \|S^{\mathcal{M}_R}_{\mathcal{M}_I}(x_v)\|_F \geq \beta, \tag{3.13}$$

where $\mathcal{T}$ is the current mesh, $\mathcal{M}_I$ is the implied metric of $\mathcal{T}$, $\mathcal{M}_R$ is the requested metric and $\mathcal{M}_{I'}$ is the implied metric of $\mathcal{T}'$.

Definition 4 can be related to the notion of achieving a quasi-unity mesh, that is a mesh with $l_{\mathcal{M}(S)}(e) \in (\frac{1}{\sqrt{2}}, \sqrt{2})$, $\forall e \in \mathcal{E}(\mathcal{T})$. The ratio of the length of a vector $\mathbf{v}$ arising from a step S acting on a metric $\mathcal{M}$ can be bound in terms of the eigvenvalues $\lambda(S)$,

$$\min_{\mathbf{v}\in\mathbb{R}^d} \frac{l_{\mathcal{M}(S)}(\mathbf{v})}{l_{\mathcal{M}}(\mathbf{v})} = e^{\frac{1}{2}\min\lambda(S)}, \qquad \max_{\mathbf{v}\in\mathbb{R}^d} \frac{l_{\mathcal{M}(S)}(\mathbf{v})}{l_{\mathcal{M}}(\mathbf{v})} = e^{\frac{1}{2}\max\lambda(S)}. \tag{3.14}$$

As a result, edge lengths can fall outside quasi-unity if $\max|\lambda(S)| > \log(2)$. Since the Frobenius norm can be written in terms of eigenvalues as $\|S\|_F \equiv \sqrt{\sum_i \lambda_i^2}$, then $\max|\lambda| \leq \|S\|_F \leq \sqrt{d}\max|\lambda(S)|$. If $\beta = \log(2)$, then this would guarantee the mesher would act if any edge was not quasi-unit. However, $\beta = \log(2)$ could require remeshing even when all edges are quasi-unit. Alternatively, setting $\beta = \sqrt{d}\log(2)$ would not require remeshing if all edges are quasi-unit.

3.3 *Metric Functional Modeling*

We now introduce the modeling process for the cost and error functionals of the metric, $\mathcal{E}(\mathcal{M})$ and $\mathbb{C}(\mathcal{M})$ respectively.

Cost Model

We will assume that the cost functional $\mathbb{C}(\mathcal{M})$ can be localized,

$$\mathbb{C}(\mathcal{M}) = \int_\Omega c(\mathcal{M}(x), x)dx, \tag{3.15}$$

where $c(\mathcal{M}(x), x) = c_p\sqrt{\det(\mathcal{M}(x))}$ is the *local cost kernel* and $c_p \in \mathbb{R}^+$ is a coefficient. For instance if $\mathbb{C}$ is total number of degrees of freedom, then c_p is the number of degrees of freedom per unit volume under the metric.

The cost functional can be localized to an element

$$\rho_\kappa \equiv \int_\kappa c(\mathcal{M}(x); x)dx = c_p \int_\kappa \sqrt{\det(\mathcal{M}(x))}dx. \tag{3.16}$$

Define $\mathcal{M}(x)$ for inside of κ as a perturbation by $S(x)$ from $\mathcal{M}_\kappa$

$$\mathcal{M}(x) = \mathcal{M}_\kappa^{\frac{1}{2}} e^{S(x)} \mathcal{M}_\kappa^{\frac{1}{2}}, \tag{3.17}$$

where the step can be decomposed $S(x) = s(x)\mathcal{I} + \tilde{S}(x)$ and $\mathrm{tr}(\tilde{S}(x)) = 0$. The elemental cost is then

$$c_p \int_\kappa \sqrt{\det(\mathcal{M}(x))}dx = \int_\kappa \sqrt{\det(\mathcal{M}_\kappa^{\frac{1}{2}} e^{s(x)\mathcal{I}+\tilde{S}(x)} \mathcal{M}_\kappa^{\frac{1}{2}})}dx \tag{3.18}$$

$$= c_p\sqrt{\det(\mathcal{M}_\kappa)} \int_\kappa e^{\frac{s(x)d}{2}} dx \tag{3.19}$$

$$= \rho_{\kappa_0} \frac{1}{|\kappa|} \int_\kappa e^{\frac{s(x)d}{2}} dx, \tag{3.20}$$

where $\rho_{\kappa_0} \equiv c_p\sqrt{\det \mathcal{M}_\kappa}|\kappa| = \mathrm{dof}(\hat{\kappa})$.

An approximate cost functional can then be formulated by evaluating equation (3.20). To do so requires prescribing a variation of $s(x)$ across κ. One interpolation scheme is barycentric interpolation, $s(x) \equiv \sum_v b_v(x)s_v$, where $b_v(x)$ are the barycentric coordinates for the vertices and $s_v = \frac{\mathrm{tr}(S_v)}{d}$. Given s_v are defined relative to the vertex-defined metrics, $\mathcal{M}_v$, this formula represents an approximation for $s(x)$, which ought to be defined relative to $\mathcal{M}_\kappa$. Conditional on this approximation, using this explicit formula for $s(x)$ is valid. This is because despite the calculation of $\mathcal{M}(x)$ requiring the solution of a non-linear optimization problem, the calculation of $\det(\mathcal{M}(x))$ can be done in closed form[105].

Equation (3.20) can be evaluated using numerical quadrature. The original Yano and Darmofal work utilizes single point quadrature which results in the elemental cost model

$$\rho_\kappa(\{S_v\}) = \rho_{\kappa_0} e^{\frac{s_\kappa d}{2}} \tag{3.21a}$$

$$\rho_{\kappa_0} = \int_\kappa c_p\sqrt{\det(\mathcal{M}_\kappa)}dx = c_p|\hat{\kappa}| = \mathrm{dof}(\hat{\kappa}), \tag{3.21b}$$

[105] Arsigny *et al.*, *Geometric means in a novel vector space structure on symmetric positive-definite matrices*. 2007

where $S_\kappa = \frac{1}{d+1}\sum_{v(\kappa)} S_v$ is an elemental step matrix, with trace decomposition $S_\kappa = s_\kappa \mathcal{I} + \tilde{S}_\kappa,\ \mathrm{tr}(\tilde{S}_\kappa) = 0$.

In the analysis of Section 3.5 we assume the use of this model for simplicity. However, the analysis remains valid for any cost model that satisfies

1. $\frac{\partial \rho_\kappa}{\partial S_v} = C_\rho \mathcal{I} \rho_\kappa$ where $C_\rho \in \mathbb{R}^+$ is a constant independent of κ and $\mathcal{T}$
2. $\rho_\kappa(0) \sim \mathbb{C}(\kappa)$ where $\mathbb{C}(\kappa)$ is the true cost of κ.

$A \sim B$ means $A \lesssim B$ and $B \lesssim A$, where $A \lesssim B$ means $\exists C < \infty :\ A \leq CB$ where C is independent of any element diameter h_κ. This is the case for the cost model of equation (3.21).

Error Locality and Modeling

To localize the error functional, we make an error locality assumption,

$$\mathcal{E}(\mathcal{M}) = \int_\Omega e(\mathcal{M}(x), x) dx, \tag{3.22}$$

where $e(\mathcal{M}(x), x)$ is the *local error kernel*. This assumption is in general only valid for certain purely local functionals, such as L^2 error of an element-wise discontinuous interpolant, however it has found justification in practice[2]. Given the locality assumption we use a set of compactly supported functions $\{\phi_i\}$ that form a partition of unity, $\sum_i \phi_i(x) = 1,\ \forall x \in \Omega$, to define local approximations

$$\mathcal{E}(\mathcal{M}) = \sum_i \epsilon_i = \sum_i \int_\Omega \phi_i e(\mathcal{M}(x), x) dx \tag{3.23a}$$

$$\epsilon_i = \int_\Omega \phi_i e(\mathcal{M}(x), x) dx \tag{3.23b}$$

$$\approx \int_\Omega \phi_i e(\mathcal{M}_i, x) dx \implies \epsilon_i = \epsilon_i(\mathcal{M}_i), \tag{3.23c}$$

where $\mathcal{M}_i$ is a discrete metric associated with ϕ_i. The local error $\epsilon_i \in \mathbb{R}$ can be positive or negative so in the modeling process we instead use the positive *local error indicator* $\eta_i = |\epsilon_i|$. If the exact local error ϵ_i is not available an estimate, $\tilde{\epsilon}_i \approx \epsilon_i$, may be used instead. In Section 3.5 we discuss the implications of using estimators in more detail.

In Yano and Darmofal's original work, which was focused on discontinuous Galerkin methods, the error models are *elemental* in that $\phi_i(x) \equiv \mathbb{1}(x \in \kappa)$, and the corresponding metrics are the elemental metrics $\mathcal{M}_\kappa$. However, in this work we use an alternative localization motivated by the work of Richter and Wick[49]: the linear 'hat' functions, $\phi_v(x_{v'}) = \delta_{v,v'},\ \forall v, v' \in v(\mathcal{T})$, where $v(\mathcal{T})$ denotes the set of vertices of $\mathcal{T}$, with the vertex defined metric $\mathcal{M}_v$. We have used this localization to optimized mesh metrics for continuous finite element methods, see Chapter 2.

[2] Yano *et al.*, *An Optimization Framework for Anisotropic Simplex Mesh Adaptation: Application to Aerodynamic Flows*. 2012

[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015

Error Model Constraint

The local error model utilized in Yano and Darmofal's work took the form

$$\eta_v(S_v) = \eta_v e^{\mathrm{tr}(R_v S_v)}, \tag{3.24}$$

where R_v is referred to as the *rate matrix*. This model is a generalization of the isotropic error model

$$\eta_v(h_v) = \eta_v {\frac{h_v}{h_{v,0}}}^{r_v^{iso}}, \tag{3.25}$$

to include the effect of local anisotropic deformations on the error[72].

Ultimately the metric optimization is posed in terms of a perturbation to the current metric, and it is desirable to restrict the magnitude of these perturbations. The models are generally constructed in terms of a linearization around the current mesh and too large a perturbation may take a model too far outside of the region on which it was trained. In Yano and Darmofal's work this restriction on the perturbations took the form of a constraint $|(S_v)_{i,j}| \leq 2\log(2)$ applied at every vertex. We replace this constraint with a regularization included in the error model directly

$$\eta_v(S_v) = \eta_v e^{\mathrm{tr}(R_v S_v) + \frac{\alpha \|R_v\|_F}{2} \|S_v\|_F^2}, \ \forall v \in v(\mathcal{T}), \tag{3.26}$$

where $\alpha \in \mathbb{R}^+$ is a constant to be specified. This regularization bounds the maximum error reduction predicted by the model, in that

$$\arg\min(\eta_v(S_v)) = -\frac{1}{\alpha} \frac{R_v}{\|R_v\|_F}. \tag{3.27}$$

Applying a constraint on the magnitude of S_v is analogous to applying a trust region to the optimization, whereas this regularization enforces the trust region through a penalty term. In the discussion that follows, $\eta_v(S_v)$ denotes equation (3.26) whilst η_v without arguments denotes the local error indicator as calculated in the estimation process. The regularized model of equation (3.26) is a key contribution of this chapter that is necessary for the result in Section 3.5.

We place no restriction on the means of calculating the *rate matrix*, $R_v \in \mathrm{Sym}_d$, however we do require that $\exists c \in \mathbb{R} : \ 0 > c \geq \mathrm{tr}(R_v)$ and $\exists C_R < \infty : \ \|R_v\|_F \leq C_R$[1]. One means of calculating such an R_v is via the MOESS algorithm for continuous discretizationdescribed in Chapter 2.

Optimization Statement

Having defined the error and cost models, it is now possible to state the optimization statement, posed in terms of vertex-defined step matrices $\{S_v\}$.

[72] Yano *et al.*, *An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation*. 2012

[1] For instance $R_v = -\mathcal{I}$, $\forall v$ satisfies this requirement

Definition 5 (Step Matrix Optimization Statement)**.**

$$\{S_v^*\} = \underset{\{S_v\}\in Sym_d}{\arg\inf} \sum_{v(\mathcal{T})} \eta_v e^{tr(R_v S_v)+\frac{\alpha\|R_v\|_F}{2}\|S_v\|_F^2} \tag{3.28a}$$

$$s.t \quad 0 \geq \sum_{\kappa(\mathcal{T})} \rho_\kappa(\{S_v\}) - \mathbb{C} \tag{3.28b}$$

$$0 \geq C_s - tr\,(S_v), \qquad \forall v \in v(\mathcal{T}) \tag{3.28c}$$

where $\rho_\kappa(\{S_v\})$ *is the cost model,* $\mathbb{C} \in \mathbb{R}_{>0}$ *is a specified target cost, and* $C_s \in \mathbb{R}_{<0}$ *is a limit on vertex coarsening.*

Equation (3.28c) prevents severe coarsening at any given vertex, which might occur if η_v was much smaller than expected due to cancellation effects. We require that $C_s \leq 0$ to allow coarsening, and that C_s be sufficiently negative that a strictly feasible interior point exist. We choose $C_s = \xi \min(-\frac{1}{\alpha}, 2\log(\frac{\mathbb{C}}{\mathbb{C}(\mathcal{T})}))$, where $\xi \geq 1$ and $2\log(\frac{\mathbb{C}}{\mathbb{C}(\mathcal{T})})$ is the step that would be required at all vertices to give $\int_\Omega c_p \det(\mathcal{M}_v(S_v)) \leq \mathbb{C}$. Thus Definition 5 always has a strictly feasible interior point, and in practice we choose $\xi = 2$. Definition 5 is a convex optimization problem and if there is a feasible strictly interior point, from Slater's condition a solution exists and is unique[106].

As a well-posed constrained optimization Definition 5 has a Lagrangian.

Lemma 1. *The Lagrangian of Definition 5 is*

$$\begin{aligned}\mathcal{L}(\{S_v\}, \lambda, \{\mu_v\}) = &\sum_{v(\mathcal{T})} \eta_v e^{tr(R_v S_v)+\frac{\alpha\|R_v\|_F}{2}\|S_v\|_F^2} \\ &+ \lambda(\sum_{\kappa\in\mathcal{T}} \rho_\kappa(\{S_v\}) - \mathbb{C}) \\ &+ \sum_v \mu_v(C_s - tr(S_v))\end{aligned} \tag{3.29}$$

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial S_v} = &(R_v + \alpha\|R_v\|_F S_v)\eta_v e^{tr(R_v S_v)+\frac{\alpha\|R_v\|_F}{2}\|S_v\|_F^2} \\ &+ \lambda C_\rho \mathcal{I} \sum_{\kappa(v)} \rho_\kappa(\{S_v\}) - \mu_v \mathcal{I}\end{aligned} \tag{3.30}$$

where $\lambda \geq 0$ *and* $\mu_v \geq 0$ *from dual feasibility.*

3.4 *Definition of convergence*

Babuška and Vogelius defined convergence in their seminal work[79] as

$$\lim_{n\to\infty} |||u - u_n||| \to 0 \tag{3.31}$$

where under mild assumptions they prove convergence of a one dimensional adaptive algorithm. Their remarkably succinct proof assumes a well-posed problem, a nested structuring of the mesh sequence and that the adaptive process satisfy 'δ regularity'. This means

[106] Boyd *et al.*, *Convex Optimization*. 2004

[79] Babuška *et al.*, *Feedback and adaptive finite element solution of one-dimensional boundary value problems*. 1984

that fixing $\delta \in [0,1]$

$$\exists \kappa' \in \{\overline{\kappa}\}_n \subset \mathcal{T}_n : \; \eta_{\kappa'} \geq \delta \max_{\kappa(\mathcal{T}_n)} \eta_\kappa, \; \forall n \tag{3.32}$$

where $\{\overline{\kappa}\}_n$ is the set of elements marked for refinement at stage n of the adaptive algorithm. If two meshes are said to be nested, denotes $\mathcal{T}_a \subset \mathcal{T}_b$, this means that

$$\exists \{\kappa'\} \in \mathcal{T}_b : \cup\{\kappa'\} \equiv \kappa, \; \forall \kappa \in \mathcal{T}_a. \tag{3.33}$$

Namely that all elements of $\mathcal{T}_b$ come from a sequence of subdivisions of $\mathcal{T}_a$. Correspondingly a finite element space defined on $\mathcal{T}_a$ is a subset of the space defined on $\mathcal{T}_b$, which is denoted $\mathcal{V}_p(\mathcal{T}_a) \subset \mathcal{V}_p(\mathcal{T}_b)$, where $\mathcal{V}_p(\mathcal{T})$ denotes a finite element space using p-th order polynomials defined on mesh $\mathcal{T}$.

Equation (3.32) requires that $\{\overline{\kappa}\}_n \neq \emptyset$, namely that at each stage an element must be subdivided. The nested structure of the refinement thus means that the number of elements must grow unboundedly so long as $\exists \kappa \in \mathcal{T}_n : \; |||u - u_n|||_\kappa > 0$, and thus $\mathbb{C}_n \to \infty$.

Under the additional assumption of *non-degeneracy* this proof naturally extends to higher dimensions. Any AFEM consisting of nested refinement without coarsening inherits a proof of convergence from equation (3.32). The refinements need not be isotropic and the process of selecting $\{\overline{\kappa}\}_n$ may be highly sophisticated.

This feedback approach does not require that $\max_{\kappa \in \mathcal{T}_n} |\kappa| \to 0$, which thus differentiates it from a *quasi-uniformity* style proof of convergence. For instance if $\exists \omega \subset \Omega : \; u|_\omega \in \mathbb{P}^p(\omega)$, then with the marking strategy of equation (3.32) the maximum element diameter need not approach zero.

In the spirit of Babuška and Vogelius we define convergence in terms of the limit as $n \to \infty$. Let $\{\mathcal{T}_n\}$ be a sequence of meshes that come from a metric adaptive process as in Definition 5. The error is defined as $\mathcal{E}_n \equiv \mathcal{J}_n - \mathcal{J}$, where $\mathcal{J}_n \equiv \mathcal{J}(u_n)$ is the *numerical* functional output and $\mathcal{J} \equiv \mathcal{J}(u)$ the *true* functional output, then convergence is defined as

$$\lim_{n \to \infty} \mathcal{E}_n = 0. \tag{3.34}$$

This definition represents a generalization of the definition of equation (3.31) as can be seen by choosing $\mathcal{J}(u_n) \equiv |||u - u_n|||$. Crucially this definition of convergence makes no assumptions on the relationship between meshes and does not require monotonic reduction in the error. It also does not place any explicit restriction on the cost of a sequence of meshes $\{\mathbb{C}(\mathcal{T}_n)\}$.

Mesh stationarity

We now introduce the concept of a *quasi-stationary* mesh.

Definition 6 (Quasi-stationary mesh)**.** *A mesh on which the solution to the optimization problem in Definition 5 has an active global cost constraint, and thus $\lambda > 0$, is* quasi-stationary*. This implies that*

$$\lambda C_\rho \equiv \frac{|w_v \eta_v - \mu_v|}{\sum_{\kappa(v)} \rho_\kappa(\{S_v\})}, \qquad \forall v \in v(\mathcal{T}), \tag{3.35}$$

where $w_v \equiv (r_v + \alpha \|R_v\|_F s_v) e^{tr(R_v S_v) + \frac{\alpha \|R_v\|_F}{2} \|S_v\|_F^2}$ and μ_v is the Lagrange multiplier associated with the constraint on the trace of S_v.

Proof. The first order optimality condition for Definition 5 requires that $\frac{\partial \mathcal{L}}{\partial S_v} = 0,\ \forall S_v$. Rearranging then gives

$$-\lambda C_\rho \mathcal{I} = \frac{(R_v + \alpha \|R_v\|_F S_v) \eta_v e^{\mathrm{tr}(R_v S_v) + \frac{\alpha \|R_v\|_F}{2} \|S_v\|_F^2} - \mu_v \mathcal{I}}{\sum_{\kappa(v)} \rho_\kappa(\{S_v\})}, \qquad \forall v \in v(\mathcal{T}). \tag{3.36}$$

Taking the trace portion and the absolute value then gives equation (3.35). □

Definition 6 means that if a metric is optimized and the global cost constraint is active, then η_v for any two vertices are then coupled by the optimality condition. The optimization may request large changes of a *quasi-stationary* mesh, however those changes are related by the coupling introduced by the cost constraint. The *quasi* descriptor is supposed to mirror the classical notion of *quasi-uniformity* whereby the ratio of element diameters for any two elements in a mesh is bounded by some *a priori* prescribed constant. The difference between *quasi-stationarity* and *quasi-uniformity* is that the constant in *quasi-stationarity* relates local errors directly and arises naturally during the optimization process, as opposed to *quasi-uniformity* which relates the local element diameters and is imposed by assumption.

Lemma 2. *For a mesh $\mathcal{T}$, if a step metric optimization of Definition 5 is performed, and*

- $\lambda > 0$
- $\exists c \in \mathbb{R} :\ 0 > c \geq r_v$
- $\exists C_R < \infty :\ \|R_v\|_F \leq C_R$

then

$$\lambda \gtrsim \eta_v, \qquad \forall v \in v(\mathcal{T}) \tag{3.37}$$

$$\lambda \sim \eta_v, \qquad \forall \tilde{v} \subset v(\mathcal{T}) \tag{3.38}$$

where $\tilde{v} \equiv \{v \in v(\mathcal{T}) :\ \mu_v = 0\}$, $A \gtrsim B$ means $\exists C \in \mathbb{R}^+ : AC \geq B$ where C is a constant, and $A \sim B$ means $A \lesssim B$ and $A \gtrsim B$.

Proof. Stationarity requires that $\frac{\partial \mathcal{L}}{\partial S_v} = 0$, and taking the trace free portion gives $\tilde{S}_v = -\frac{1}{\alpha}\frac{\tilde{R}_v}{\|R_v\|_F}$, $\forall v$. The lower bound of C_s and dual feasibility of $\lambda \geq 0$ means that $s_v \in [C_s, \frac{-r_v}{\alpha\|R_v\|_F}]$, $\forall v$. Thus, $\|S_v\|_F < \infty$, $\forall v$. The bound on $\|R_v\|_F$ then ensures that w_v remains bounded, where w_v is defined as in Definition 6, and consequently

$$\exists[\gamma^-, \gamma^+] \subset \mathbb{R}_{>0} : \; w_v \in [\gamma^-, \gamma^+], \; \forall v. \tag{3.39}$$

The bounds on s_v ensure that $\sum_{\kappa(v)} \rho_\kappa(\{S_v\})$ remains bounded,

$$\exists[\rho^-, \rho^+] \subset \mathbb{R}_{>0} : \; \sum_{\kappa(v)} \rho_\kappa(\{S_v\}) \in [\rho^-, \rho^+], \; \forall v. \tag{3.40}$$

The constants $\gamma^\pm$ and $\rho^\pm$ depend on α, C_s and $\mathcal{M}$ (but not $\mathcal{T}$). The active global cost constraint means $\lambda > 0$ and thus γ^-, $\rho^- > 0$. Taking the trace decomposition of $\frac{\partial \mathcal{L}}{\partial S_v} = 0$ then gives

$$\lambda \equiv \frac{w_v \eta_v - \mu_v}{C_\rho \sum_{\kappa(v)} \rho_\kappa(\{S_v\})} \tag{3.41}$$

$$\geq \frac{w_v \eta_v}{C_\rho \sum_{\kappa(v)} \rho_\kappa(\{S_v\})}, \qquad \forall v \in v(\mathcal{T}). \tag{3.42}$$

□

A convergence condition for a Riemannian metric field

One sufficient condition that a bounded and stable finite element method on a sequence of non-degenerate meshes, $\{\mathcal{T}_n\}$, will converge to zero error is that $\max_{\kappa(\mathcal{T}_n)} |\kappa| \to 0$, where $|\kappa|$ is the d-volume of κ[8]. This is because as $\max_{\kappa(\mathcal{T}_n)} |\kappa| \to 0$, the error of the best interpolation in the finite element space $\mathcal{V}_p(\mathcal{T}_n)$ goes to zero.

In order to write a similar expression in terms of the metric field, we first need to define a lemma.

Lemma 3. *If a mesh is non-degenerate, then*

$$\exists C : \; \frac{1}{C} \leq \frac{|\kappa|}{|\kappa'|} \leq C, \; \forall\{\kappa, \, \kappa'\} \in \kappa(v) \tag{3.43}$$

Proof. Non-degeneracy implies $\#\kappa(v) < \infty$, where # denotes the cardinality of a set, and thus any two κ, $\kappa' \in \kappa(v)$ can be connected by a finite sequence of elements. Non-degeneracy also implies

$$\exists C : \; \frac{1}{C} \leq \frac{|\kappa|}{|\kappa'|} \leq C, \; \forall\{\kappa, \, \kappa'\} \; : \partial\kappa \cap \partial\kappa' \neq \emptyset. \tag{3.44}$$

Repeated application of equation (3.44) then furnishes equation (3.43).

□

Lemma 3 enables the definition an equivalent condition under mesh-metric duality directly in terms of the vertex-defined metric.

[8] Brenner *et al.*, *The Mathematical Thoery of Finite Element Methods, Third Edition*. 2008

Lemma 4. *For a sequence of non-degenerate meshes, if*

$$\max_{v(\mathcal{T}_n)} \det(\mathcal{M}_v)^{-\frac{1}{2}} \to 0$$

where $\mathcal{M}_v$ is the vertex implied metric, then $\|u - \Pi u\|_{L^2(\Omega)} \to 0$, where Π is the L^2 projector and $u \in L^2(\Omega)$.

Proof. The explicit definition $\mathcal{M}_\kappa \equiv J_\kappa^{-T} J_\kappa^{-1}$ means that $|\kappa| \to 0 \iff \det(\mathcal{M}_\kappa)^{-\frac{1}{2}} \to 0$. The determinant of $\mathcal{M}_v$ is given by the expression[105]

$$\det(\mathcal{M}_v) = \prod_{\kappa(v)} \det(\mathcal{M}_\kappa)^{\frac{1}{\#\kappa(v)}}. \tag{3.45}$$

Thus if $|\kappa(v)| \to 0$ then $\det(\mathcal{M}_v)^{-\frac{1}{2}} \to 0$. Conversely if $\det(\mathcal{M}_v)^{-\frac{1}{2}} \to 0$ then $\exists \kappa \in \kappa(v) : \det(\mathcal{M}_\kappa)^{-\frac{1}{2}} \to 0$ and from Lemma 3 this in turn implies that $\det(\mathcal{M}_\kappa)^{-\frac{1}{2}} \to 0,\ \forall \kappa \in \kappa(v)$. Thus $\det(\mathcal{M}_v)^{-\frac{1}{2}} \to 0 \iff |\kappa(v)| \to 0$ which implies that $\max_{v(\mathcal{T}_n)} \det(\mathcal{M}_v)^{-\frac{1}{2}} \to 0 \iff \max_{v(\mathcal{T}_n)} |\kappa(v)| \to 0$ and the result follows. □

We now relate this to the situation where the cost constraint is inactive.

Lemma 5. *If $\lambda_n = 0$ and $r_v < 0$, then the conditional expression*

$$\eta_v > 0 \implies S_v = \frac{-1}{\alpha} \frac{R_v}{\|R_v\|_F},\ s_v = tr(S_v) > 0, \tag{3.46}$$

holds for all vertices of $\mathcal{T}_n$.

Proof. If $\lambda = 0$, then requiring $\frac{\partial \mathcal{L}}{\partial S_v} = 0$ gives

$$0 = (R_v + \alpha \|R_v\|_F S_v)\eta_v e^{\mathrm{tr}(R_v S_v) + \frac{\alpha \|R_v\|_F}{2}\|S_v\|_F^2} - \mu_v \mathcal{I},\ \forall v \in v(\mathcal{T}_n). \tag{3.47}$$

Taking the trace gives

$$\mu_v = (r_v + \alpha \|R_v\| s_v)\eta_v e^{\mathrm{tr}(R_v S_v) + \frac{\alpha \|R_v\|_F}{2}\|S_v\|_F^2}, \tag{3.48}$$

where $r_v < 0$. Complementary slackness means $\mu_v > 0 \implies s_v = C_s < 0$ but then equation (3.48) would give $\mu_v < 0$, a contradiction, thus $\mu_v = 0$. Making the substitution gives

$$S_v = \frac{-1}{\alpha} \frac{R_v}{\|R_v\|_F} \tag{3.49}$$

□

[105] Arsigny *et al.*, *Geometric means in a novel vector space structure on symmetric positive-definite matrices.* 2007

3.5 Proof of convergence

In this section we prove

- The error for a sequence of MOESS meshes must converge as $\mathbb{C}_n \to \infty$
- Under an additional assumption, the asymptotic rate of convergence is the best achievable by the polynomial order used in the discretization.

The key idea of the proof is to utilize the coupling from Lemma 2 for *quasi-stationary* meshes in order to relate convergence in one region of the mesh to elsewhere. This idea is closely related to *quasi-uniformity* where element diameters, and thereby local interpolation errors, are explicitly coupled. The main differences compared to the proof in this section are three-fold. Firstly, the coupling is posed in terms of the *continuous mesh model*, namely via the metric field $\mathcal{M}(x)$. Secondly, the interaction between the requested metric and the mesher is defined implicitly using Definition 4 rather than through any explicit relation. Thirdly, the coupling is posed directly in terms of the error to be controlled rather than indirectly through a geometric relation. The approach taken in the analysis is rooted in optimization which contrasts with that of standard AFEMs, where the proof of convergence generally relies upon the construction of a contraction relation between consecutive meshes.

Theorem 1 (Convergence of Step Matrix Optimization). *Let $u \in L^2(\Omega)$, then define $u_n \in \mathcal{V}_p(\mathcal{T}_n)$ to be the discrete approximation of u with a stable finite element discretization, where the mesh $\mathcal{T}_n$ comes from a step matrix optimization process with cost $\mathbb{C}_n$ (Definition 5) using metric mesher $\mathcal{M} : \mathbb{T}(\Omega) \times \mathbb{M}(\Omega) \to \mathbb{T}(\Omega)$.*

Define $\mathcal{E}_n = \sum_{v(\mathcal{T}_n)} \epsilon_v$ as the error in a computed output functional $\mathcal{E}_n = \mathcal{J}(u_n) - \mathcal{J}(u)$ computed with solution u_n. Construct a monotonic increasing sequence $\{\mathbb{C}_n\}$ by setting $\mathbb{C}_{n+1} > \mathbb{C}_n$ if $\sum_{v(\mathcal{T}_n)} \eta_v > 0$. Then if

(A.1) $\lim_{n\to\infty} \mathbb{C}_n = \infty$ *or* $\mathcal{E}_{N_{\max}} = 0$

(A.2) $\exists c \in \mathbb{R} :\ 0 > c \geq r_v$

(A.3) $\exists C_R < \infty :\ \|R_v\|_F \leq C_R$

(A.4) $\eta_v \lesssim |\kappa(v)|$

(A.5) $|\epsilon_v| \lesssim \eta_v$

(A.6) $\mathcal{M} :\ \mathbb{T}(\Omega) \times \mathbb{M}(\Omega) \to \mathbb{T}(\Omega)$ *is a β contractive metric mesher (Definition 4)*

(A.7) $\exists \epsilon_\alpha > 0 : \ \alpha = \frac{1}{\beta + \epsilon_\alpha}$ *where* α *is the local model regularization coefficient*

it follows that

$$\lim_{n \to \infty} \mathcal{E}_n = 0. \tag{3.50}$$

Proof. (A.1) prevents finite sequences that do not converge to zero error, so we need only consider infinite sequences. Partition the sequence of meshes $\{\mathcal{T}_n\} = \{\hat{\mathcal{T}}_n\} \cup \{\overline{\mathcal{T}}_n\}$, into those that are *quasi-stationary* $\{\hat{\mathcal{T}}_n\}$, and those that are *not quasi stationary* $\{\overline{\mathcal{T}}_n\}$. We now show that both subsequences must be finite or converge to zero error.

Consider $\{\hat{\mathcal{T}}_n\}$. Under Affine-Invariant interpolation $\det(\mathcal{M}(x)) = \prod_v \det(\mathcal{M}_v)^{b_v(x)}$ where $b_v(x)$ are barycentric coordinates. Under the action of vertex-defined step matrices, $S_v = s_v \mathcal{I} + \mathrm{tr}(\tilde{S}_v)$, the determinant is thus given by

$$\det(\mathcal{M}(S(x))) = e^{\sum_v b_v(x) s_v} \prod_v \det(\mathcal{M}_v)^{b_v(x)}.$$

This combined with (A.2) means that if $\mathbb{C}_{n+1} > \mathbb{C}_n$ then $\exists v \in v(\mathcal{T}_n) : s_v > 0$, and thus that

$$\exists x \in \Omega : \ \det(\mathcal{M}_{R_n}(x))^{-\frac{1}{2}} < \det(\mathcal{M}_{I_n}(x))^{-\frac{1}{2}}.$$

Given $\mathcal{M}$ is β-contractive, either $\mathcal{M}$ acts such that

$$\exists x \in \Omega : \ \det(\mathcal{M}_{I_{n+1}}(x))^{-\frac{1}{2}} < \det(\mathcal{M}_{I_n}(x))^{-\frac{1}{2}},$$

or as $\mathbb{C}_n$ continues to increase, eventually $\lambda_n = 0$. Then from (A.7) it follows that $\|S_v\| \geq \beta$, $\forall v$ and (A.6) means that $\mathcal{M}$ *must* act. Consequently $\mathbb{C}_n \to \infty \implies \min_x \det(\mathcal{M}_{I_n}(x))^{-\frac{1}{2}} \to 0$, and that $\exists \{v_n^-\}$ such that $|\kappa(v_n^-)| \to 0$. Thus from (A.4), $\eta_{v_n^-} \to 0$. This in turn requires $\lambda_n \to 0$ from Lemma 2.

Define a sequence $\{v_n\}$ by choosing one vertex from each $v(\hat{\mathcal{T}}_n) \backslash v_n^-$, these are also related to λ_n by Lemma 2, thus $\lambda_n \to 0 \implies \eta_{v_n} \to 0$. It holds that $\mathcal{E}(\hat{\mathcal{T}}_n) \not\to 0 \implies \exists \{v_n\} : \ \eta_{v_n} \not\to 0$. However, $\{v_n\}$ was arbitrary so such a sequence cannot exist, thus for any infinite length sequence $\{\hat{\mathcal{T}}_n\}$, $\{\sum_{v(\hat{\mathcal{T}}_n)} \eta_v\} \to 0$.

Now consider $\{\overline{\mathcal{T}}_n\}$. Given (A.2), (A.3) and $\lambda_n = 0$, $\forall \overline{\mathcal{T}}_n$, we can apply Lemma 5, from which it follows that $s_v > 0$ and $\|S_v\|_F = \frac{1}{\alpha}$ for all vertices where $\eta_v > 0$. (A.7) means that $\|S_v\|_F \geq \beta$, and thus from (A.6) that $\mathcal{M}$ will return a new mesh that better conforms to the metric request. Then (A.4) and Lemma 4 mean that $\{\sum_{v(\overline{\mathcal{T}}_n)} \eta_v\} \to 0$ for any infinite sequence.

This then completes the proof because if $\{\hat{\mathcal{T}}_n\}$ or $\{\overline{\mathcal{T}}_n\}$ are infinite, then $\lim_{n \to \infty} \sum_{v(\mathcal{T}_n)} \eta_v = 0$, and from the construction of $\{\mathcal{T}_n\}$ at least one must be infinite, or $\exists N : \sum_{v(\mathcal{T}_N)} \eta_v = 0$. Then from (A.5) it follows that $\{\sum_{v(\mathcal{T}_n)} \eta_v\} \to 0 \implies \{\mathcal{E}_n\} \to 0$. ☐

Theorem 1 requires that the local indicator, η_v, is a reliable estimate of ϵ_v. This reliability of the local error indicator ensures that controlling η_v will ultimately control the true local error, ϵ_v. Theorem 1 could also be written in terms of controlling the sum of the error indicators $\sum_{v(\mathcal{T}_n)} \eta_v$. This approach is taken in Carstensen *et al.*[84] as they take the position that only η_v is generally available.

[84] Carstensen *et al.*, *Axioms of adaptivity*. 2014

Many estimators, such as residual estimators for energy norm error[36] can be shown to be reliable. Some output error estimators can also be shown to be reliable for some problems. This is generally achieved through energy norm error estimation on the primal and adjoint simultaneously[4,54–57]. However for many problems reliable output error estimators are not available, generally this occurs when a PDE is not coercive. Alternative output error estimators for such problems exist, for instance the dual weighted residual estimator[45], which is not reliable in general. The construction of a reliable dual weighted residual error estimator remains an open research problem[107].

Proof of Optimal Convergence

Theorem 1 proves that the step matrix optimization algorithm will converge to zero error for any divergent sequence $\{\mathbb{C}_n\}$ but it cannot make any statement about the *rate of convergence*. This is necessary because of the lack of restriction on $\{\mathbb{C}_n\}$. If the global cost constraint remains inactive throughout, then the asymptotic rate of convergence comes from the global regularity of the problem. This is because as all regions are refining at the same rate, those with the slowest rate of convergence (least regularity) will eventually dominate the error.

This scenario is the opposite of the practical scenario. In general the sequence $\{\mathbb{C}_n\}$ is chosen to increase slowly in order that the mesh is optimized as much as possible at lower cost levels. Thus for practical sequences, the global cost constraint is almost always active. In this scenario, the coupling introduced by Lemma 2 means that η_v for any two vertices are coupled. As a result of this coupling, the meshes will have a grading that minimizes the modeled error. Graded meshes are known to be required to achieve higher-order convergence for solutions with singularities[108], and in Theorem 2 we show that the MAFEM described in this work achieves this grading as a direct consequence of the optimization process.

[108] Schwab, *p- and hp- Finite Element Methods*. 1998

In order to prove the result, we first introduce various definitions for partitioning a mesh.

Definition 7 (Mesh subsets)**.** *Let* $K \subseteq \mathcal{T}$ *and* $K \cup K^c \equiv \mathcal{T}$.

Define $\mathring{v}(K) \subseteq v(K)$ *by* $\mathring{v}(K) \equiv v(K) \backslash v(K^c)$.
Define $\mathring{K} \subseteq K$ *by* $v(\mathring{K}) \equiv \mathring{v}(K)$.
Define $\overline{K} \supseteq K$ *by* $\mathring{v}(\overline{K}) \equiv v(K)$.

Definition 7 means that $\overline{K}$ is K with an additional 'layer' of elements. This layer of elements can be denoted $\overline{K} \cap \overline{K^c} \neq \emptyset$. Examples of these definitions are shown in figure 3.3 for a one dimensional mesh of 4 elements. The cost of a given subset $\mathbb{C}_K$ is defined as the cost per element multiplied by the number of elements in that subset, thus $\mathbb{C}_K + \mathbb{C}_{K^c} \equiv \mathbb{C}_{\mathcal{T}}$.

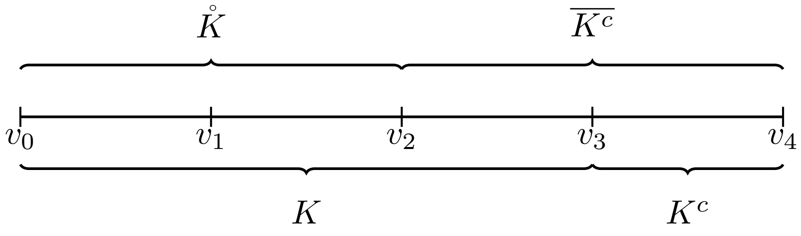


Figure 3.3: Examples of mesh subsets for a $1D$ domain of 4 elements.

Definition 7 can be used to separate the mesh into collections of elements with a higher order convergence rate, and those with a lower order convergence rate.

Theorem 2 (Rate of Convergence of Step Matrix Optimization)**.** *Let $\{\mathcal{T}_n\}$ be a sequence of non-degenerate meshes from a step optimization satisfying the assumptions of Theorem 1. Assuming*

$$\exists N : \mathbb{C}_{n+1} > \mathbb{C}_n \implies \lambda_n > 0,\ \mu_v = 0,\ \forall v \in v(\mathcal{T}_n),\ \forall n > N,$$

Then

$$\mathcal{H} = \sum_{v(\mathcal{T})} \eta_v \lesssim \mathbb{C}_{K_m}^{-\frac{m}{d}} (1 + \frac{\mathbb{C}_{K_s}}{\mathbb{C}_{K_m}}), \tag{3.51}$$

where $K_m \subseteq \mathcal{T}$ is the largest subset such that

$$\eta_v \lesssim h_v^{m_v+d}, \quad \forall v \in \mathring{v}(K_m),\ m_v \geq m, \tag{3.52}$$

where $m > 0$ is a rate of convergence, $K_s \equiv \overline{K_m^c}$ and $h_v = \max_{\kappa(v)} h_\kappa$ is a length scale for $\kappa(v)$.

Proof. Let $K \subset \mathcal{T}$. Quasi-stationarity and $\mu_v = 0$ means

$$w_v \eta_v \equiv C_\rho \lambda_n \sum_{\kappa(v)} \rho_\kappa(\{S_v\}), \quad \forall v \in v(\mathcal{T}_n), \tag{3.53}$$

which when summed over $\mathring{v}(K)$ gives

$$\sum_{\mathring{v}(K)} w_v \eta_v = \lambda C_\rho \sum_{\mathring{v}(K)} \sum_{\kappa(v)} \rho_\kappa(\{S_v\}) \tag{3.54}$$

$$\lesssim \lambda |v(\kappa)| \sum_K \rho_\kappa(\{S_v\}) \tag{3.55}$$

$$\gtrsim \lambda \sum_K \rho_\kappa(\{S_v\}), \tag{3.56}$$

where the inqualities in equations (3.55) and (3.56) are both with respect to equation (3.54). Lemma 2 bounds w_v and $\|S_v\|_F$, giving

$$\mathcal{H}_K = \sum_{\mathring{v}(K)} \eta_v \sim \lambda \mathbb{C}_K. \tag{3.57}$$

Using AI interpolation[105],

$$\det(\mathcal{M}(x)) \equiv \prod_{v(\kappa)} \det(\mathcal{M}_v)^{b_v(x)}, \tag{3.58}$$

where $b_v(x)$ are the barycentric coordinates where $x \in \kappa$. The metric cost functional in region K is written

$$\mathbb{C}_K(\mathcal{M}) = \int_K c_p \sqrt{\det(\mathcal{M}(x))} dx \tag{3.59}$$

$$\leq c_p |K| \max_{v(K)} \sqrt{\det(\mathcal{M}_v)}. \tag{3.60}$$

We are free to define h_v as a local length scale, thus

$$h_v \sim \det(\mathcal{M}_v)^{-\frac{1}{2d}}. \tag{3.61}$$

Define $v^+ = \arg\max_{v(K_m)} \sqrt{\det(\mathcal{M}_v)}$ then noting $|K_m| \leq |\Omega|$ and applying equations (3.52) and (3.60) gives

$$\mathbb{C}_K \lesssim |\Omega| \eta_{v^+}^{-\frac{d}{(m+d)}} \tag{3.62}$$

$$\eta_{v^+} \lesssim \mathbb{C}_{K_m}^{-\frac{m+d}{d}}. \tag{3.63}$$

Summing over $\mathring{v}(K_m)$ and noting $|\mathring{v}(K_m)| \sim \mathbb{C}_{K_m}$ from non-degeneracy then gives

$$\mathcal{H}_{K_m} = \sum_{\mathring{v}(K_m)} \eta_v \tag{3.64}$$

$$\lesssim |\mathring{v}(K_m)| \eta_{v^+} \tag{3.65}$$

$$\lesssim \mathbb{C}_{K_m}^{-\frac{m}{d}}. \tag{3.66}$$

Define $K_s \equiv \overline{K_m^c}$ (note $K_m \cap K_s \neq \emptyset$), this partitions the global error indicator

$$\sum_{v(\mathcal{T})} \eta_v = \sum_{\mathring{v}(K_s)} \eta_v + \sum_{\mathring{v}(K_m)} \eta_v \tag{3.67}$$

$$= \mathcal{H}_{K_s} + \mathcal{H}_{K_m}. \tag{3.68}$$

Applying equation (3.57) to both gives

$$\mathcal{H}_{K_s} \sim \lambda \mathbb{C}_{K_s} \tag{3.69}$$

$$\mathcal{H}_{K_m} \sim \lambda \mathbb{C}_{K_m}. \tag{3.70}$$

[105] Arsigny *et al.*, *Geometric means in a novel vector space structure on symmetric positive-definite matrices*. 2007

Equation (3.66) implies

$$\lambda \lesssim \mathcal{H}_{K_m} \mathbb{C}_{K_m}^{-1} = \mathbb{C}_{K_m}^{-\frac{m+d}{d}}, \tag{3.71}$$

and therefore

$$\mathcal{H}_{K_s} \lesssim \mathbb{C}_{K_m}^{-\frac{m+d}{d}} \mathbb{C}_{K_s}. \tag{3.72}$$

The global error indicator is then bounded

$$\mathcal{H} \lesssim \mathbb{C}_{K_m}^{-\frac{m}{d}} (1 + \frac{\mathbb{C}_{K_s}}{\mathbb{C}_{K_m}}). \tag{3.73}$$

□

Theorem 2 means that if any singularities in the solution remain bounded by a finite number of elements, then a MAFEM will achieve the highest asymptotic order of accuracy possible given the test space p and the effectiveness of the error indicator η_v. For instance if performing L^2 projection of a function with a corner singularity the rate of convergence will be $p+1$. This is because the effect of the singularity is contained within the finite number of elements attached to the corner and their neighbours.

3.6 Numerical Results

We now demonstrate the performance of a MAFEM of the form described by Definition 5. The algorithm used to construct the rate matrices used in the local models, R_v, is the Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm for continuous discretization, see Chapter 2. The R_v are fit to data coming from 'local solves' which are used to model the effect of local mesh perturbations on the local error estimates η_v.

The PDE is Laplace's equation

$$-\Delta u = 0, \quad x \in \Omega, \tag{3.74}$$

where $\Omega \equiv [-1,1] \times [-1,1] \backslash [-1,0] \times [-1,0]$ is a re-entrant corner. The boundary conditions on $\partial\Omega$ are Dirichlet and are chosen to match the exact solution

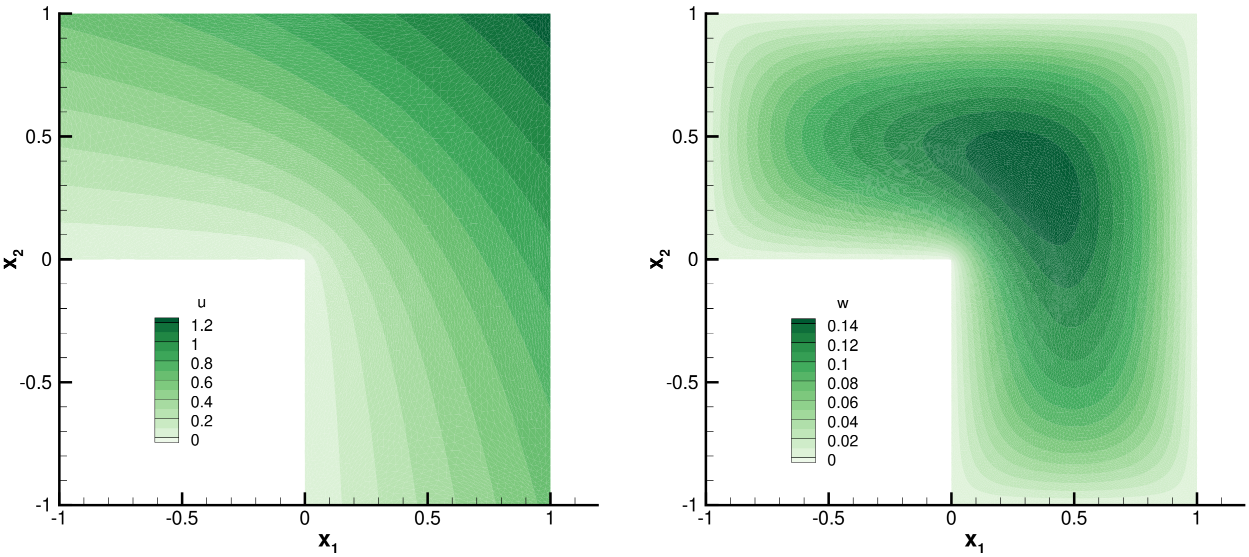
$$u(x_1, x_2) = (x_1^2 + x_2^2)^{\frac{\alpha}{2}} \sin(\alpha(\tan^{-1}(\frac{x_2}{x_1}) + \theta_0)) \tag{3.75}$$

where $\alpha = \frac{2}{3}$ and $\theta_0 = \frac{\pi}{2}$. The output functional of interest is the integral of the solution $\int_\Omega u dx$ with exact solution $\mathcal{J}(u) = \frac{9}{20}\left(1 + 2^{\frac{4}{3}}\right)$. The output functional has a corresponding adjoint equation,

$$-\Delta w - 1 = 0, \quad x \in \Omega, \tag{3.76}$$

with homogeneous Dirichlet boundary conditions. The primal and adjoint both have a geometric singularity at the re-entrant corner of the domain, as seen in Figure 3.4. The expected convergence rate in the H^1 norm is $O(h^{\frac{2}{3}})$ for both primal and adjoint, giving an asymptotic rate of convergence for the output of $O(h^{\frac{4}{3}})$ for uniform refinement with an adjoint consistent discretization[109].

109. Strang *et al.*, *An Analysis of the Finite Element Method*. 1988



(a) Primal

(b) Adjoint

Figure 3.4: The primal and adjoint solutions both exhibit singularities on the re-entrant corner of the domain.

The metric-mesher used to generate the meshes from a given metric request is `avro`[70], which employees a cavity-based mesh operator in order to perform local mesh operations, where the cavity is defined as $\kappa(v)$. The action of the cavity operator can, in a significant simplification, be split into its action for two different situations. If there exist edge lengths outside of the quasi unit range attached to a vertex, then operations equivalent to edge splits, collapses and/or swaps may occur within its cavity. If the local mesh around a vertex is *quasi-unit*, both in terms of edge lengths and element quality[2], then the cavity operation will be equivalent to vertex smoothing.

If we consider 6 equilateral triangles attached to a vertex in two dimensions, the metric at the central vertex is the identity $\mathcal{M}_v \equiv \mathcal{I}$. An isotropic step refinement, $S_v = s_v\mathcal{I}$, would then need to satisfy $\|S_v\|_F \geq \sqrt{d}\log(2)$ before edge splits would occur and progress be made. According to Definition 4, `avro` is then a $\sqrt{d}\log(2)$ beta contractive metric mesher.

A quadratic regularizer coefficient of $\alpha = \frac{1}{1.1\sqrt{d}\log(2)}$ is used, which satisfies the assumption of Theorem 1 with $\epsilon_\alpha = 0.1\beta$.

The optimization is repeated at each cost level for 20 iterations, by which stage $\mu_v = 0,\ \forall v$, thereby satisfying the assumptions of Theo-

[2] Mesh quality is defined by Ibanez *et al.*[110] as

$$Q \equiv \frac{\left(\frac{|\kappa|\sqrt{\det(\mathcal{M}_{\max})}}{|\hat{\kappa}|}\right)^{\frac{2}{d}}}{\frac{1}{|\mathcal{E}(\kappa)|}\sum_{\mathcal{E}(\kappa)} l_{\mathcal{M}_{\max}}(e)}$$

where

$$\mathcal{M}_{\max} \equiv \underset{v(\kappa)}{\arg\max} \det(\mathcal{M}_v).$$

Mesh generators will often seek to keep this number above a specified threshold in addition to achieving *quasi-unity*. This is in order to avoid degenerate simplicies which can occur with the relaxed notion of unity.

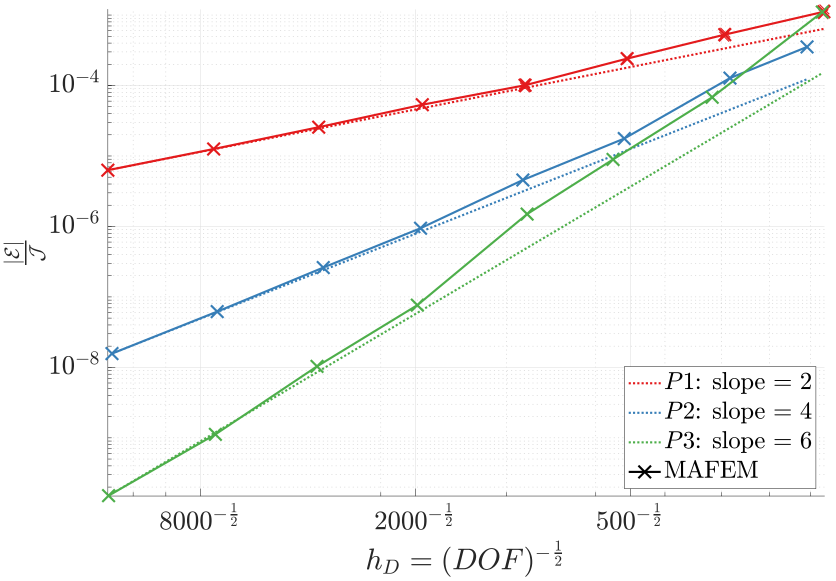


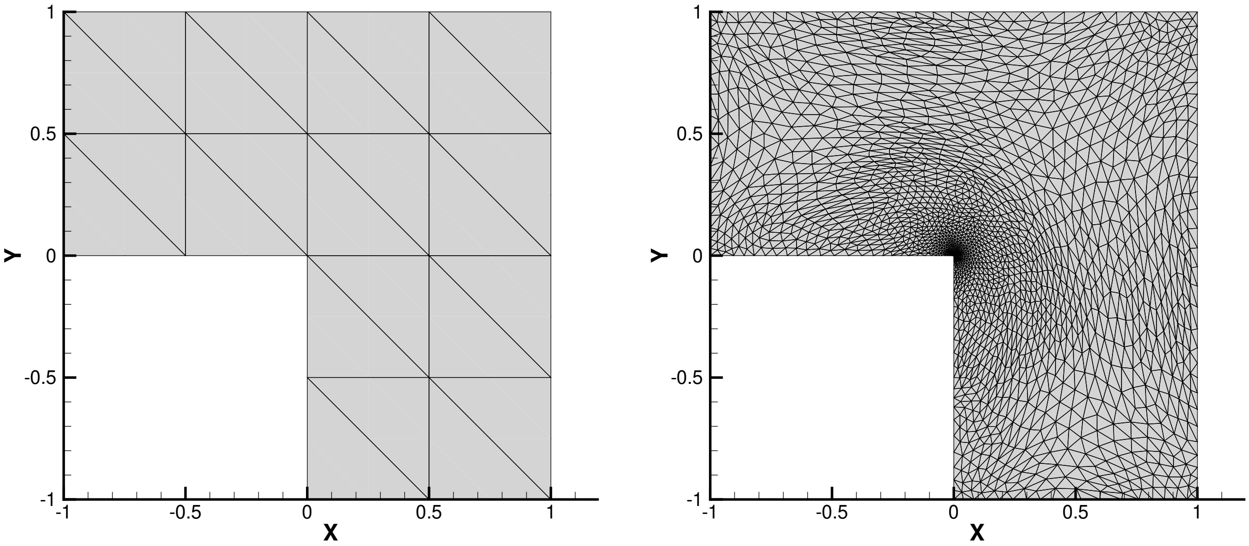
Figure 3.5: Convergence results for the case of $-\Delta u - f = 0$ with output functional $\mathcal{J}(u) = \int_{\Omega} u$. The y-axis is the true error computed using the exact solution.

rem 2. Figure 3.5 shows the convergence results for the normalized error, computed using the final mesh for each cost level. The rate slopes shown in the plot correspond to $2p$, the highest order asymptotic output convergence expected of a p-th order discretization of Poisson's equation[51], in line with result of Theorem 2.

A selection of meshes for approximately 2000 DOFs are shown in figure 3.6 and exhibit significant grading in the mesh density. This grading is expected in order to capture the singularity in the solution and the adjoint focused at the re-entrant corner.

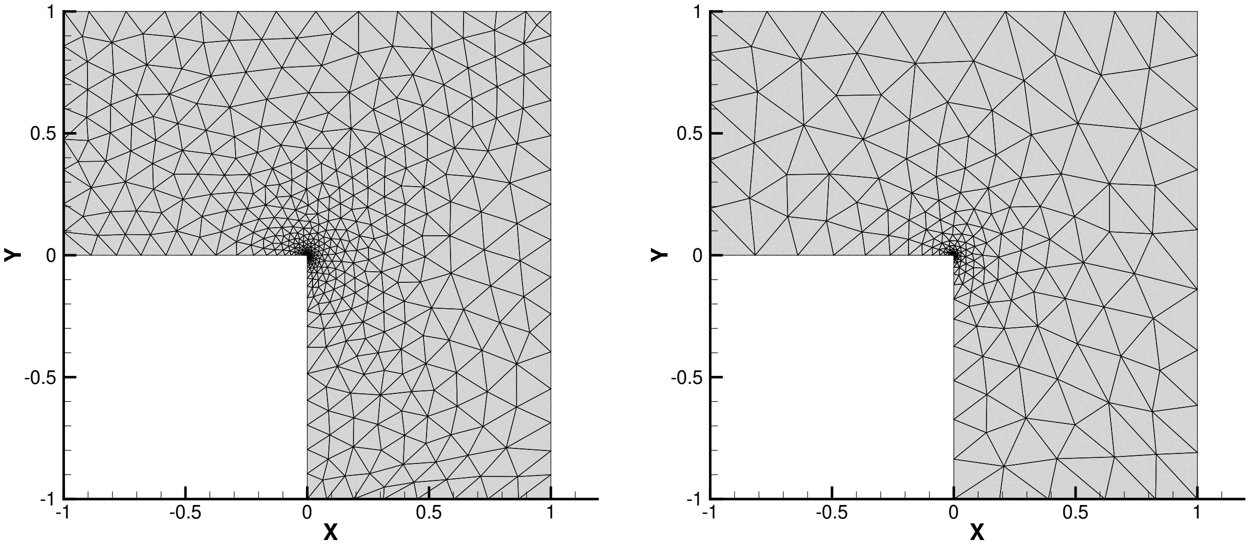
---

[51] Carson *et al.*, *Analysis of Output-Based Error Estimation for Finite Element Methods*. 2017



(a) Initial Grid

(b) $p = 1$



(c) $p = 2$

(d) $p = 3$

Figure 3.6: The initial mesh along with the some example meshes for approximately 2000 DOFs

Chapter 4

Application to Navier-Stokes equation

This chapter applies the algorithm developed in Chapter 2 to more complex physics than demonstrated previously: the compressible Navier Stokes equations. The Navier Stokes equations are highly non-linear, and can exhibit a wide range of scales, anisotropic features, and develop discontinuities. Typical geometries can have singular features and be relatively complex.

As a result, designing efficient meshes for higher-order methods can be difficult. An appropriate mesh for one configuration may be ineffective if boundary conditions or geometry change, as can be typical in the design phase or during off-design analysis. The complexity of the physics can also mean that lower order methods can require considerable computational resources to arrive at a satisfactory solution. As such, automated mesh adaptation combined with higher order methods presents both great difficulties and great possible benefits for Navier Stokes simulations.

Governing Equations

The compressible Navier Stokes equations are a non-linear system of conservation equations that can be written

$$u_t + \nabla \cdot (\mathcal{F}(u) - \mathcal{G}(u, \nabla u)) = 0 \qquad \text{on } \Omega,$$

where $u \equiv [\rho, \rho V_j, \rho E]^T \in \mathbb{R}^{d+2}$ is the conservative state vector, ρ is the density, V_j is the j-th component of the velocity and E is the total internal energy. The advective flux in the i-th coordinate direction is given by

$$\mathcal{F}_i(u) \equiv \begin{pmatrix} \rho V_i \\ \rho V_j V_i + \delta_{ij} p \\ \rho H V_i \end{pmatrix},$$

where δ_{ij} is the Kronecker delta product. The pressure, p, and total enthalpy, H, are given by

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho V_i V_i\right)$$
$$H = E + \frac{p}{\rho},$$

and γ is the ratio of specific heats. The viscous flux in the i-th coordinate direction is given by

$$\mathcal{G}_i(u, \nabla u) \equiv \begin{pmatrix} 0 \\ \tau_{ij} \\ k_T \frac{\partial T}{\partial x_i} + \tau_{ij} V_j \end{pmatrix},$$

where k_T is the thermal conductivity, $T = \frac{p}{\rho R}$ is the temperature and R the gas constant. The viscous stress is given by

$$\tau_{ij} = \mu \left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i}\right) + \delta_{ij} \lambda \frac{\partial V_k}{\partial x_k},$$

where $\lambda = -\frac{2}{3}\mu$ is the bulk viscosity and μ the dynamic viscosity which is a function of temperature, however in this work we use constant μ for simplicity. The thermal conductivity is related to the dynamic viscosity μ by

$$k_T \equiv c_p \frac{\mu}{\mathrm{Pr}},$$

where $c_p = \frac{\gamma R}{\gamma - 1}$ is the ratio of specific heat at constant pressure and $\mathrm{Pr} = 0.72$ is the Prandtl number.

4.1 Variational Multi-Scale with Discontinuous sub-scales

The Reynolds number measures the ratio of inertial forces to viscous forces within a fluid and is defined as $\mathrm{Re} \equiv \frac{\rho |\mathbf{V}| L}{\mu}$, where ρ is the density, $\mathbf{V}$ is the velocity, μ is the dynamic viscosity and L is a characteristic length scale. In general aerodynamic flows the Reynolds number is much greater than one, and the flow is said to be advection-dominated.

Continuous Galerkin discretization, though requiring fewer DOFs on a given mesh when compared to Discontinuous Galerkin discretization, suffers from stability issues for advection-dominated PDEs. The

CG discretization derives its stability from the coercivity induced by diffusive operators. However, for advection-dominated PDEs, the diffusion can be insufficient as the constant in the stability bound increases as a function of the Peclet number. The DG discretization derives stability for advection-dominated flows through upwinding of the advective flux across element traces. A CG solution is C^0 continuous, thus upwinding of the advective flux across element traces is not possible.

An alternative means of providing the required advective stability for CG is through the addition of element-wise weighted strong form residuals to the global residual expression. Three popular methods of achieving this are Streamline Upwind Petrov Galerkin (SUPG)[23], Galerkin Least Squares (GLS)[25] and Variational Multi-Scale (VMS)[26]. Each of these methods consists of adding a weighted primal strong form residual to the global residual, $\mathcal{R}_h(v,u)$, giving the stabilized residual

$$\mathcal{R}_h^{\text{stab}}(v,u) = (\overline{\mathcal{L}}(w), \tau(\mathcal{L}u - f))_{\mathcal{T}} + \mathcal{R}_h(v,u) \tag{4.1}$$

$$\overline{\mathcal{L}}^{\text{SUPG}}(w) = \mathcal{F}w \tag{4.2}$$

$$\overline{\mathcal{L}}^{\text{GLS}}(w) = \mathcal{L}w \tag{4.3}$$

$$\overline{\mathcal{L}}^{\text{VMS}}(w) = -\mathcal{L}^* w \tag{4.4}$$

where τ is the intrinsic timescale or stabilization parameter, $\mathcal{F}$ is the advective flux Jacobian, $\mathcal{L}$ is the strong form primal operator and $\mathcal{L}^*$ is the adjoint strong form operator. GLS can be shown to be stable but is adjoint inconsistent for $p \geq 2$ and thus unsuitable for higher-order output based adaptation without modifications[111]. VMS is adjoint consistent, however stability analysis shows that the negative sign on the diffusive operator in $-\mathcal{L}^* w$ can result in destabilization[1] for $p \geq 2$ without a careful choice of τ[112,113]. For complex physics, systems of equations with more than one characteristic direction, and physical dimensions greater than one, such τ are generally unknown.

At the core of VMS is the separation of the solution into *resolved* coarse scales and *unresolved* fine scales. The resolved scales are those captured by the discrete solution space, whilst the fine scales are the infinite dimensional components that are not captured in the discrete solution space. VMS attempts to model the effect of these unresolved fine scales on the coarse scales through element-wise approximate Green's functions. The Residual Free Bubble (RFB) method[27] is a similar approach to VMS, which enriches the solution space with so-called residual free bubble functions that satisfy the strong form residual of the PDE. Though originating in different approaches, VMS and RFB are closely related in that both attempt to model the unresolved scales and can be shown to be equivalent[28]. Variational Multi-Scale with

[23] Hughes, *A simple scheme for developing upwind finite elements.* 1978

[25] Hughes *et al.*, *A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations.* 1989

[26] Hughes *et al.*, *Variational Multiscale Analysis: The fine-scale Green's function, projection, optimization, localization, and stabilized methods.* 2007

[111] Cyr *et al.*, *Approaches for adjoint-based a posteriori analysis of stabilized finite element methods.* 2014

[1] Consider linear advection diffusion,

$$\nabla \cdot (\mathbf{a}u - \nu \nabla u) = 0.$$

Then the stability requirement for VMS is written

$$(\mathbf{a}u + \nu \nabla u, \tau(\mathbf{a}u - \nu \nabla u))_{\mathcal{T}} \geq 0.$$

The opposite signs on the two viscous terms means that τ must go to zero in the viscous limit.

[112] Franca *et al.*, *Stabilized finite element methods for the Stokes problem.* 1991

[113] Franca *et al.*, *Stabilized finite element methods: I. Application to the advective-diffusive model.* 1992

[27] Brezzi *et al.*, *Choosing bubbles for advection-diffusion problems.* 1994

[28] Brezzi *et al.*, $b = \int g$. 1997

Discontinuous sub-scales (VMSD) is a new discretization that inherits from both VMS and RFB, and is closely related to the Discontinuous Residual Free Bubble (DRFB) method of Sangalli[30]. The main difference compared to DRFB is that VMSD includes the viscous flux and stabilization in the local fine scale problems.

The discontinuous scales in VMSD are stabilized using upwinding and Symmetric Interior Penalty (SIP)[114], however the solution jump is with respect to the continuous solution on the trace. Defining the jump in this manner allows the discontinuous scales to be decoupled. Consequently the discontinuous perturbation can be statically eliminated in the global system, resulting in the same global stencil as an unstabilized CG method. For a more complete description of the discretization, we refer to Arthur Huang's thesis[115].

[30] Sangalli, *A discontinuous residual-free bubble method for advection-diffusion problems.* 2004

[114] Douglas Jr. *et al.*, *Interior Penalty Procedures for Elliptic and Parabolic Galerkin Meethods.* 1976

[115] Huang, *An Adaptive Variational Mutltiscale Method with Discontinuous Subscales for Aerodynamics Flows.* 2019

Discretization

The multi-scale methodology separates the solution

$$u = \bar{u} + u' \tag{4.5}$$

where $u \in \mathcal{V}$ is the full solution, $\bar{u} \in \bar{\mathcal{V}}$ is the coarse scale solution, and $u' \in \mathcal{V}' \equiv \mathcal{V} \backslash \bar{\mathcal{V}}$ is the fine scale solution. This can be written $u \in \bar{\mathcal{V}} \oplus \mathcal{V}'$ where $\oplus$ requires the definition of projectors $\Pi : \mathcal{V} \to \bar{\mathcal{V}}$ and $\Pi' \equiv \mathcal{I} - \Pi : \mathcal{V} \to \mathcal{V}'$, thus the component in $\mathcal{V}'$ is what is not captured in $\bar{\mathcal{V}}$. Substituting the decomposition into a standard Galerkin method results in the system

$$(\bar{u}, u') \in \bar{\mathcal{V}} \times \mathcal{V}' : \quad \mathcal{R}(\bar{v}, v'; \bar{u}, u') = 0, \quad \forall (\bar{v}, v') \in \bar{\mathcal{V}} \times \mathcal{V}', \tag{4.6}$$

where the decomposition is also applied to the test functions $v = \bar{v} + v'$, and the residual is given by

$$\mathcal{R}(\bar{v}, v'; \bar{u}, u') = (-\nabla(\bar{v} + v'), \mathcal{F}(\bar{u} + u') - \mathcal{G}(\bar{u} + u', \nabla(\bar{u} + u')))_{\Omega} \tag{4.7a}$$

$$+ \langle \bar{v} + v', (\mathcal{F}(\bar{u} + u') - \mathcal{G}(\bar{u} + u', \nabla(\bar{u} + u'))) \cdot \mathbf{n} \rangle_{\partial\Omega}. \tag{4.7b}$$

Equation (4.7) can be decomposed

$$\mathcal{R}(\bar{v}, v'; \bar{u}, u') = \mathcal{R}_I(\bar{v}; \bar{u}, u') + \mathcal{R}_{II}(v'; \bar{u}, u') + \mathcal{R}_{BC}(\bar{v}, \bar{u})$$

where $\mathcal{R}_{BC}(\cdot, \cdot)$ contains the boundary condition terms. We now associate $\mathcal{V}$ with a particular mesh $\mathcal{T}$, and enrich $\mathcal{V}'$ with element-wise discontinuous functions. The sub-residuals are then defined as

$$\begin{aligned}
\mathcal{R}_I(\bar{v};\bar{u},u') &= \big(-\nabla\bar{v}, \mathcal{F}(\bar{u}+u') - \mathcal{G}(\bar{u}+u', \nabla(\bar{u}+u'))\big)_{\mathcal{T}} \\
&\quad - \langle \mathcal{G}_{\nabla u}^T \nabla\bar{v}\cdot\mathbf{n}, u'\rangle_{\partial\mathcal{T}} \qquad (4.8a) \\
\mathcal{R}_{II}(v';\bar{u},u') &= \big(-\nabla v', \mathcal{F}(\bar{u}+u') - \mathcal{G}(\bar{u}+u', \nabla(\bar{u}+u'))\big)_{\mathcal{T}} \\
&\quad + \langle v', \big(\mathcal{F}(\bar{u}+u') - \mathcal{G}(\bar{u}+u', \nabla(\bar{u}+u'))\big)\cdot\mathbf{n}\rangle_{\partial\mathcal{T}} \\
&\quad + \langle \mathcal{N}v' - \mathcal{G}_{\nabla u}^T\nabla v'\cdot\mathbf{n}, u'\rangle_{\partial\mathcal{T}} \\
&\quad + \langle v', |\mathcal{F}_u\cdot\mathbf{n}|u' - \mathcal{G}_{\nabla u}\nabla u'\cdot\mathbf{n}\rangle_{\partial\mathcal{T}} \qquad (4.8b) \\
\mathcal{R}_{BC}(\bar{v},\bar{u}) &= \langle \bar{v}, (\mathcal{F}(u_B) - \mathcal{G}(u_B,\nabla\bar{u}))\cdot\mathbf{n}\rangle_{\partial\mathcal{T}\cap\partial\Omega} \\
&\quad + \langle \mathcal{N}\bar{v} - \mathcal{G}_{\nabla u}^T\nabla\bar{v}\cdot\mathbf{n}, \bar{u}-u_B\rangle_{\partial\mathcal{T}\cap\partial\Omega} \qquad (4.8c)
\end{aligned}$$

where

$$(\cdot,\cdot)_{\mathcal{T}} \equiv \sum_{\kappa(\mathcal{T})}(\cdot,\cdot)_\kappa, \qquad \langle\cdot,\cdot\rangle_{\partial\mathcal{T}} \equiv \sum_{\kappa(\mathcal{T})}\langle\cdot,\cdot\rangle_{\partial\kappa},$$

$\mathbf{n}$ is the outwards facing normal, $\mathcal{N} > 0$ is a Nitsche parameter and the linearizations $\mathcal{F}_u$ and $\mathcal{G}_{\nabla u}$ are taken about $\bar{u}+u'$ in $\mathcal{R}_{II}$, $\bar{u}$ in $\mathcal{R}_{BC}$ and u_B is a boundary state. The enrichment of $\mathcal{V}'$ with $L^2(\Omega)$ is the point of divergence with respect to VMS, as this results in the element trace integrals in equation (4.8b). It is this enrichment of $\mathcal{V}'$ with element-wise discontinuous functions that enables the usage of upwinding in order to stabilize the advective flux. The definition of $\mathcal{V}'$ in terms of $\bar{\mathcal{V}}$ means that if $\bar{\mathcal{V}} \equiv H^1(\Omega)$ and $u \in \mathcal{V} \equiv H^1(\Omega)$, namely that *all* the scales are resolved by $\bar{\mathcal{V}}$, then the perturbation satisfies $u' = 0$.

Finite dimensional spaces $\bar{\mathcal{V}}_{h\bar{p}}$ and $\mathcal{V}'_{hp'}$ are then defined as

$$\bar{\mathcal{V}}_{h\bar{p}} \equiv \{v \in C^0(\Omega):\ v|_\kappa \in \mathbb{P}^{\bar{p}}(\kappa),\ \forall\kappa\in\mathcal{T}\} \qquad (4.9)$$

$$\mathcal{V}'_{hp'} \equiv \{v \in L^2(\Omega)\backslash\bar{\mathcal{V}}_{h\bar{p}}:\ v|_\kappa \in \mathbb{P}^{p'}(\kappa),\ \forall\kappa\in\mathcal{T}\}. \qquad (4.10)$$

The exclusion of the space $\bar{\mathcal{V}}_{h\bar{p}}$ from $\mathcal{V}'_{hp'}$ means that any elemental bubble functions in $\bar{\mathcal{V}}_{h\bar{p}}$ cannot appear in $\mathcal{V}'_{hp'}$[2]. In this work we fix the perturbation polynomial order, p', to be equal to that used in the continuous scales, $\bar{p}$, though this need not be the case[3]. The corresponding discrete test and trial functions are then inserted into the residuals of Equation (4.8), and the Nitsche parameter is defined as

$$\mathcal{N} = \frac{8p^2\mathbf{n}^T\mathcal{G}_{\nabla u}\mathbf{n}}{h}$$

where $h = \frac{|\kappa|}{|f|}$ is a length scale for trace $f \in \partial\kappa$, $\mathbf{n}$ is the unit normal, the p^2 scaling comes from a scaling analysis and the 8 is a heuristic.

Equation (4.8b) employs upwinding between $u'_{hp'}$ and $\bar{u}_{h\bar{p}}$, which gives it advective stability. However, posing the jumps between $\bar{u}_{h\bar{p}} + u'_{hp'}$ and $\bar{u}_{h\bar{p}}$, rather $\bar{u}_{h\bar{p}} + u'_{hp'}$ in the adjacent element, means the $u'_{hp'}$ can be statically eliminated. VMSD is closely related to the Embedded

[2] Such bubble functions appear in the polynomials when $p > d$, where d is the physical dimension.

[3] A particularly interesting possibility is the choice of $p' = 0$ which is very closely related to classical stabilized methods.

Discontinuous Galerkin (EDG) discretization, the difference being that for EDG the continuous solution is restricted to $\partial\mathcal{T}$ rather than extending into the element interior. For additional details of the relationship between VMSD and EDG, we refer the reader to Arthur Huang's PhD thesis[115].

4.2 *Delta Wing*

A three dimensional laminar delta wing test case was described by Wang *et al.*[73] and has been used extensively in the higher order literature to evaluate the performance of higher-order methods and adaptive algorithms[7,75,95]. The geometry is modified from the original description of Klaij, van der Vegt and van der Ven[116] based on the experiment by Riley and Lowson[117].

The freestream conditions for the flow are Mach number of 0.3, Reynolds number of 4000 using the root chord as the length scale, and 12.5° angle of attack. The output functional driving the adaptation is the drag coefficient, $C_d = \frac{D}{\frac{1}{2}\rho|\mathbf{V}|^2 S_{ref}}$. The boundary condition on the surface of the wing is an isothermal no-slip condition, and the wall temperature is equal to the freestream value used in the far-field boundary condition. A precise description of the geometry is given in Park *et al.*[75].

The non-linear system of equations is solved using a Pseudo-Time Continuation (PTC) damped Newton's method with line search. The PTC algorithm computes an element local time step based on the characteristic speed, element size and a Courant-Friedichs-Lewy (CFL) number, this CFL number is driven towards infinity in order to recover Newton-like convergence rates[4]. The Portable Extensible Toolkit for Scientific computation (PETSc)[118] is used to solve the linear system at each step of the PTC algorithm. The iterative solver is restarted Generalized Minimum Residual (GMRES) preconditioned using an Incomplete Lower Upper (ILU) factorization with fill-in of 4 with blocking occuring at the system of equation level. The parallel preconditioner used is a restricted additive Schwarz preconditioner, utilizing a single layer of overlap. The adjoint system is solved with the same linear solver used for the primal solution.

Figure 4.1 shows the surface pressure distribution and slices through the Mach number field for a $p = 2$ adaptation, demonstrating the expected vortex roll up starting from the leading edge of the airfoil. Figure 4.2 shows the surface pressure distributions from the top and bottom views, with and without the mesh for a $p = 1$ adaptation.

Comparing Figures 4.2a and 4.2b, the adaptation has focused on a stream-wise feature on the top surface, that is inset from the leading edge. Figure 4.4 shows slices of $\omega \cdot \mathbf{V}$ and demonstrates that the adap-

---

115. Huang, *An Adaptive Variational Mutltiscale Method with Discontinuous Subscales for Aerodynamics Flows*. 2019

73. Wang *et al.*, *High-order CFD methods: current status and perspective*. 2013

117. Riley *et al.*, *Development of a three-dimensional free shear layer*. 1998

75. Park *et al.*, *Verification of Unstructured Grid Adaptation Components*. 2019

[4] The particular PTC controller comes from private correspondence with Dr. Steven Allmaras, and is available in Appendix C.1

118. Balay *et al.*, *PETSc Users Manual*. 2004

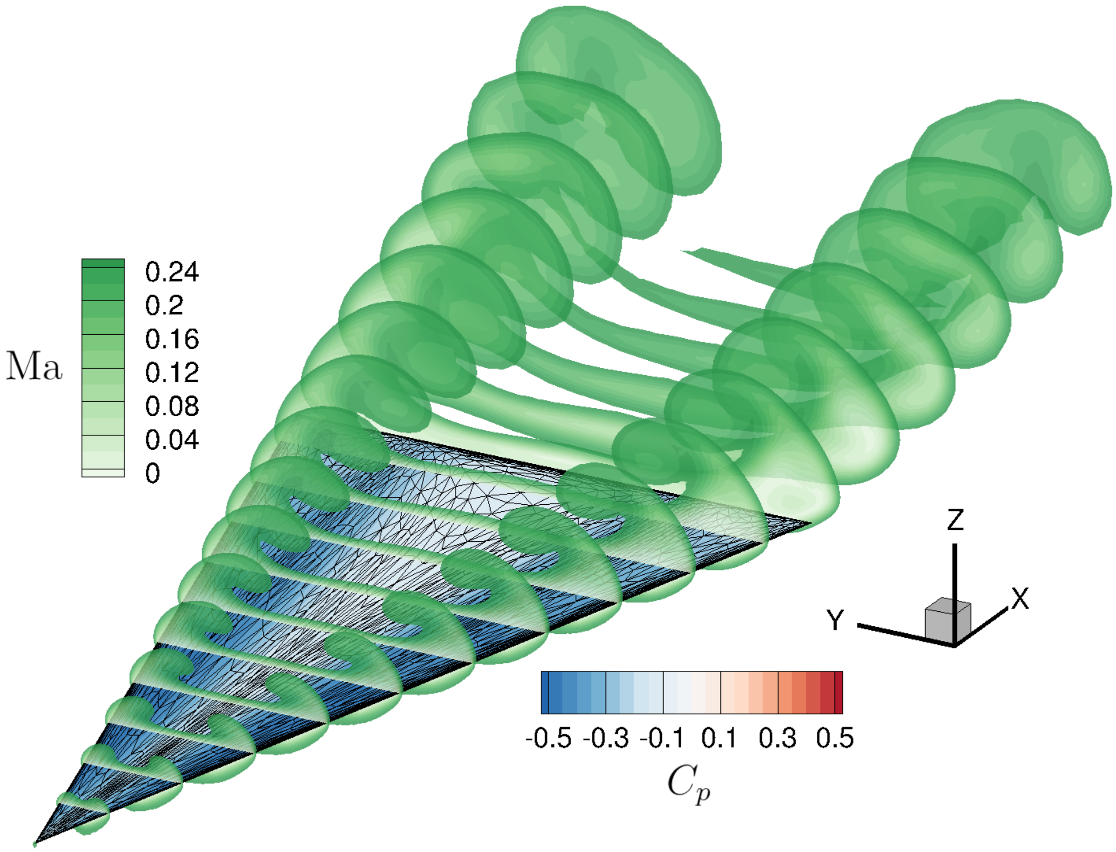
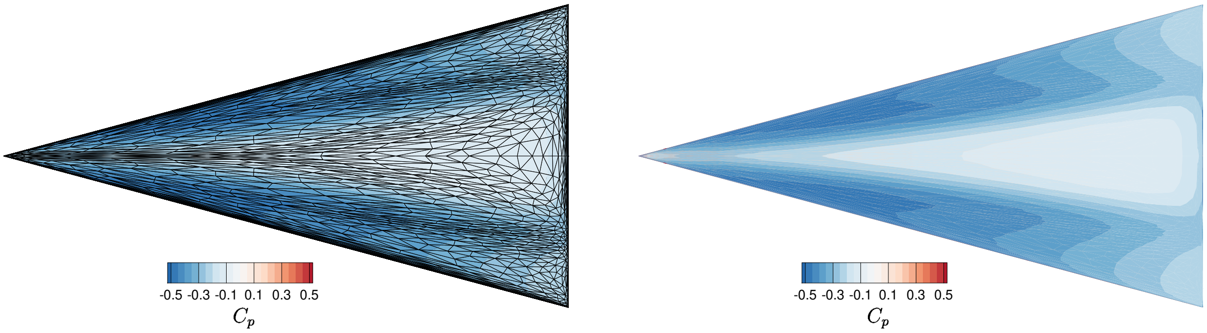
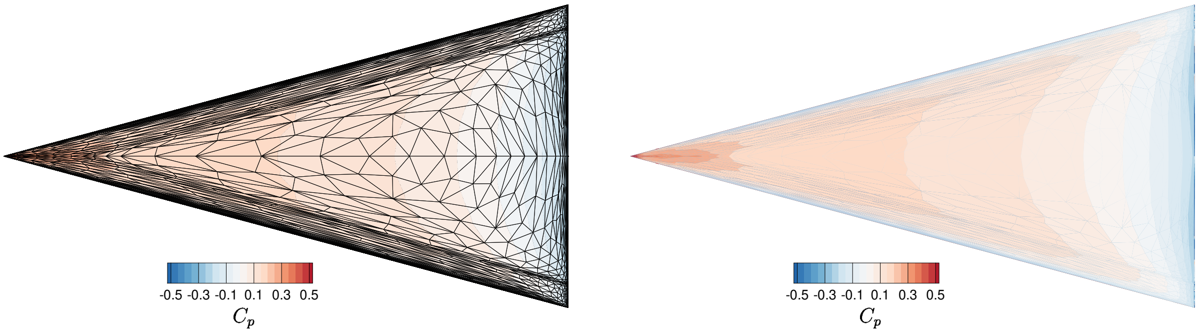


Figure 4.1: Surface pressure distribution and Mach number slices for a VMSD drag output adaptation with $p = 2$ and 80×10^3 DOF



(a) Top surface mesh

(b) Top surface pressure coefficient



(c) Bottom surface mesh

(d) Bottom surface pressure coefficient

Figure 4.2: Surface pressure coefficients for a VMSD drag output adaptation with $p = 1$ and 50×10^3 DOF

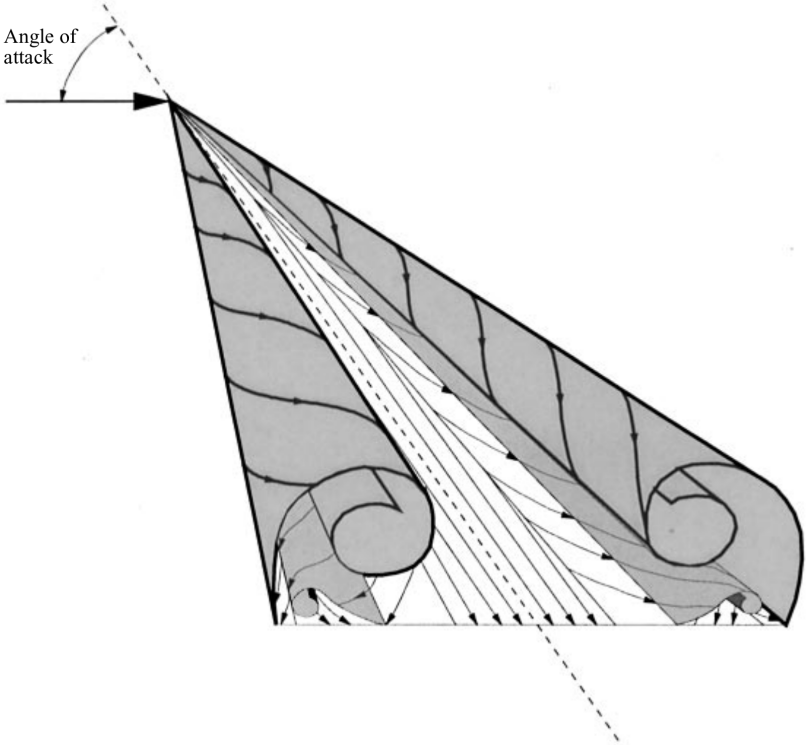


Figure 4.3: The vortical flow field over a delta wing at high angle of attack, reproduced from the paper of Riley and Lowson[117]

tation has focused on the location where the primary vortex intersects with the top surface. This is also the starting location for the secondary vortex that rolls up underneath the primary vortex, seen in Figure 4.3 and Figure 4.4.

Figures 4.2c and 4.2d show the bottom surface of the delta wing, with the chamfered geometry visible on the thin surfaces just inset from the leading edge. The dense anisotropic mesh on the chamfered surfaces, and in the vicinity of the trailing edge, contrasts with the relatively coarse isotropic mesh across the main section of the bottom surface.

4.3 Comparison with Discontinuous Galerkin

In this section the C-MOESS algorithm with the VMSD discretization is compared against the D-MOESS algorithm with the DG discretization utilizing Bassi and Rebay's second viscous stabilization[50]. The error estimation and error modeling for DGBR2/D-MOESS are both performed elementally, in particular the local error model incorporates a quadratic regularization similar to that used in the vertex models[5].

$$\eta_\kappa(\{S_v\}) = \eta_\kappa e^{\mathrm{tr}(R_\kappa S_\kappa) + \frac{\alpha_{DG}\|R_\kappa\|_F}{2(d+1)} \sum_{v(\kappa)} \|S_v\|_F^2}$$

50. Bassi *et al.*, *GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations*. 2000

[5] The smaller value of α for DG is because the regularization takes a weighted average of the attached R_κ directions. The unconstrained stationarity condition for S_v with the elemental model gives

$$S_v = -\frac{1}{\alpha_{DG}} \sum_{\kappa(v)} w_\kappa \frac{R_\kappa}{\|R_\kappa\|_F}$$

where $\sum_{\kappa(v)} w_\kappa = 1$, $w_\kappa = \frac{\eta_\kappa e^{X_\kappa}}{\sum_{\kappa(v)} \eta_\kappa e^{X_\kappa}}$ and $X_\kappa \equiv \mathrm{tr}\left(R_\kappa S_\kappa\right) + \frac{\alpha_{DG}\|R_\kappa\|}{2(d+1)} \sum_{\kappa(v)} \|S_v\|_F^2$.

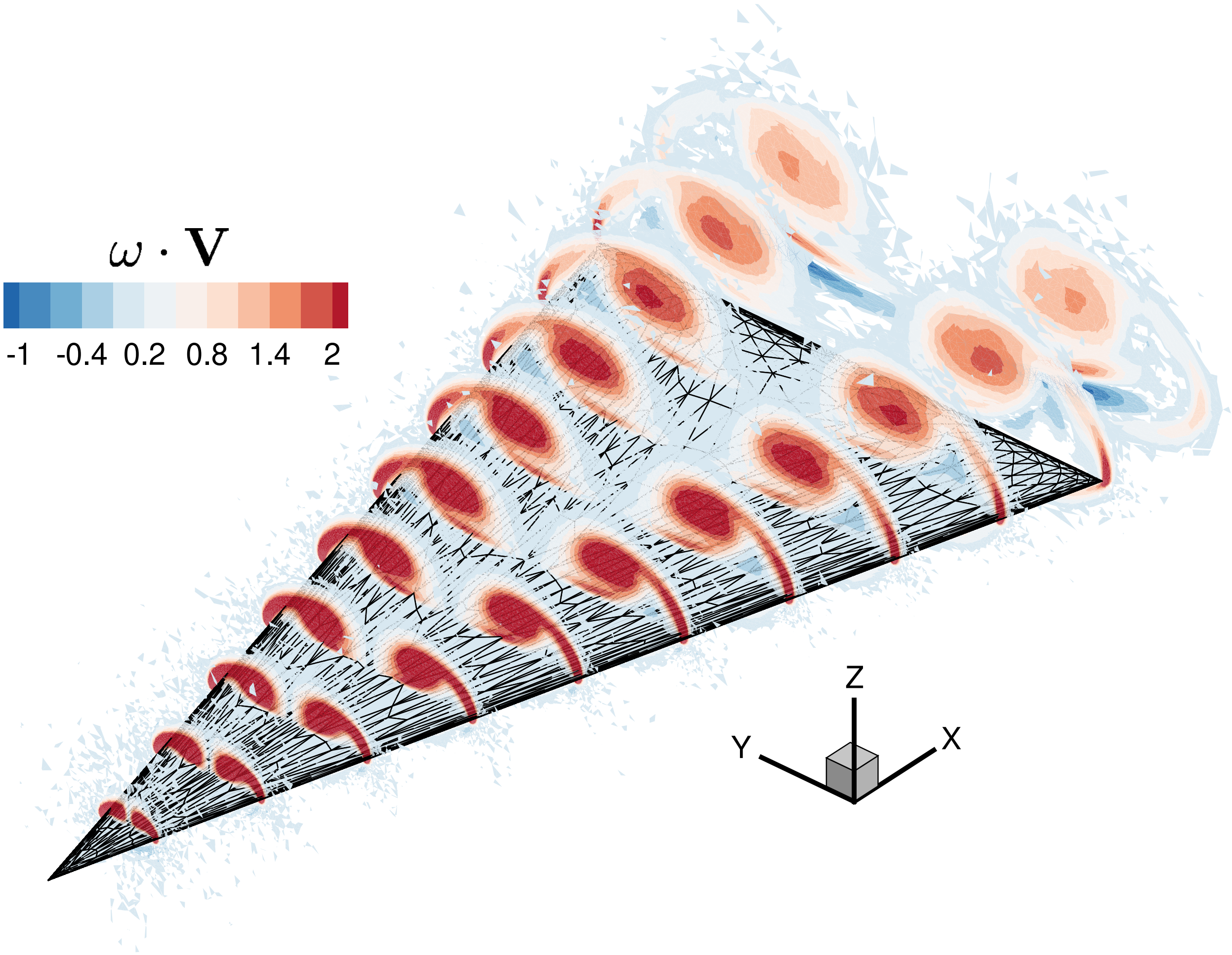


Figure 4.4: Surface pressure distribution and slices of $\omega \cdot \mathbf{V}$ for a VMSD drag output adaptation with $p = 2$ and 5.12×10^4 DOF

where $\alpha_{DG} = \frac{1}{1.5\sqrt{d}\log(2)}$, compared to $\alpha_{VMSD} = \frac{1}{1.1\sqrt{d}\log(2)}$.

To generate the convergence plots, each adaptation is started from the same mesh of 42 elements with a cost request of 1000, twenty adaptation iterations are then performed before the cost is incremented and the process repeated. This results in the final adapted mesh from the previous cost request being used as the initial mesh for the next.

Figure 4.5 is a plot of the error indicator, $\mathcal{H}$, versus a notional mesh size $h_D = (\text{DOF})^{-\frac{1}{3}}$. For DG, $\mathcal{H} = \sum_{\kappa(\mathcal{T})} \eta_\kappa$ and for VMSD, $\mathcal{H} = \sum_{v(\mathcal{T})} \eta_v$. The final 10 iterations at each cost level are plotted in a scatter, with a line plotted through the average. VMSD achieves the same error indicator as the DG discretization, with markedly fewer DOF, particularly for $p = 1$, which is in line with the savings predicted in Chapter 2.

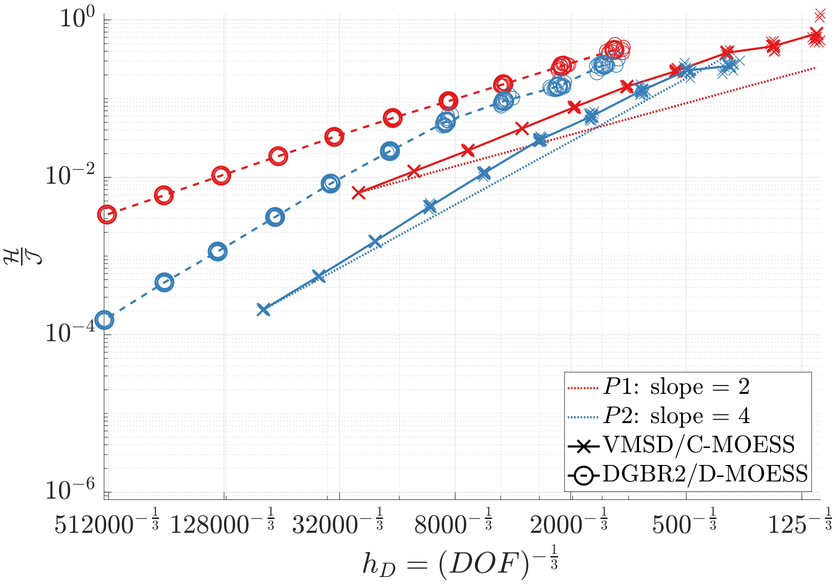


Figure 4.5: Convergence results for the error indicator versus mesh size $h_D = (\text{DOF})^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value.

Figure 4.6 shows the error estimate versus mesh size h_D. The data are significantly offset compared to Figure 4.5 indicating cancellation effects that are not accounted for in the MOESS optimization statement. The error estimate values are in good agreement with the 'true' error shown in Figure 4.7. The rates of convergence in Figures 4.5 to 4.7 are all equal to or greater than the expected $2p$ rate of convergence indicated by the fine dotted lines in the plots.

Figures 4.8 to 4.10 show the same data as Figures 4.5 to 4.7 plotted against $h_E = (N_e)^{-\frac{1}{3}}$ where N_e is the number of elements. For a given number of elements DGBR2 has less error than VMSD. This is expected because on a given mesh, a DG solution will have less error than a CG solution provided the stability constants for the two discretizations are

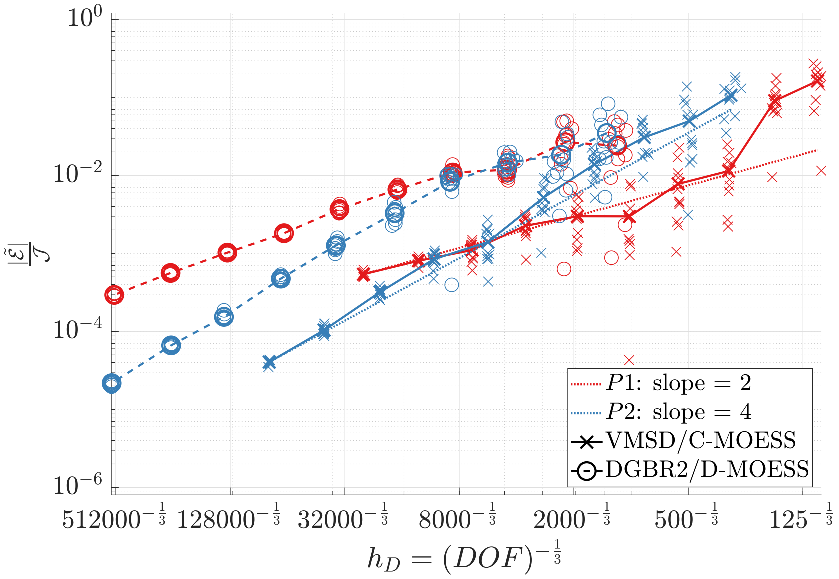


Figure 4.6: Convergence results for the error estimate versus mesh size $h_D = (\text{DOF})^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value.

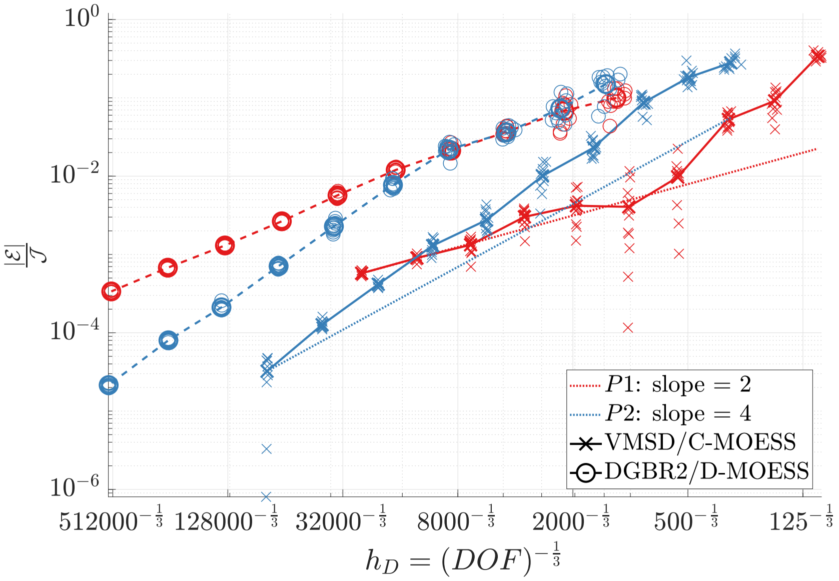


Figure 4.7: Convergence results for the error calculated relative to a 'truth' value of $\mathcal{J} = 1.656\,791\,305\,6 \times 10^{-1}$ versus mesh size $h_D = (\text{DOF})^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value. The dashed black line indicates 0.1% error in the output functional.

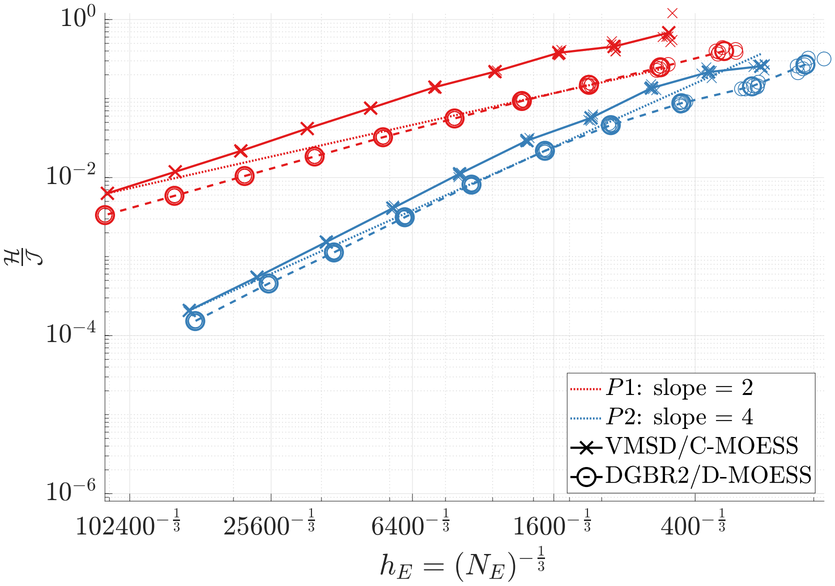


Figure 4.8: Convergence results for the error indicator versus $h_E = (N_e)^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value.

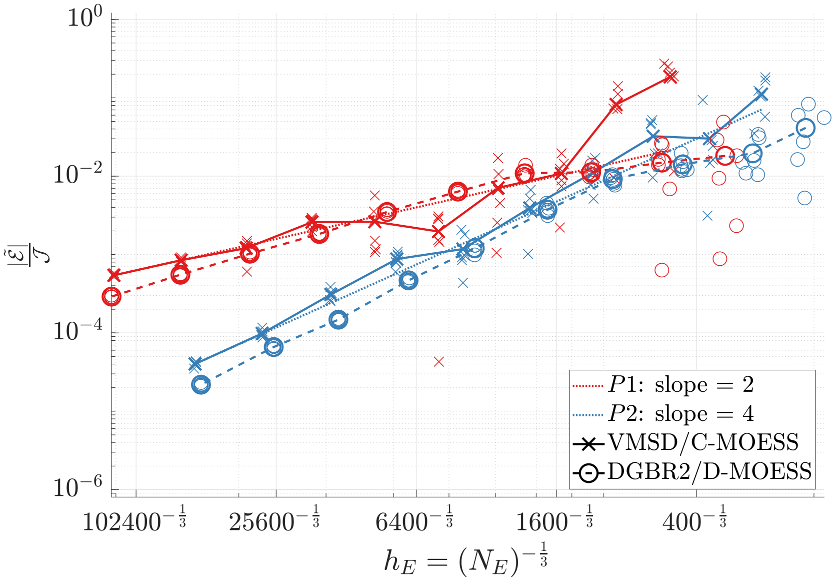


Figure 4.9: Convergence results for the error estimate versus $h_E = (N_e)^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value.

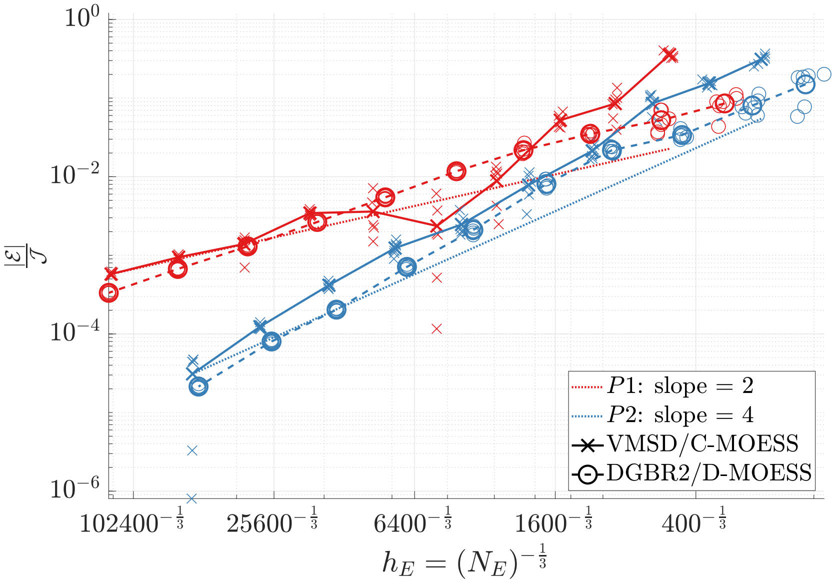


Figure 4.10: Convergence results for the error calculated relative to a 'truth' value of $\mathcal{J} = 1.656\,791\,305\,6 \times 10^{-1}$ versus $h_E = (N_e)^{-\frac{1}{3}}$ for the delta wing case. The scattered data represent the final 10 iterations at a given cost level, whilst the lines are plotted through the mean value. The dashed black line indicates 0.1% error in the output functional.

similar. This is due to the discontinuous space being an enrichment of the continuous one.

Run Time Comparison

To make timing comparisons between the two algorithms, we estimate the required number of DOF to achieve less than 0.1% error in the output functional. The error is computed relative to a 'truth' value taken from a 512×10^3 DOF DGBR2 adaptation with $p = 3$, which gives $\mathcal{J} = 1.656\,791\,305\,6 \times 10^{-1}$. The error estimate used is the Dual Weighted Residual[45], and for VMSD we use the vertex based partition of unity of Richter and Wick[49]. The linear solver is PETSc using GMRES with a restart of 300, an ILU preconditioner with fill-in of 4, a relative tolerance of 1×10^{-12}, and blocking at the PDE level. Parallel preconditioning is performed using Additive Schwarz with an overlap of one and 16 partitions. The nonlinear solver uses the PTC algorithm described in Appendix C.1, taking a single Newton step per PTC iteration and with a residual norm tolerance of 1×10^{-7} measured in the L^2 norm. The average number of primal DOF, elements and relative output error over the last 10 iterations are shown in Table 4.1; as expected the number of elements for VMSD is greater than the number of DOF and vice-versa for DGBR2.

The adaptation is run until the error indicator is approximately constant and then average times taken over ten iterations. The comparisons are done using 16 Intel Haswell cores with 2.6GHz clocks and 4MB cache each, the communication between processor cores is per-

[45] Becker *et al.*, *A feed-back approach to error control in finite element methods: Basic analysis and examples*. 1996

[49] Richter *et al.*, *Variational localizations of the dual weighted residual estimator*. 2015

	VMSD	DGBR2
$p = 1$	2.475×10^4	2.660×10^5
$p = 2$	1.661×10^4	7.774×10^4

(a) Average Number of Primal Navier Stokes DOF

	VMSD	DGBR2
$p = 1$	1.359×10^5	6.650×10^4
$p = 2$	1.128×10^4	7.774×10^3

(b) Average Number of Elements

	VMSD	DGBR2
$p = 1$	4.052×10^{-4}	4.007×10^{-4}
$p = 2$	3.45×10^{-4}	3.852×10^{-4}

(c) Average Relative 'True' Output Error $\frac{|\mathcal{J} - \mathcal{J}_h|}{\mathcal{J}}$

Table 4.1: Average number of DOF, Elements and 'true' output error for the delta wing timing comparison cases.

formed via MPI. The timings are broken down into the primal solve, the adjoint solve in $p + 1$ (and p for VMSD), the error estimation, the local solves, the meshing, the optimization and the implied metric calculation. The meshing is performed using EPIC[67], and the optimization is performed using the Method of Moving Asymptotes (MMA) algorithm[97], implemented within NLOPT[98]. The meshing is performed using two processor cores with shared memory parallelism and the optimization is done in serial[6]. The primal solve, adjoint solve, estimation and local solves utilize all 16 processor cores through distributed memory parallelism. CPU time for the parallel operations is obtained by multiplying the wall clock time by the number of processor cores, for instance 16 for the primal solve and 2 for the meshing.

The timing results are summarized in Figures 4.11 to 4.13 with the run times tabulated and the breakdowns displayed for the two algorithms and $p = 1, 2$. Figure 4.11e shows the wall time, Figure 4.12e shows the overall CPU time, and Figure 4.13e the wall time excluding the meshing and optimization operations which are not run with 16 processor cores. All of the linear solves and non-linear solves achieved their respective tolerances, and the non-linear solve required approximately $5 - 8$ steps of the PTC algorithm for each primal solve[7].

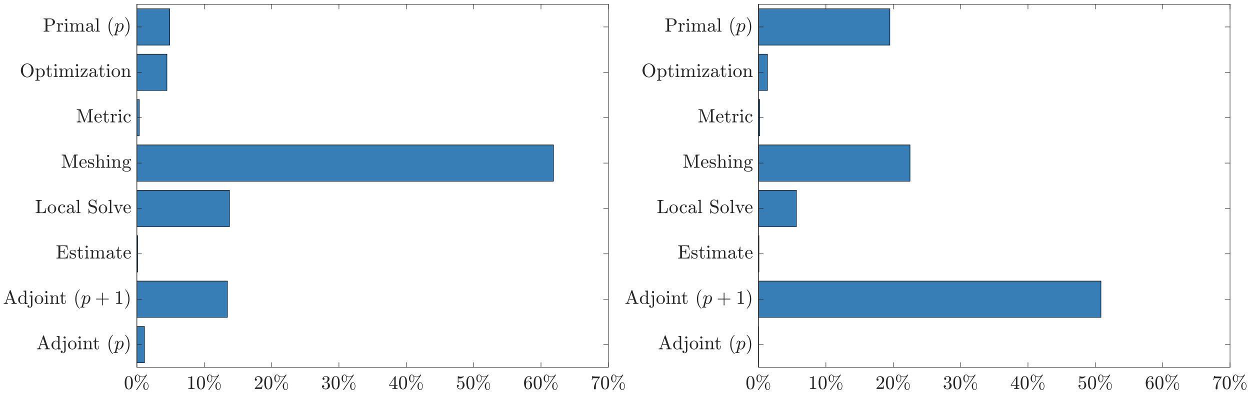
Figure 4.11e suggests that for both algorithms in this testing configuration, increasing the polynomial order results in a faster algorithm, achieving the same error in less time. The fastest of the four combinations is DGBR2 with $p = 2$ which is 5% faster than VMSD $p = 2$ in second place. Comparing the breakdowns for VMSD to those of DGBR2 it becomes apparent that the C-MOESS adaptations are spending a significantly greater proportion of their time on meshing and the local solves. The $p = 1$ VMSD average element count from Ta-

67. Michal *et al.*, *Anisotropic Mesh Adaptation through Edge Primitive Operations*. 2012

97. Svanberg, *A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations*. 2002

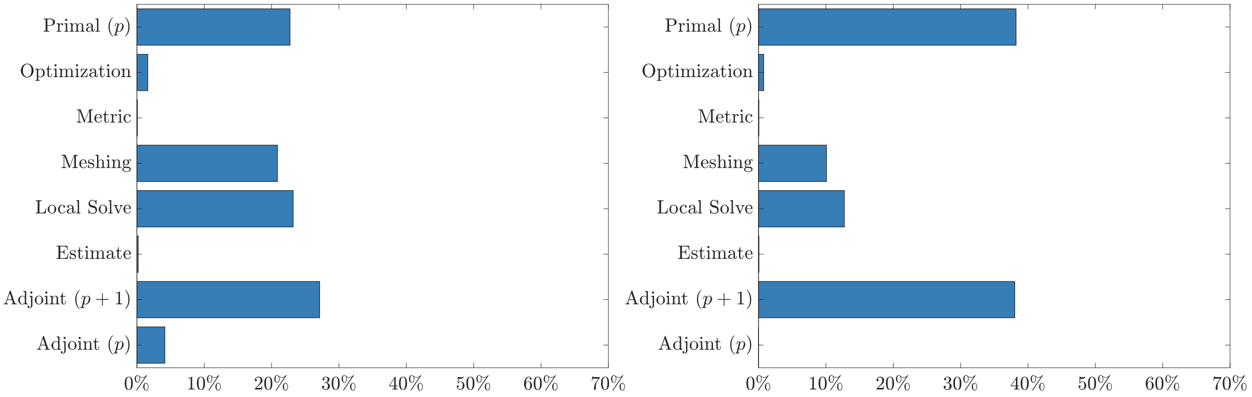
[6] The meshing was performed using only two processors due to the limitations of the EPIC mesher at the time. There is no fundamental reason it cannot be performed on more cores ultimately, though the more parallel meshing capability is at present less robust than the more serial. The optimization was performed in serial because NLOPT is not implemented in parallel; however, performing the optimization in parallel is in principle possible given there is only one globally coupled Lagrange multiplier λ.

[7] This is a very small number compared to the $>$ 50 typically required of a Reynolds Averaged Navier Stokes (RANS) simulation, as such this particular problem represents a relatively mild non-linear problem. For a more difficult case the proportion of the total time spent in the primal would increase significantly and the overall benefits of VMSD/C-MOESS are likely to become more apparent.



(a) VMSD $p = 1$, $\mathbb{C} = 25 \times 10^3$

(b) DGBR2 $p = 1$, $\mathbb{C} = 257 \times 10^3$



(c) VMSD $p = 2$, $\mathbb{C} = 12 \times 10^3$

(d) DGBR2 $p = 2$, $\mathbb{C} = 72 \times 10^3$

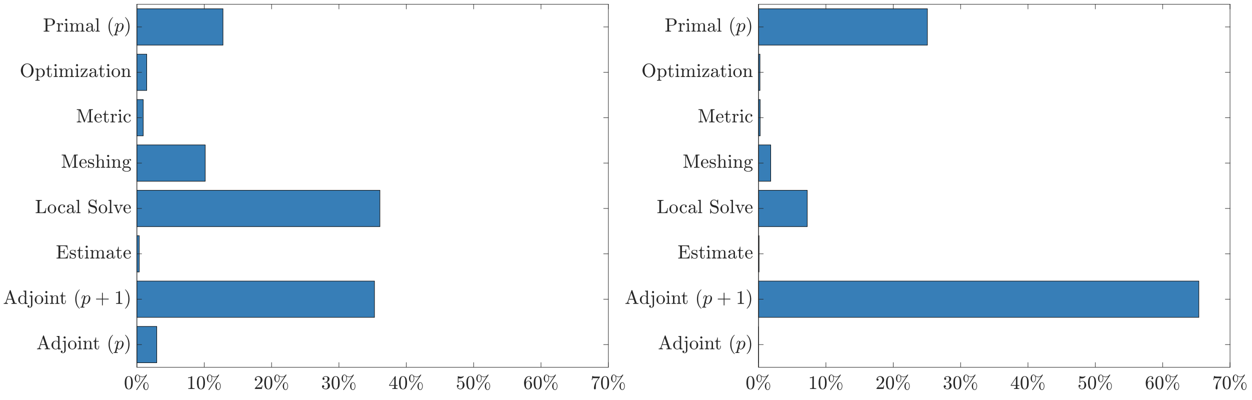
	VMSD	DGBR2
$p = 1$	1157.28	964.20
$p = 2$	339.85	323.04

(e) Wall Time (s)

	VMSD	DGBR2
$p = 1$	3.58	2.98
$p = 2$	1.05	1

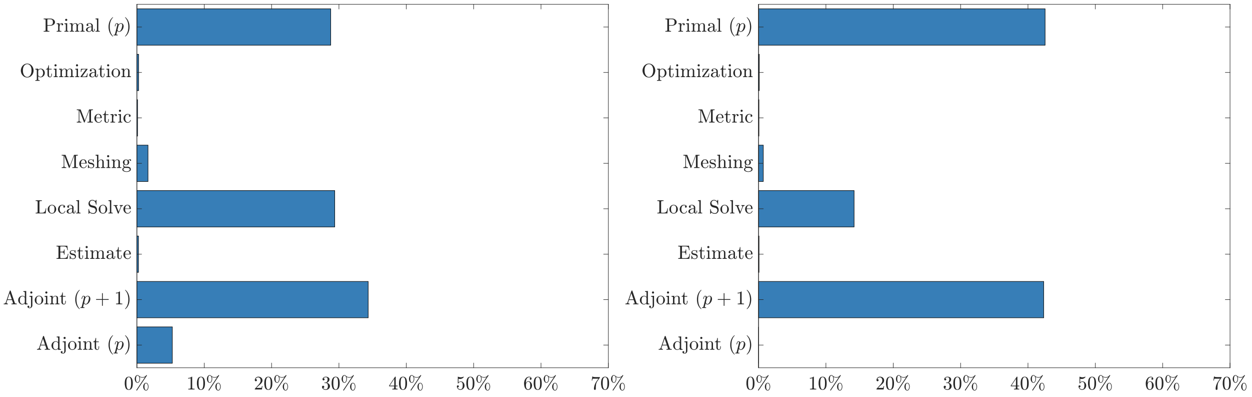
(f) Normalized Wall Time

Figure 4.11: Average Total Wall Time and breakdowns for an adaptation iteration. Meshing is performed with two threads and optimization is performed in serial. $\mathbb{C}$ is the requested cost from the optimization.



(a) VMSD $p = 1$, $\mathbb{C} = 25 \times 10^3$

(b) DGBR2 $p = 1$, $\mathbb{C} = 257 \times 10^3$



(c) VMSD $p = 2$, $\mathbb{C} = 12 \times 10^3$

(d) DGBR2 $p = 2$, $\mathbb{C} = 72 \times 10^3$

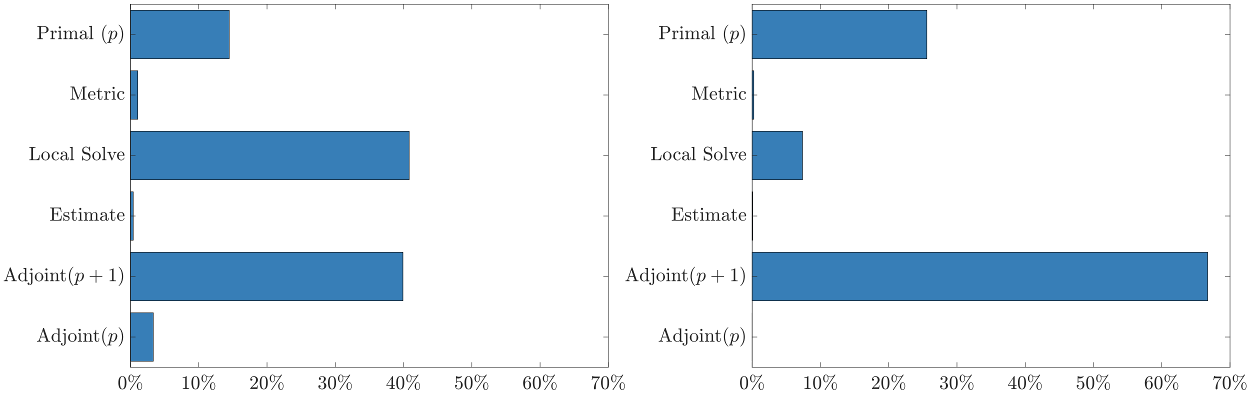
	VMSD	DGBR2
$p = 1$	7056.30	11 996.69
$p = 2$	4296.26	4644.96

(e) CPU Time (s)

	VMSD	DGBR2
$p = 1$	1.64	2.79
$p = 2$	1	1.08

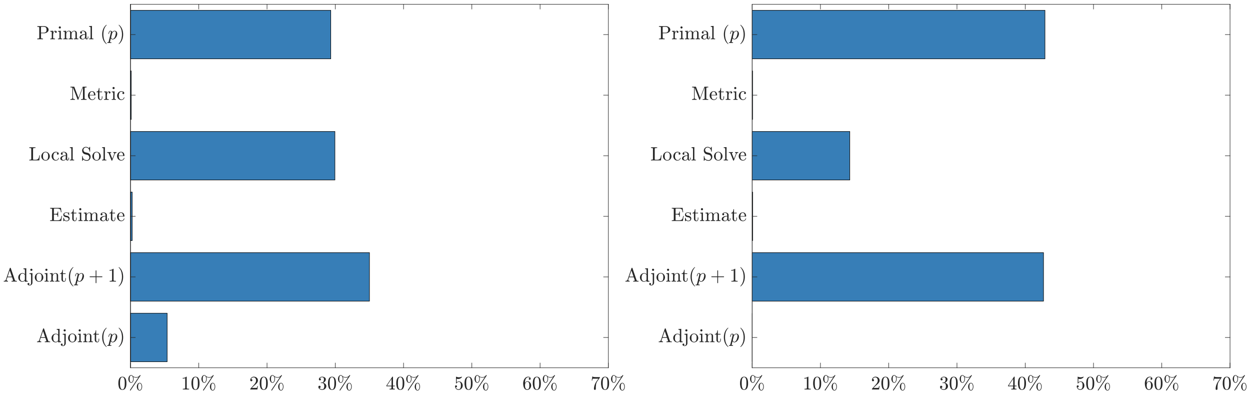
(f) Normalized CPU Time

Figure 4.12: Average Total CPU Time and breakdowns for an adaptation iteration. CPU time is taken to be number of processor cores multiplied by wall time for processes that are run in parallel. $\mathbb{C}$ is the requested cost from the optimization.



(a) VMSD $p = 1$, $\mathbb{C} = 25 \times 10^3$

(b) DGBR2 $p = 1$, $\mathbb{C} = 257 \times 10^3$



(c) VMSD $p = 2$, $\mathbb{C} = 12 \times 10^3$

(d) DGBR2 $p = 2$, $\mathbb{C} = 72 \times 10^3$

	VMSD	DGBR2
$p = 1$	389.83	734.65
$p = 2$	263.39	287.96

(e) Parallel Wall Time (s)

	VMSD	DGBR2
$p = 1$	1.48	2.79
$p = 2$	1	1.09

(f) Normalized Parallel Wall Time

Figure 4.13: Average Wall Time for Parallel Operations, and breakdowns for an adaptation iteration. $\mathbb{C}$ is the requested cost from the optimization.

ble 4.1b corresponds to 2.04 times as many elements as for DGBR2. DGBR2 spent 216.83s on average meshing, compared to 715.82s for VMSD, a factor of 3.3 increase. This suggests that the mesher is performing a larger quantity of work per element, in addition to operating on a larger number of elements. Large requested changes to the metric would correspond to more oscillatory behaviour in the output quantities, as the mesh would vary more significantly from iteration to iteration.

The average step norm for $\{S_v\}$ can be defined as

$$\|\{S_v\}\| = \sqrt{\frac{1}{|v(\mathcal{T})|} \sum_{v(\mathcal{T})} \|S_v\|_F^2}$$

with a smaller value of $\|\{S_v\}\|$ corresponding to a smaller requested change to the implied metric. This quantity has been computed for each of the cases, averaged over the last 10 iterations. The results of Table 4.2 show that for $p = 1$ VMSD is requesting changes which are, measured under the norm, 1.8 times as large as those from DGBR2[8]. This approximate doubling in the size of the requested step matrix field, combined with the doubling of the number of elements accounts for the factor of 3.3 increase in meshing time for VMSD compared to DGBR2. The results of Table 4.2 are supported by the comparatively large scatter of the data points comparing VMSD to DGBR2 for $p = 1$ in Figure 4.7.

	VMSD	DGBR2
$p = 1$	0.8906	0.4902
$p = 2$	0.9585	0.6615

Table 4.2: Average Step Norm, $\|\{S_v\}\| = \sqrt{\frac{1}{|v(\mathcal{T})|} \sum_{v(\mathcal{T})} \|S_v\|_F^2}$, averaged over 10 iterations.

The choice of $\alpha_{VMSD} = \frac{1}{1.1\sqrt{d}\log 2}$ was motivated by the notion of *quasi-unity*, as outlined in Chapter 3, and comes from assuming that EPIC is a $\sqrt{d}\log 2$ contractive mesher[9]. In general however, EPIC may perform edge splits and collapses inside of this range, meaning it is difficult to classify EPIC precisely within the β contractive mesher framework. This sensitivity of EPIC with respect to changes in the metric may in part explain the scatter observed in Figure 4.7 for lower DOF requests.

The total CPU time controls for the effect of the serial bottle-necks, and the data can be seen in Figure 4.12e where for either polynomial order VMSD combined with C-MOESS is now the fastest algorithm. The improvement of VMSD over DGBR2 is 70% and 8% for $p = 1$ and $p = 2$ respectively. The ratio of the p adjoint solve to the primal solve for VMSD highlights that the primal solution requires very few linear solves, indicating that the problem though non-linear is relatively benign. As with the wall-time comparisons of Figure 4.11e, the VMSD

[8] One possible reason for this discrepancy is that the regularization provides an additional averaging role for the elemental model, coming from the fact the working variables are vertex-defined step matrices. The unconstrained stationarity condition for S_v with the elemental model gives

$$S_v = -\frac{1}{\alpha_{DG}} \sum_{\kappa(v)} w_\kappa \frac{R_\kappa}{\|R_\kappa\|_F}$$

where $\sum_{\kappa(v)} w_\kappa = 1$, and $w_\kappa = \frac{\eta_\kappa e^{X_\kappa}}{\sum_{\kappa(v)} \eta_\kappa e^{X_\kappa}}$ and $X_\kappa \equiv \operatorname{tr}(R_\kappa S_\kappa) + \frac{\alpha_{DG}\|R_\kappa\|}{2(d+1)} \sum_{\kappa(v)} \|S_v\|_F^2$.

[9] The rationale for this choice is that edge lengths inside of the range $[\frac{1}{\sqrt{2}}, \sqrt{2}]$ are treated as *metric conforming* under the notion of *quasi-unity*.

cases are spending a significantly larger proportion of their time on the local solves compared to DGBR2. For all of the cases the $p+1$ adjoint is at least approximately $\frac{1}{3}$ of the overall cost of the iteration. This reflects the relative importance of the linear solver in the overall performance of the algorithm.

Figure 4.13e shows wall time results with the optimization and meshing timings removed. The VMSD results improve when compared to the CPU-time results, which is a reflection of the increased time spent meshing compared to the other cases. For PDEs where the non-linear system is more difficult to solve, e.g. the Reynolds Averaged Navier Stokes equations, the number of non-linear iterations to solve will be significantly larger. In which case the proportion of CPU time for the primal solve will likely increase relative to the local solves, the cost of which will grow more linearly than the full primal solves.

The reason for the additional relative cost of the local solves for VMSD compared to DGBR2 is that the Jacobian construction is approximately twice as expensive. This cost arises due to all the additional computations required in order to perform the static condensation. This additional computational cost in the Jacobian construction is the trade-off made in order to reduce the size of the global solve. The edge-based local solve patches for C-MOESS mean that the total number of element Jacobian evaluations on a mesh during the local solve is approximately 12 times[10] the number of edges of the mesh. The element-based local solve patch for D-MOESS mean that the total number of element Jacobian evaluations on a mesh is approximately 12 times[11] the number of elements.

This additional cost for the Jacobian construction is also seen in the global solves, where for VMSD the ratio of linear solve time to Jacobian construction time for the primal is $\frac{1}{2.5}$ for $p=1$ and $\frac{1}{1.63}$ for $p=2$. This contrasts with DGBR2 where the ratio of linear solve time to Jacobian construction time is 35.4 for $p=1$ and 16.14 for $p=2$. The trend is more pronounced when considering the $p+1$ adjoint solves: DGBR2 has ratios of 85.2 and 19.74 for $p=1,2$ respectively, compared to 5.22 and 2.61 for VMSD $p=1,2$ respectively. The relatively large cost of the linear solve here represents a limitation of the study. In particular the performance of DG might be improved using a *state-of-the-art* preconditioner such as block ILU with Minimum Discarded Fill (MDF) ordering or some such, as the DG algorithm is largely designed to work well with such preconditioners. For this study we neglected this approach because the design of block preconditioners for CG is significantly more complex. A more advanced study would involve the comparison of VMSD/DGBR2 with discretization specific preconditioners.

An additional important feature for comparison of the two algo-

[10] 6 elements attached to an edge, each split into two elements.

[11] 6 edges to a tetrahedra, each split resulting in two elements.

	VMSD	DGBR2
$p = 1$	7	$\frac{5}{2}$
$p = 2$	$\frac{23}{7}$	2

(a) Adjoint (p') to primal (p) DOF ratio

	Primal	Adjoint
$p = 1$	10.74	3.96
$p = 2$	4.67	2.84

(b) Observed DOF ratio: $\frac{DOF_{DGBR2}}{DOF_{VMSD}}$

Table 4.3: Ratio of DOFs for the adjoint in p' to primal in p computed using the formulas which come from Equation (2.39) of Chapter 2. The observed DOF ratios for the primal come from Table 4.1a, and the DOF ratios for the adjoint come from the application of Table 4.3a to the primal ratios.

rithms is memory footprint. The number of DOF required for a VMSD solution to arrive at a given error is less than that required for a DGBR2 solution, as seen in Table 4.1a. As the memory requirement is approximately proportional to the number of degrees of freedom, a DG solution will require more memory for a given accuracy. Using the data from Table 4.1a; for $p = 2$, compared to VMSD, DGBR2 requires 8% more CPU time and a factor of 4.67 more DOFs in the primal, whilst for $p = 1$, DGBR2 requires 70% more CPU time and a factor of 10.74 more DOFs in the primal. To get an approximate number of DOF in the adjoint, we can scale the values of Table 4.1a using the formulas in Chapter 2[12], which are summarized in Table 4.3a. Using the adjoint DOFs, DGBR2 requires a factor 3.84 more DOFs for $p = 1$ and a factor of 2.84 for $p = 2$. Table 4.3b summarizes the ratio of DOF requirements for DGBR2 compared to VMSD.

	VMSD	DGBR2	$\frac{M_{DGBR2}}{M_{VMSD}}$
$p = 1$	8.18×10^3	19.85×10^3	2.43
$p = 2$	6.35×10^3	12.25×10^3	1.93

(a) Maximum PETSc memory (MB): Primal

	VMSD	DGBR2	$\frac{M_{DGBR2}}{M_{VMSD}}$
$p = 1$	28.97×10^3	63.29×10^3	2.18
$p = 2$	14.67×10^3	27.9×10^3	1.90

(b) Maximum PETSc memory (MB): Adjoint

Table 4.4: Maximum reported PETSc memory requirement for primal and adjoint linear solves, measured in MB.

Tables 4.4a and 4.4b shows the maximum reported memory requirements from PETSc for the linear solves during the timing runs. This includes the memory usage in the preconditioner, Krylov vectors and the Jacobian. The memory savings are approximately a factor of two for all p for primal and adjoint. This reduction compared to the DOF ratios of Table 4.3b is likely a function of the choice of ILU(4) for the preconditioner: the VMSD DOFs are more tightly coupled, and thus we would expect the memory used in the preconditioner to grow faster for VMSD than for DG. Despite this, VMSD still exhibits memory savings when compared to DGBR2, and using lower level of fill-in will result in larger memory savings[115]. A particular limitation of the study in this section is this decision to utilize the same preconditioner for both discretizations. An alternative means of comparing the two would be to develop/utilize specialized preconditioners specific

[12] See Equation (2.39). These are exact for discontinuous discretization and an approximation for continuous discretization.

115. Huang, *An Adaptive Variational Mutltiscale Method with Discontinuous Subscales for Aerodynamics Flows*. 2019

to each discretization, for instance element block-ILU with MDF for DG. The development of such a specialized preconditioner for VMSD is however in a nascent stage, as such the decision was taken to utilize the same preconditioner for the two discretizations, so as to attempt to make a fairer comparison.

Table 4.5a contains an approximate number of DOFs per processor during the timing comparisons, obtained by multiplying the numbers in Table 4.1a by $\frac{5}{16}$[13]. A performance heuristic using PETSc for solving linear systems in parallel is aiming for greater than 2×10^4 (at a minimum 1×10^4) linear algebra DOFs per processor[14]. Thus the VMSD primal linear solves are slightly over parallelized, and as a consequence the CPU time of the primal linear solves for VMSD might be reduced by using a smaller number of processor cores. However, the local solves, residual evaluations and Jacobian evaluations scale very well with increasing number of processor cores, thus the local solve wall-time time would be reduced by using more processor cores. Reconciling these conflicting requirements suggests that a parallelization strategy using a greater number of processor cores in the local solves, residual evaluations and Jacobian evaluations than in the linear solves may improve CPU time performance.

	VMSD	DGBR2
$p = 1$	7734	83 125
$p = 2$	5190	24 247

(a) Primal (p) DOF per processor

	VMSD	DGBR2
$p = 1$	54 140	207 813
$p = 2$	17 053	48 494

(b) Adjoint ($p+1$) DOF per processor

Table 4.5: Approximate number of linear algebra DOFs per processor for VMSD and DGBR2 during the timing comparison. Adjoint DOF are computed using Table 4.3a.

4.4 *Conclusions*

In this chapter we demonstrated the C-MOESS algorithm of Chapter 2 for a laminar Navier Stokes simulation of a delta wing in three dimensions. The stabilization was performed using the Variational Multi-Scale with Discontinuous sub-scales (VMSD) discretization. The resulting algorithm is compared to the DGBR2 implementation of Yano, demonstrating that the VMSD algorithm can achieve the same error with significantly fewer number of degrees of freedom. The resulting output functionals are shown to be converging at higher-order, independent of the presence of geometric singularities, in line with the theoretical work of Chapter 3.

The second half of the chapter is concerned with a run-time comparison of the two versions of MOESS. It was found that for the configuration of linear solver/non-linear solver considered, the reduced number of DOFs of VMSD translates into reduced CPU time to solution compared to DGBR2, with the most pronounced effect being for

[13] 5 Navier Stokes variables and 16 processor cores.

[14] https://www.mcs.anl.gov/petsc/documentation/faq.html

$p = 1$. In the future a more sophisticated comparison study should be performed utilizing discretization specific preconditioning strategies, for instance block ILU with MDF for DG. The design of an efficient preconditioner for VMSD is an open-research question. The impact of serial bottlenecks on the performance of the algorithm was shown, this suggests:

1. High performance meshers are of great importance for reducing the overall run time of a MOESS algorithm.
2. The efficiency of VMSD can be improved by utilizing a more sophisticated parallelization strategy that employs different numbers of processor cores for the linear solves compared to the local solves.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 *Contributions and Conclusions*

The primary foci of the work contained in this thesis are two-fold: improving the *efficiency* and the *rigour* of the MOESS algorithm. The efficiency of the MOESS algorithm was addressed in the extension of MOESS to continuous finite element discretization, thereby overcoming the DOF inefficiency of the DG discretization. The resulting algorithm, developed in Chapter 2 and denoted C-MOESS, required the design of a new local solve process that was cognizant of the sharing of degrees of freedom amongst adjacent elements. This sharing of degrees of freedoms meant that the original DG-style elemental local solve process was ill-suited. C-MOESS is built around a new *edge*-based local patch combined with a *vertex*-based error model that is able to account for the multi-element support of the continuous basis functions. The algorithm was demonstrated for L^2 error control in two dimensions and a linear advection diffusion problem in three dimensions; the expected DOF efficiency of CG compared to DG was achieved.

The rigour of the MOESS algorithm was addressed in Chapter 3 with the construction of a proof that the MOESS algorithm will converge to zero error as the requested cost increases. This is to the authors' knowledge the first such proof for a Metric Adaptive Finite Element Method (MAFEM); the existing literature for convergence of adaptive finite element methods are predicated on the nested construction of the solution spaces whereas a MAFEM does not have this nested structure. The error on successive meshes was instead related by appeal to the optimization statement which introduces global coupling between local errors throughout the mesh. A notable feature of the construction of this proof was the introduction of an abstract definition of a metric mesher: the β-contractive metric mesher. This technical construction was shown to be necessary given the lack of structure directly relating two consecutive meshes. Additionally this optimization based framework enabled a proof of higher order rate of convergence for a MAFEM provided the number of elements required to capture any singularities in the solution does not grow too fast. The expected rates of convergence were demonstrated for Poisson's equation on a re-entrant corner, achieving the higher-order rate of convergence predicted by the analysis, despite the presence of singularities in the solution and adjoint.

Chapter 4 applied the methods developed in Chapters 2 and 3 to a three dimensional Navier Stokes simulation of a delta wing geometry. In order to stabilize the CG discretization a novel stabilization method, Variational Multi-Scale with Discontinuous sub-scales (VMSD), is utilized. The VMSD discretization employs a discontinuous solution space to approximate the scales unresolved by the global C^0 solution space. VMSD combined with the new MOESS extension to continuous discretizations is compared to a MOESS implementation using DGBR2 and elemental models. The results demonstrate that the DOF efficiency shown in Chapter 2 is also achieved for more complex physics than the linear problems of Chapters 2 and 3. Finally a timing comparison was made between C-MOESS with VMSD and the original MOESS algorithm with DGBR2. VMSD combined with C-MOESS outperformed the incumbent DGBR2/D-MOESS algorithm for both $p = 1$ and $p = 2$; VMSD was able to achieve the same error levels with reduced CPU time and reduced memory. The importance of meshing was also highlighted: the wall time of the meshing operations is a significant bottleneck in the algorithm.

5.2 Future Work

Based upon the conclusions of this work, we make some suggestions for future avenues of research.

Higher Order Error-Metric Models and Higher-Order Metric Meshers

The original MOESS algorithm employed elemental error models, constructed around elemental error estimates and an elemental metric model. However, metric meshers are generally currently unable to operate on elementally defined metrics, instead using vertex-defined metrics. These elemental models would then have to operate on the average of the attached vertex-defined step matrices as opposed to an elemental step matrix. The nodal partition of unity approach utilized in Chapter 2 produced a local error model that is defined at the vertices of the mesh, thereby coupling the models closer to the metric representation of the mesher. However, the partition of unity approach need not be limited to linear basis functions. The Bernstein polynomials provide a higher-order partition of unity and could be use to arrive at a higher-order error-metric model.

It is likely however that in order for a higher-order error-metric model to be effective, native higher order metric mesh generation would be required. A higher-order metric mesher would utilize mesh curvature to arrive at improved metric and geometric conformity. Mesh curvature is necessary for higher-order discretizations of curved geometry[119], however presently such meshes are generated by first generating a linear metric conforming mesh and then curving it during post-processing. The possible benefits of curving the mesh during the re-meshing process in terms of improved metric conformity were outlined by Caplan *et al.*[120],[121]. The development of a higher order error-metric model in conjunction with a higher-order metric-mesher that natively curves meshes and takes higher-order metrics as input is an interesting proposition.

Parallel Mesh Generation

The results of Chapter 4 demonstrated that for larger highly parallelized problems the effect of the serial meshing bottleneck on wall time is considerable. Metric-meshers exists that are capable of fully parallel operation for instance `refine`[1], and Loseille has extended his cavity-based operator to parallel[122]; however, there remains additional research opportunities in achieving parallel metric mesh generation.

Safeguarded Output Error Estimation

The proof of convergence Chapter 3 is predicated upon the error indicator being a reliable error estimate for the true error. In order that the proof of asymptotic rate of convergence to be of practical importance, the error indicator is required to be efficient. However, one of the most useful error estimates for output error estimation, the Dual Weighted Residual (DWR) estimate, cannot be said to be reliable; the dual weighted residual error estimate can be zero even when the true error is non-zero, particularly for non-linear convection dominated problems. The work of Nochetto *et al.* develops a *safe-guarded* DWR estimate, however their approach has not been extended to more general PDEs and settings such as those of Chapter 4. Regardless, work towards improving the robustness and reliability of localizable output error estimates is warranted.

[119] Bassi *et al.*, *High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations*. 1997

[120] Caplan *et al.*, *Anisotropic Geometry-Conforming d-simplicial Meshing via Isometric Embeddings*. 2017

[121] Caplan *et al.*, *Isometric Embedding of Curvilinear Meshes Defined on Riemannian Metric Spaces*. 2018

[1] https://github.com/nasa/refine

[122] Loseille *et al.*, *Unique Cavity-Based Operator and Hierarchical Domain Partitioning for Fast Parallel Generation of Anisotropic Meshes*. 2017

Parallelism

The local solve process is embarrassingly parallel, as such the computational work can be arbitrarily divided up amongst processors. Thus the wall-time required for the local-solve process can always be reduced by adding more processors[2]. The conditioning for the *preconditioned* global system becomes poorer with each additional processor; adding additional processors beyond a certain point will *increase* the time spent solving a linear system. This effect is generally accounted for by preventing the number of DOF per processor dropping below a certain threshold.

In C-MOESS, to reduce the CPU time of the local solves requires introducing increasing numbers of processors, whilst to reduce the CPU time of the primal solve requires using fewer processors. This conflicting requirement suggests that improved performance might be achieved with a more sophisticated parallelism strategy whereby a larger number of processors are used for the local solves than for the global solves. Additionally the reduced memory requirements in the global system may make it possible to use a sparse-direct linear solver as opposed to an iterative solver for some practical sized problems. This parallelism might be achieved via GPU accelerators which could be used to perform the *embarrassingly* parallel local solves without resulting in deleterious performance of the linear solves.

Extension to four dimensions

The DOF efficiency of continuous Galerkin discretizations increases with dimension: the approximate number of elements sharing a vertex in $2D$ is 6, in $3D$ is 20 and in $4D$ is 120. As a consequence the benefits of the C-MOESS algorithm ought to be even more pronounced for a $3+1D$ space-time problem. Hyperbolicity in the time direction means that space-time simulations are particularly well suited to the MOESS algorithm[7]. The recent arrival of 4 dimensional metric-based mesh generation, in the form of `avro`, means that four dimensional MOESS based adaptation has been demonstrated for DG with L^2 projection and linear scalar advection diffusion[70]. However, the discontinuous Galerkin discretization of such problems requires prodigious quantities of memory and will be compounded by moving to systems of equations. Continuous finite element discretizations coupled with C-MOESS should offer significant cost reductions compared to DG and D-MOESS.

[2] Up to the total number of local solves required.

7. Yano, *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*. 2012

70. Caplan, *Four-dimensional Anisotropic Mesh Adaptation for Spacetime Numerical Simulations*. 2019

APPENDIX A

APPENDIX TO CHAPTER 2

To prove the bounds on the rate matrices, we first prove a lemma.

Lemma 6 (Geometric Invariant of the regular d-simplex). *If* $\mathbf{e}_i$ *make up the edges of a regular* d*-simplex then*

$$\sum_i \mathbf{e}_i \mathbf{e}_i^T = \frac{d+1}{2}\mathcal{I} \tag{A.1}$$

where $\mathcal{I}$ *is the identity matrix.*

Proof. Two edges $\mathbf{e}_i$, $\mathbf{e}_j$ do not share a triangle if $\mathbf{e}_i + \mathbf{e}_j + \mathbf{e}_k \neq 0$, $\forall k$ and $\mathbf{e}_i + \mathbf{e}_j - \mathbf{e}_k \neq 0$, $\forall k$, in which case $\mathbf{e}_i^T \mathbf{e}_j = 0$. If two edges do share a triangle in the simplex, then $\mathbf{e}_i^T \mathbf{e}_j = \pm\frac{1}{2}$ which follows from

$$(\mathbf{e}_i + \mathbf{e}_j + \mathbf{e}_k)^T(\mathbf{e}_i + \mathbf{e}_j + \mathbf{e}_k) = 0 \tag{A.2}$$

The number of edges that share a triangle with $\mathbf{e}_i$ (other than itself) is equal to $2(d-1)$. Defining $X = \sum_i \mathbf{e}_i \mathbf{e}_i^T$, the inner product can then be rearranged as

$$\mathbf{e}_i^T (\sum_i \mathbf{e}_i \mathbf{e}_i^T) \mathbf{e}_i = 1 + \sum_{i \neq j} \mathbf{e}_i^T \mathbf{e}_j \tag{A.3}$$

$$= 1 + \frac{2(d-1)}{4} = \frac{d+1}{2}, \ \forall i \tag{A.4}$$

then given $\{\mathbf{e}_i\}$ form a basis for $\mathbb{R}^d$, it follows that $X = \frac{d+1}{2}\mathcal{I}$. □

A.1 Elemental Rate Matrix Bounds

We now apply Lemma 6 to show that bounds on R_κ are a consequence of bounds on the data f_{κ_j}.

$\sum_j f_j < 0 \implies tr\,(R_\kappa) < 0$

Take $\mathcal{J}$ to be the set of edge splits, then write the S_{κ_j} in terms of the outer products

$$S_{\kappa_j} = \lambda_1 \mathbf{e}_j \mathbf{e}_j^T + \lambda_2 \mathbf{u}_j \mathbf{u}_j^T, \tag{A.5}$$

where $\lambda_1 + \lambda_2 = 2\log(2)$ are the only non-zero eigen values. The vectors $\mathbf{e}_j$ are the edges of a regular simplex, and the set $\{\mathbf{u}_j\}$ are normal to the respective $\mathbf{e}_j$ and point from the centroid of $\mathbf{e}_j$ to the centroid of κ. Thus the set $\{\mathbf{u}_j\}$ also make up the edges of another regular simplex. We can then write the summation

$$\begin{aligned} \sum_{j\in\mathcal{J}} S_{\kappa_j} &= \lambda_1 \sum_j \mathbf{e}_j \mathbf{e}_j^T + \lambda_2 \sum_j \mathbf{u}_j \mathbf{u}_j^T && \text{(A.6)} \\ &= (\lambda_1 + \lambda_2) \frac{d+1}{2} \mathcal{I}, && \text{(A.7)} \end{aligned}$$

then this can be substituted into the expression for R_κ

$$\begin{aligned} \operatorname{tr}\left(R_\kappa \sum_{j\in\mathcal{J}} S_{\kappa_j} \right) &= \sum_{j\in\mathcal{J}} f_{\kappa_j} && \text{(A.8)} \\ \operatorname{tr}\left(R_\kappa 2\log(2) \frac{d+1}{2} \mathcal{I} \right) &= \sum_{j\in\mathcal{J}} f_{\kappa_j} < 0 && \text{(A.9)} \\ \operatorname{tr}\,(R_\kappa) &= \frac{\sum_{j\in\mathcal{J}} f_{\kappa_j}}{d(d+1)\log(2)} < 0. && \text{(A.10)} \end{aligned}$$

$\mathcal{J}$ can be expanded to also contain an isotropic refinement because for this case $\hat{S}^{iso} = 2\log(2)\mathcal{I}$. Thus $\sum_j f_j < 0 \implies \operatorname{tr}\,(R_\kappa) < 0$.

$|\sum_j f_j| < \infty \implies \|R_\kappa\|_F < 0$

The rate matrix R_κ comes from solving the system

$$R_\kappa = \underset{X\in \mathrm{Sym}_d}{\arg\min} \sum_{j\in\mathcal{J}} \left(f_{\kappa_j} - \operatorname{tr}\left(X S_{\kappa_j} \right) \right)^2, \tag{A.11}$$

which is a quadratic minimization. Equation (A.11) has a unique solution and is an interpolation when $|\mathcal{J}| = \begin{pmatrix} d+1 \\ 2 \end{pmatrix}$. It is uniquely solved by requiring

$$\operatorname{tr}\left(R_\kappa S_{\kappa_j} \right) = f_{\kappa_j},\ \forall j \in \mathcal{J}. \tag{A.12}$$

Equation (A.12) can be expressed

$$\mathbf{S}_{\kappa_j}^T \mathbf{R}_\kappa = \mathbf{f}_{\kappa_j},\ \forall j \in \mathcal{J}, \tag{A.13}$$

where $\mathbf{S}_{\kappa_j}^T,\ \mathbf{R}_\kappa,\ \mathbf{f}_{\kappa_j} \in \mathbb{R}^{|\mathcal{J}|}$ are vectorized versions of the symmetric matrices. This defines a system $A\mathbf{R}_\kappa = \mathbf{f}_\kappa$, and given the solution exists and is unique, A must be invertible and thus $\|R_\kappa\|_F = \|\mathbf{R}_\kappa\|_2 \leq \|A^{-1}\|_2\|\mathbf{f}_\kappa\|_2$. That $\|A^{-1}\|_2$ is independent of κ follows from the fact that $\exists Q_\kappa : \ S_{\kappa_j} \equiv Q_\kappa \hat{S}_j Q_\kappa^T$. Then from the equivalence of norms it follows that $\|\mathbf{R}_\kappa\|_1 < \infty$ implies $\exists C_R : \ \|R_\kappa\|_F \leq C_R$. Given $f_{\kappa_j} \leq 0, \ \forall j$, then $\|f_{\kappa_j}\| = -\sum_j f_{\kappa_j}$.

A.2 Vertex Rate Matrix Bounds

In fitting a vertex model, $|\mathcal{E}|$ is not known *a priori* and the step matrices for every configuration are unknown. Thus

$$R_v = \underset{X \in \mathrm{Sym}_d}{\arg\min} \sum_{e \in \mathcal{E}(v)} \left(f_{v_e} - \mathrm{tr}\left(X S_{v_e}\right)\right)^2 \tag{A.14}$$

is not an interpolation but does still have a unique minimizer which comes from setting the derivative with respect to X equal to zero,

$$0 = \sum_{e \in \mathcal{E}(v)} \left(f_{v_e} - \mathrm{tr}\left(X S_{v_e}\right)\right). \tag{A.15}$$

We now apply Lemma 6 to show that bounds on R_v are a consequence of bounds on the data f_{v_e}.

$|\sum_e f_e| < \infty \implies \|R_v\|_F < 0$

In similar notation to equation (A.13) we vectorize R_v and define it as the solution to

$$(A^T A)\mathbf{R}_v = A^T \mathbf{f}_v. \tag{A.16}$$

$(A^T A)$ is square and invertible, thus provided $\|\mathbf{f}_v\|_2$ is bounded then $\exists C_R : \|R_v\|_F \leq C_R$.

$\sum_e f_e < 0 \implies tr\left(R_v\right) < 0$

In order to prove this result, we re-write the edge step matrixes, S_{v_e}, in equation (A.15) in terms of elemental step matrices, S_{κ_j}. S_{κ_e} denotes that the step comes from splitting a given edge $e \in \mathcal{E}(\mathcal{T})$ where as S_{κ_j} comes from $j \in \mathcal{J}$, where $\mathcal{J}$ are the set of edges of the reference simplex. In short, S_{κ_e} uses a global indexing system, whilst S_{κ_j} is an element local indexing system.

Noting that $S_{v_e} = \frac{1}{|\kappa(v)|} \sum_{\kappa \in \kappa(v)} S_{\kappa_e}$, it follows that

$$\begin{aligned}
\sum_{e \in \mathcal{E}(v)} S_{v_e} &\equiv \sum_{e \in \mathcal{E}(v)} \left(\frac{1}{|\kappa(v)|} \sum_{\kappa \in \kappa(e)} S_{\kappa_e} \right) && \text{(A.17)} \\
&= \frac{1}{|\kappa(v)|} \sum_{\kappa \in \kappa(v)} \sum_{j \in \mathcal{J}} Q_\kappa^T \hat{S}_j Q_\kappa && \text{(A.18)} \\
&= \frac{1}{|\kappa(v)|} \sum_{\kappa \in \kappa(v)} Q_\kappa^T (\sum_{j \in \mathcal{J}} \hat{S}_j) Q_\kappa && \text{(A.19)} \\
&= \frac{1}{|\kappa(v)|} \sum_{\kappa \in \kappa(v)} Q_\kappa^T \frac{(d+1)\log(2)}{2} \mathcal{I} Q_\kappa && \text{(A.20)} \\
&= \frac{(d+1)\log(2)}{2} \mathcal{I}, && \text{(A.21)}
\end{aligned}$$

where $\mathcal{J}$ denotes the edges of the reference element, $\hat{\kappa}$. The key to this manipulation is that within each element attached to the vertex, each edge is visited once.

Making this substitution into equation (A.15) then furnishes the result

$$\begin{aligned}
\operatorname{tr}\left(R_v \sum_{e \in \mathcal{E}(v)} S_{v_e} \right) &= \sum_{e \in \mathcal{E}(v)} f_{v_e} && \text{(A.22)} \\
\operatorname{tr}\left(R_v \log(2)(d+1)\mathcal{I} \right) &= \sum_{e \in \mathcal{E}(v)} f_{v_e} && \text{(A.23)} \\
r_v = \operatorname{tr}\left(R_v \right) &= \frac{\sum_{e \in \mathcal{E}(v)} f_{v_e}}{d(d+1)\log(2)}. && \text{(A.24)}
\end{aligned}$$

APPENDIX B

APPENDIX TO CHAPTER 3

B.1 Mesh properties

Definitions 8 to 11 are common with the definitions from Ciarlet and Lions [1] and Brenner and Scott[8], where the requirement for *a priori* error estimates is that $\{\mathcal{T}_n\}$ be a non-degenerate family of meshes.

Definition 8 (Non-Degenerate/Regular Mesh). *A mesh, $\mathcal{T}$, with corresponding elemental metrics $\{\mathcal{M}_\kappa\}$ is* non-degenerate *or* regular *if*

$$\exists \rho_{\mathcal{T}} > 0 : \; \rho_{\mathcal{T}} \leq \frac{B_\kappa}{h_\kappa} \sim \frac{\lambda_{\min,\kappa}^{\frac{1}{2}}}{\lambda_{\max,\kappa}^{\frac{1}{2}}}, \quad \forall \kappa \in \mathcal{T} \tag{B.1}$$

where λ_κ are the eigenvalues of metric $\mathcal{M}_\kappa$, $\lambda_{\min,\kappa} = \frac{1}{h_\kappa^2}$ and $\lambda_{\max,\kappa} \sim \frac{1}{B_\kappa^2}$ and $A \sim B \implies \exists c > 0 : \frac{1}{c}B \leq A \leq cB$. This is a minimal angle requirement for all κ.

Definition 9 (Non-Degenerate/Regular Set). *A set of meshes $\{\mathcal{T}_n\}$, $n \in \mathbb{N}$ with corresponding sets of elemental metrics, $\{\{\mathcal{M}_\kappa\}_n\}$ is* non-degenerate *or* regular *if*

$$\exists \rho > 0 : \; \rho \leq \frac{B_\kappa}{h_\kappa} \sim \frac{\lambda_{\min,\kappa}^{\frac{1}{2}}}{\lambda_{\max,\kappa}^{\frac{1}{2}}}, \quad \forall \kappa \in \mathcal{T}_n, \, \forall \mathcal{T}_n \in \{\mathcal{T}_n\} \tag{B.2}$$

namely

$$\rho = \inf_{\mathcal{T} \in \{\mathcal{T}_n\}} \rho_{\mathcal{T}} > 0 \tag{B.3}$$

where $\rho_{\mathcal{T}}$ is defined as in definition 8.

Definition 10 (Mesh family). *Let Ω be a given domain, $\{\mathcal{T}_n\}$ be a set of meshes, and define h as the largest element diameter*

$$\max_{\kappa \in \mathcal{T}_h}(diam(\kappa)) \leq h diam(\Omega). \tag{B.4}$$

A set is said to be a family of meshes if, fix $h > 0$

$$\exists N : \max_{\kappa \in \mathcal{T}_h}(diam(\kappa)) \leq h diam(\Omega), \ \forall n > N \tag{B.5}$$

Definition 11 (Quasi-uniformity and local quasi-uniformity)**.** *A family of meshes* $\{\mathcal{T}_n\}$ *is is said to be quasi-uniform if*

$$\exists \rho > 0 : \min_{\kappa \in \mathcal{T}_h}(diam(B_\kappa)) \geq \rho h diam(\Omega), \quad \forall h \in (0, 1] \tag{B.6}$$

A family of meshes is said to be locally quasi-uniform if it is also a non-degenerate set (see definition 9)

A set of meshes is any collection of meshes, a family is one where the largest element diameter h also decreases. In short, for a set of meshes, $n \to \infty \;\not\Longrightarrow\; h \to 0$, but for a family of meshes, $n \to \infty \implies h \to 0$.

APPENDIX C

PSEUDO TIME CONTROLLER

C.1 Pseudo Time Controller

In this section we outline the particular CFL controller used in the Pseudo Time Continuation (PTC) algorithm used in the non-linear solver of Chapter 4. This algorithm is based on private correspondence with Dr. Steven Allmaras and there does not exist a formal citation. This appendix does not constitute claiming authorship of the algorithm, and instead purely as documentation. The description at the start of this appendix is also very closely aligned with that of Ceze and Fidkowski[123].

The PTC algorithm seeks to find solutions to $\mathcal{R}(w_{hp}, u_{hp}) = 0$ by instead solving $(w_{hp}, \partial_t u_{hp})_{\mathcal{T}} + \mathcal{R}(w_{hp}, u_{hp}) = 0$ using a stable time discretization method until any transients have subsided. The semi-discretized system of equations is written

$$\mathbf{M}\frac{d\mathbf{U}}{dt} = -\mathbf{R}(\mathbf{U}), \tag{C.1}$$

where **M** is a mass-matrix, **U** is a state vector and **R** the discrete residual operator. The goal is not time-accuracy of the solution, but instead stability, thus the backwards Euler method is typically used, resulting in the Newton system

$$\left(\frac{\mathbf{M}}{\Delta t} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}|_{\mathbf{U}_k}\right)\Delta\mathbf{U}_k = -\mathbf{R}(\mathbf{U}_k). \tag{C.2}$$

For a very large time step Δt, this approximates a pure Newton step. For a very small value of Δt, the system is dominated by a mass matrix inversion. Correspondingly, if a small value of Δt is chosen, the system is easier to solve, but less progress is made towards the true steady state solution, conversely if Δt is larger, the system will be more non-

[123] Ceze *et al.*, *Constrained pseudo-transient continuation.* 2015

linear and thus harder to solve, but more progress made towards the true steady state solution.

Instead of a global time step as in equation (C.2), element-wise local time steps can be defined in terms of a CFL number.

$$\Delta t_\kappa = CFL \frac{L_\kappa}{\lambda_\kappa^{\max}} \tag{C.3}$$

where $\lambda_\kappa^{\max}$ is the maximum wave-speed in element κ, and L_κ is a measure of the element's size. Given this set of element-local time steps, a $\Delta \mathbf{U}_k$ can be calculated and an under-relaxed update applied,

$$\mathbf{U}_{k+1} = \mathbf{U}_k + \omega_k \Delta \mathbf{U}_k, \tag{C.4}$$

where $\omega_k \leq 1$ comes from a line search that checks for validity of the resulting state $\mathbf{U}_{k+1}$.

CFL controller

At each iteration the CFL is updated by considering the change in the residual given the size of the step ω_k. Define

$$\alpha(\omega_k) = \frac{\mathbf{R}(\mathbf{U}_k + \omega_k \Delta \mathbf{U}_k)}{\mathbf{R}(\mathbf{U}_k)}, \tag{C.5}$$

then α^* is defined by

$$\alpha^* = \begin{cases} \alpha(\omega_k), & \mathbf{U}_k + \omega_k \Delta \mathbf{U}_k \text{ valid } \forall \omega_k \in [0, 1] \\ \frac{1}{\omega_k} \alpha(\omega_k), & \text{otherwise} \end{cases}. \tag{C.6}$$

Note that this does not mean that ω_k is necessarily equal to 1. Given a value of α^* the *inverse* CFL is the updated

$$\frac{1}{\text{CFL}} = \left(\frac{\alpha^*}{\alpha_T} \right)^\beta \frac{1}{\text{CFL}} \tag{C.7}$$

where $\alpha_T = \frac{7}{10}$ and $\beta = \frac{1}{2}$ are hyper parameters. Additional limiters are also applied to the inverse CFL such that $\frac{1}{CFL} \in [0, 1 \times 10^6]$.

Appendix D

Axioms of Adaptivity

In this appendix we very briefly outline the axioms of adaptivity, as defined by Carstensen *et al.*[84]. If these axioms are satisfied, then an Adaptive Mesh Refinement method will converge in the energy norm. The approach can also be extended to output error estimation through application to the adjoint energy norm, as done by Feischl *et al*[85].

[84] Carstensen *et al.*, *Axioms of adaptivity*. 2014

[85] Feischl *et al.*, *An abstract analysis of optimal goal-oriented adaptivity*. 2016

D.1 Axioms of Convergence

The axiomatic proofs make use of a mesh distance function, $\mathbb{d}(\mathcal{T},\mathcal{T}')$ between two triangulations $\mathcal{T},\mathcal{T}' \in \mathbb{T}$ where $\mathbb{T}$ is the set of admissable triangulations. $\mathbb{d}(\mathcal{T},\mathcal{T}')$ is required to satisfy

$$\mathbb{d}(\mathcal{T},\mathcal{T}') \geq 0,\ \mathcal{T} \neq \mathcal{T}' \tag{D.1}$$

$$\frac{1}{C_{\text{dist}}}\mathbb{d}(\mathcal{T},\mathcal{T}'') \leq \mathbb{d}(\mathcal{T},\mathcal{T}') + \mathbb{d}(\mathcal{T}',\mathcal{T}''),\ \forall \mathcal{T},\mathcal{T}',\mathcal{T}'' \in \mathbb{T} \tag{D.2}$$

$$\mathbb{d}(\mathcal{T},\mathcal{T}') \leq C_{\text{dist}}\mathbb{d}(\mathcal{T}',\mathcal{T}),\ \forall \mathcal{T},\mathcal{T}' \in \mathbb{T} \tag{D.3}$$

for some uniform constant $C_{\text{dist}} > 0$. Defining then $\eta_{w,*}(\{K\})$ by

$$\eta_{w,*} = \sqrt{\sum_{K\in\{K\}} \eta_w(u_h(\mathcal{T}_*),K)^2} \tag{D.4}$$

as the estimates computed on the set $\{K\} \subseteq \mathcal{T}_*$ using the solution computed on $\mathcal{T}_*$. Then for $\mathcal{T}_* \in \text{refine}(\mathcal{T}_\circ)$, the axioms (using the notation of Feischl *et al.*[85]) are defined as

(A1) *Stability on non-refined elements:* where w is the quantity being estimated (e.g. u or ψ)

$$\exists C_{\text{stab}} > 0 :\ |\eta_{w,*}(\mathcal{T}_\circ \cap \mathcal{T}_*) - \eta_{w,\circ}(\mathcal{T}_\circ \cap \mathcal{T}_*)| \leq C_{\text{stab}}\mathbb{d}(\mathcal{T}_\circ,\mathcal{T}_*) \tag{D.5}$$

(A2) *Reduction on refined elements:*

$$\exists q_{\text{red}} \in (0,1),\ C_{\text{red}} > 0 :\ \eta_{w,*}(\mathcal{T}_* \backslash \mathcal{T}_\circ)^2 \leq q_{\text{red}} \eta_{w,\circ}(\mathcal{T}_\circ \backslash \mathcal{T}_*)^2 + C_{\text{red}} \mathbb{d}(\mathcal{T}_\circ, \mathcal{T}_*) \tag{D.6}$$

(A3) *General Quasi-Orthogonality:* $\exists \epsilon_{\text{qo}} > 0,\ C_{\text{qo}} \geq 1$ such that the series of meshes $\{\mathcal{T}_j\}$ from the adaptive process satisfy,

$$\sum_{j=l}^{N} \left(\mathbb{d}(\mathcal{T}_{j+1}, \mathcal{T}_j)^2 - \epsilon_{\text{qo}} \eta_{w,j}^2 \right) \leq C_{\text{orth}}(\epsilon_{\text{qo}}) \eta_{w,l}^2,\ \forall l, n \in \mathbb{N} : l \leq N \tag{D.7}$$

(A4) *Discrete reliability:* $\exists C_{\text{rel}} > 0 :\ \forall \mathcal{T}_\circ \in \mathbb{T},\ \mathcal{T}_* \in \text{refine}(\mathcal{T}_\circ)\ \exists \mathcal{R}_w(\mathcal{T}_\circ, \mathcal{T}_*) \subseteq \mathcal{T}_\circ,\ \mathcal{T}_\circ \backslash \mathcal{T}_* \subseteq \mathcal{R}_w(\mathcal{T}_\circ, \mathcal{T}_*)$ such that

$$\mathbb{d}(\mathcal{T}_*, \mathcal{T}_\circ) \leq C_{\text{rel}} \eta_w(\mathcal{R}_w(\mathcal{T}_\circ, \mathcal{T}_*)), \qquad \# \mathcal{R}_w(\mathcal{T}_\circ, \mathcal{T}_*) \leq C_{\text{rel}}(\mathcal{T}_\circ \backslash \mathcal{T}_*) \tag{D.8}$$

Though somewhat technical in nature these requirements allow separation of the proof of convergence from the specifics of the PDE and discretization, thus giving access to proofs of convergence for wide set of PDEs. In simplified language the axioms approximately mean

(A1) The change in estimates on non-refined elements must be limited by the distance between a mesh $\mathcal{T}_\circ$ and its refinement $\mathcal{T}_*$. This means that were a single element of $\mathcal{T}_\circ$ to be refined, the estimates as computed on the remaining elements can not change by an arbitrary amount.

(A2) The estimates computed on the new refined elements must be bounded above by a fixed fraction of the error in the non-refined elements, modulo the distance between the two meshes.

(A3) This is a relaxation of the notion of orthogonality when applied to the sequence of meshes generated by the adaptation.

(A4) The error estimate as computed on a set $\mathcal{R}_w(\mathcal{T}_\circ, \mathcal{T}_*) \subseteq \mathcal{T}_\circ$ of simplices that contains at least the refined elements of $\mathcal{T}_*$ must provide a reliable estimate of the distance between the two meshes. This must hold for all possible refinements and means that if the estimate on this envelope of the refined region is driven to 0, then the change must also be 0. Thus the indicator provides a reliable estimate of the distance between a mesh and the refinement.

BIBLIOGRAPHY

[1] P. Ciarlet, “Basic error estimates for elliptic problems,” in *Finite Element Methods (Part 1)*, ser. Handbook of Numerical Analysis, Vol. 2, Elsevier, 1991, pp. 17–351 (cit. on pp. 15, 65, 117).

[2] M. Yano and D. Darmofal, “An Optimization Framework for Anisotropic Simplex Mesh Adaptation: Application to Aerodynamic Flows,” AIAA 2012–0079, 2012 (cit. on pp. 15, 37, 39, 69).

[3] J. Bikard, T. Coupez, G. Della Valle, and B. Vergnes, “Simulation of bread making process using a direct 3D numerical method at microscale: Analysis of baking step,” *International journal of material forming*, Vol. 5, No. 1, pp. 11–24, 2012 (cit. on p. 15).

[4] A. T. Patera and J. Peraire, “A general Lagrangian formulation for the computation of *A-Posteriori* finite element bounds,” in *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, T. J. Barth and H. Deconinck, Eds., Springer-Verlag, 2002, pp. 159–206 (cit. on pp. 15, 24, 32, 78).

[5] X. Pennec, P. Fillard, and N. Ayache, “A Riemannian framework for tensor computing,” *Int. J. Comput. Vision*, Vol. 66, No. 1, pp. 41–66, 2006 (cit. on pp. 16, 35, 36, 66).

[6] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, “Log-Euclidean Metrics for Fast and Simple Calculus on Diffusion Tensors,” *Magnetic Resonance in Medicine*, Vol. 56, pp. 411–421, 2006 (cit. on p. 16).

[7] M. Yano, “An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes,” PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 2012 (cit. on pp. 17, 23, 90, 112).

[8] S. C. Brenner and L. R. Scott, *The Mathematical Thoery of Finite Element Methods, Third Edition*. New York: Springer, 2008 (cit. on pp. 17, 64, 74, 117).

[9] I. Babuska, B. A. Szabo, and I. N. Katz, “The p-version of the finite element method,” *SIAM Journal on Numerical Analysis*, Vol. 18, No. 3, pp. 515–545, 1981 (cit. on p. 18).

[10] W. H. Reed and T. R. Hill, “Triangular mesh methods for the neutron transport equation,” LA-UR-73-479, Los Alamos Scientific Laboratory, 1973 (cit. on p. 18).

[11] B. Cockburn, G. Karniadakis, and C. Shu, “The development of discontinuous Galerkin methods,” in *Lecture Notes in Computational Science and Engineering*, Vol. 11, Springer, 2000 (cit. on p. 19).

[12] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, “Unified analysis of discontinuous Galerkin methods for elliptic problems,” *SIAM journal on numerical analysis*, Vol. 39, No. 5, pp. 1749–1779, 2002 (cit. on pp. 19, 21).

[13] R. Hartmann and P. Houston, "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*, H. Deconinck, Ed., Von Karman Institute for Fluid Dynamics, Rhode Saint Genèse, Belgium, 2009 (cit. on p. 19).

[14] B. Cockburn, B. Dong, and J. Guzman, "A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems," *Math. Comp.*, Vol. 77, No. 264, pp. 1887–1916, 2008 (cit. on p. 19).

[15] N. Nguyen, J. Peraire, and B. Cockburn, "An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations," *J. Comput. Phys.*, Vol. 228, pp. 3232–3254, 2009 (cit. on p. 19).

[16] B. Cockburn, J. Gopalakrishnan, and R. Lazarov, "Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems," *SIAM J. Numer. Anal.*, Vol. 47, No. 2, pp. 1319–1365, 2009 (cit. on pp. 19, 21).

[17] B. Cockburn and J. Gopalakrishnan, "A characterization of hybridized mixed methods for second order elliptic problems," *SIAM Journal on Numerical Analysis*, Vol. 42, No. 1, pp. 283–301, 2004 (cit. on p. 19).

[18] B. Cockburn, J. Gopalakrishnan, and F.-J. Sayas, "A Projection-Based Error Analysis of HDG Methods," *Math. Comp.*, Vol. 79, pp. 1351–1367, 2010 (cit. on p. 19).

[19] B. Cockburn, J. Guzmán, S.-C. Soon, and H. K. Stolarski, "An analysis of the embedded discontinuous Galerkin method for second-order elliptic problems," *SIAM Journal on Numerical Analysis*, Vol. 47, No. 4, pp. 2686–2707, 2009 (cit. on p. 19).

[20] S. Güzey, B. Cockburn, and H. Stolarski, "The embedded discontinuous Galerkin method: Application to linear shell problems," *International journal for numerical methods in engineering*, Vol. 70, No. 7, pp. 757–790, 2007 (cit. on p. 19).

[21] D. S. Kamenetskiy, "On the relation of the Embedded Discontinuous Galerkin method to the stabilized residual-based finite element methods," *Applied Numerical Mathematics*, Vol. 108, pp. 271–285, 2016 (cit. on p. 19).

[22] N. Nguyen, J. Peraire, and B. Cockburn, "Hybridizable discontinuous galerkin methods," English, in *Spectral and High Order Methods for Partial Differential Equations*, ser. Lecture Notes in Computational Science and Engineering, J. S. Hesthaven and E. M. Rønquist, Eds., Vol. 76, Springer Berlin Heidelberg, 2011, pp. 63–84 (cit. on p. 19).

[23] T. J. R. Hughes, "A simple scheme for developing upwind finite elements," *Internat. J. Numer. Methods Engrg.*, Vol. 12, pp. 1359–1365, 1978 (cit. on pp. 20, 87).

[24] A. N. Brooks and T. J. R. Hughes, "Streamline upwind / Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations," *Comput. Methods Appl. Mech. Engrg.*, Vol. 32, pp. 199–259, 1982 (cit. on p. 20).

[25] T. J. R. Hughes, L. P. Franca, and G. M. Hulbert, "A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations," *Comput. Methods Appl. Mech. Engrg.*, Vol. 73, pp. 173–189, 1989 (cit. on pp. 20, 87).

[26] T. J. R. Hughes and G. Sangalli, "Variational Multiscale Analysis: The fine-scale Green's function, projection, optimization, localization, and stabilized methods," *SIAM J. Numer. Anal.*, Vol. 45, No. 2, pp. 539–557, 2007 (cit. on pp. 20, 87).

[27] F. Brezzi and A. Russo, "Choosing bubbles for advection-diffusion problems," *Mathematical Models and Methods in Applied Sciences*, Vol. 4, No. 04, pp. 571–587, 1994 (cit. on pp. 20, 87).

[28] F. Brezzi, L. Franca, T. Hughes, and A. Russo, "$B = \int G$," *Computer Methods in Applied Mechanics and Engineering*, Vol. 145, No. 3, pp. 329–339, 1997 (cit. on pp. 20, 87).

[29] T. J. Hughes, G. Scovazzi, P. B. Bochev, and A. Buffa, "A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 195, No. 19-22, pp. 2761–2787, 2006 (cit. on p. 20).

[30] G. Sangalli, "A discontinuous residual-free bubble method for advection-diffusion problems," *Journal of engineering mathematics*, Vol. 49, No. 2, pp. 149–162, 2004 (cit. on pp. 20, 88).

[31] I. Babuska and W. C. Rheinboldt, "A posteriori error analysis of finite element solutions for one-dimensional problems," *SIAM Journal on Numerical Analysis*, Vol. 18, No. 3, pp. 565–589, 1981 (cit. on pp. 21, 62).

[32] R. E. Bank and A. Weiser, "Some a posteriori error estimators for elliptic partial differential equations," *Mathematics of Computation*, Vol. 44, No. 170, pp. 283–301, 1985 (cit. on pp. 21, 62).

[33] M. Ainsworth and J. T. Oden, "A posteriori error estimation in finite element analysis," *Computer Methods in Applied Mechanics and Engineering*, Vol. 142, No. 1, pp. 1–88, 1997 (cit. on pp. 21, 24, 62).

[34] I. Babuška and W. C. Rheinboldt, "A-posteriori error estimates for the finite element method," *International Journal for Numerical Methods in Engineering*, Vol. 12, No. 10, pp. 1597–1615, 1978 (cit. on pp. 21, 23).

[35] I. Babuvška and W. C. Rheinboldt, "Error estimates for adaptive finite element computations," *SIAM Journal on Numerical Analysis*, Vol. 15, No. 4, pp. 736–754, 1978 (cit. on p. 21).

[36] R. Verfürth, "A posteriori error estimation and adaptive mesh-refinement techniques," *Journal of Computational and Applied Mathematics*, Vol. 50, No. 1, pp. 67–83, 1994 (cit. on pp. 21, 62, 78).

[37] R. Becker, P. Hansbo, and M. G. Larson, "Energy norm a posteriori error estimation for discontinuous Galerkin methods," *Computer Methods in Applied Mechanics and Engineering*, Vol. 192, No. 5-6, pp. 723–733, 2003 (cit. on p. 21).

[38] C. Carstensen, T. Gudi, and M. Jensen, "A unifying theory of a posteriori error control for discontinuous Galerkin FEM," *Numerische Mathematik*, Vol. 112, No. 3, pp. 363–379, 2009 (cit. on p. 21).

[39] B. Cockburn and W. Zhang, "A posteriori error estimates for HDG methods," *Journal of Scientific Computing*, Vol. 51, No. 3, pp. 582–607, 2012 (cit. on p. 21).

[40] ——, "A posteriori error analysis for hybridizable discontinuous Galerkin methods for second order elliptic problems," *SIAM Journal on Numerical Analysis*, Vol. 51, No. 1, pp. 676–693, 2013 (cit. on p. 21).

[41] I. Babuška and A. Miller, "The post-processing approach in the finite element method—part 1: Calculation of displacements, stresses and other higher derivatives of the displacements," *International Journal for numerical methods in engineering*, Vol. 20, No. 6, pp. 1085–1109, 1984 (cit. on p. 21).

[42] ——, "The post-processing approach in the finite element method- part 2: The calculation of stess intensity factors," *International Journal for numerical methods in engineering*, Vol. 20, No. 6, pp. 1111–1129, 1984 (cit. on p. 21).

[43] ——, “The post-processing approach in the finite element method—Part 3: A posteriori error estimates and adaptive mesh selection,” *International Journal for Numerical Methods in Engineering*, Vol. 20, No. 12, pp. 2311–2324, 1984 (cit. on p. 21).

[44] J. W. Barrett and C. M. Elliott, “Total flux estimates for a finite-element approximation of elliptic equations,” *IMA journal of numerical analysis*, Vol. 7, No. 2, pp. 129–148, 1987 (cit. on p. 21).

[45] R. Becker and R. Rannacher, “A feed-back approach to error control in finite element methods: Basic analysis and examples,” *East-West J. Numer. Math.*, Vol. 4, pp. 237–264, 1996 (cit. on pp. 21, 23, 32, 53, 56, 62, 78, 98).

[46] ——, “An Optimal Control Approach to A Posteriori Error Estimation in Finite Element Methods,” in *Acta Numerica*, A. Iserles, Ed., Cambridge University Press, 2001 (cit. on pp. 22, 55).

[47] O. C. Zienkiewicz and J. Z. Zhu, “The Superconvergent Patch Recovery and *a posteriori* error estimates. Part 1: The recovery technique,” *Internat. J. Numer. Methods Engrg.*, Vol. 33, pp. 1331–1364, 1992 (cit. on p. 22).

[48] ——, “The Superconvergent Patch Recovery and *a posteriori* error estimates. Part 2: Error Estimates and Adaptivity,” *Internat. J. Numer. Methods Engrg.*, Vol. 33, pp. 1365–1382, 1992 (cit. on p. 22).

[49] T. Richter and T. Wick, “Variational localizations of the dual weighted residual estimator,” *J. Computational Applied Mathematics*, Vol. 279, pp. 192–208, 2015 (cit. on pp. 23, 32, 41, 53, 56, 62, 69, 98).

[50] F. Bassi and S. Rebay, “GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations,” in *Discontinuous Galerkin Methods: Theory, Computation and Applications*, K. Cockburn and Shu, Eds., Berlin: Springer, 2000, pp. 197–208 (cit. on pp. 23, 93).

[51] H. A. Carson, D. L. Darmofal, M. C. Galbraith, and S. R. Allmaras, “Analysis of Output-Based Error Estimation for Finite Element Methods,” *Applied Numerical Mathematics*, Vol. 118, pp. 182–202, 2017 (cit. on pp. 23, 32, 55, 83).

[52] B. F. de Veubeke, “Displacement and equilibrium models in the finite element method,” O. Zienkiewicz and G. Holister, Eds., pp. 145–197, 1965 (cit. on p. 23).

[53] P. Ladeveze and D. Leguillon, “Error estimate procedure in the finite element method and applications,” *SIAM Journal on Numerical Analysis*, Vol. 20, No. 3, pp. 485–509, 1983 (cit. on pp. 23, 24, 43).

[54] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*. John Wiley & Sons, 2000, Vol. 37 (cit. on pp. 24, 27, 78).

[55] J. Peraire and A. Patera, “Bounds for linear-functional outputs of coercive partial differential equations: Local indicators and adaptive refinement,” *Studies in Applied Mechanics*, Vol. 47, pp. 199–216, 1998 (cit. on pp. 24, 62, 78).

[56] A. M. Sauer-Budge, J. Bonet, A. Huerta, and J. Peraire, “Computing bounds for linear functionals of exact weak solutions to Poisson’s equation,” *SIAM J. Numer. Anal.*, Vol. 42, No. 4, pp. 1610–1630, 2004 (cit. on pp. 24, 32, 62, 78).

[57] A. M. Sauer-Budge and J. Peraire, “Computing bounds for linear functionals of exact weak solutions to the advection-diffusion-reaction equation,” *SIAM J. Sci. Comput.*, Vol. 26, No. 2, pp. 636–652, 2004 (cit. on pp. 24, 32, 62, 78).

[58] L. Machiels, Y. Maday, and A. T. Patera, "A "flux-free" nodal Neumann subproblem approach to output bounds for partial differential equations," *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, Vol. 330, No. 3, pp. 249–254, 2000 (cit. on p. 25).

[59] N. Parés, P. Díez, and A. Huerta, "Subdomain-based flux-free a posteriori error estimators," *Computer Methods in Applied Mechanics and Engineering*, Vol. 195, No. 4-6, pp. 297–323, 2006 (cit. on p. 25).

[60] J. S. Wong, "A-Posteriori Bounds on Linear Functionals of Coercive 2nd Order PDEs Using Discontinuous Galerkin Methods," PhD thesis, Massachusetts Institute of Technology, 2006 (cit. on p. 25).

[61] M. Ceze and K. J. Fidkowski, "Output-driven anisotropic mesh adaptation for viscous flows using discrete choice optimization," AIAA 2010–170, 2010, p. 170 (cit. on pp. 25, 32, 63).

[62] R. Hartmann and P. Houston, "Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations," *J. Comput. Phys.*, Vol. 183, No. 2, pp. 508–532, 2002 (cit. on p. 25).

[63] R. Hartmann, "Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations," *Internat. J. Numer. Methods Fluids*, Vol. 51, pp. 1131–1156, 2006 (cit. on p. 25).

[64] A. Loseille and F. Alauzet, "Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error," *SIAM J. Numer. Anal.*, Vol. 49, No. 1, pp. 38–60, 2011 (cit. on pp. 25, 26, 32, 34, 63, 64).

[65] ——, "Continuous Mesh Framework Part II: Validations and Applications," *SIAM J. Numer. Anal.*, Vol. 49, No. 1, pp. 61–86, 2011 (cit. on pp. 25, 26, 32, 34, 63, 64).

[66] F. Alauzet and A. Loseille, "A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics," *Computer-Aided Design*, Vol. 72, pp. 13–39, 2016 (cit. on pp. 25, 26, 32, 36, 64).

[67] T. Michal and J. Krakos, "Anisotropic Mesh Adaptation through Edge Primitive Operations," AIAA 2012-159, January 2012 (cit. on pp. 26, 36, 99).

[68] F. Hecht, BAMG: Bidimensional Anisotropic Mesh Generator, `http://www-rocq1.inria.fr/gamma/cdrom/www/bamg/eng.htm`, INRIA-Rocquencourt, France, 1998 (cit. on p. 26).

[69] A. Loseille and R. Löhner, "Anisotropic Adaptive Simulations in Aerodynamics," AIAA 2010-169, 2010 (cit. on pp. 26, 36).

[70] P. C. Caplan, "Four-dimensional Anisotropic Mesh Adaptation for Spacetime Numerical Simulations," PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 2019 (cit. on pp. 26, 36, 48, 82, 112).

[71] D. A. Ibanez, "Conformal Mesh Adaptation on Heterogeneous Supercomputers," PhD thesis, Rensselaer Polytechnic Institute, 2016 (cit. on p. 26).

[72] M. Yano and D. L. Darmofal, "An Optimization-Based Framework for Anisotropic Simplex Mesh Adaptation," *J. Comput. Phys.*, Vol. 231, No. 22, pp. 7626–7649, 2012 (cit. on pp. 26, 32, 34, 35, 40, 41, 48, 50, 64, 70).

[73] Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. T. Huynh, *et al.*, "High-order CFD methods: Current status and perspective," *International Journal for Numerical Methods in Fluids*, Vol. 72, No. 8, pp. 811–845, 2013 (cit. on pp. 26, 90).

[74] K. Fidkowski, "A local sampling approach to anisotropic metric-based mesh optimization," in *54th AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2016 (cit. on pp. 26, 32, 64).

[75] M. A. Park, A. Balan, W. K. Anderson, M. C. Galbraith, P. C. Caplan, H. A. Carson, T. Michal, J. A. Krakos, D. S. Kamenetskiy, A. Loseille, F. Alauzet, L. Frazza, and N. Barral, "Verification of Unstructured Grid Adaptation Components," AIAA 2019-1723, 2019 (cit. on pp. 26, 90).

[76] V. Dolejší, "Anisotropic hp-adaptive method based on interpolation error estimates in the H 1-seminorm," *Applications of Mathematics*, Vol. 60, No. 6, pp. 597–616, 2015 (cit. on p. 26).

[77] A. M. Rangarajan, A. Balan, and G. May, "Mesh adaptation and optimization for Discontinuous Galerkin methods using a continuous mesh model," in *AIAA Modeling and Simulation Technologies Conference*, 2016, p. 2142 (cit. on pp. 27, 64).

[78] A. Rangarajan, A. Balan, and G. May, "Mesh Optimization for Discontinuous Galerkin Methods Using a Continuous Mesh Model," *AIAA Journal*, Vol. 56, No. 10, pp. 4060–4073, 2018 (cit. on p. 27).

[79] I. Babuška and M. Vogelius, "Feedback and adaptive finite element solution of one-dimensional boundary value problems," *Numerische Mathematik*, Vol. 44, No. 1, pp. 75–102, February 1984 (cit. on pp. 27, 62, 71).

[80] W. Dörfler, "A convergent adaptive algorithm for Poisson's equation," *SIAM Journal on Numerical Analysis*, Vol. 33, No. 3, pp. 1106–1124, 1996 (cit. on pp. 27, 62).

[81] P. Morin, R. H. Nochetto, and K. G. Siebert, "Data oscillation and convergence of adaptive FEM," *SIAM Journal on Numerical Analysis*, Vol. 38, No. 2, pp. 466–488, 2000 (cit. on pp. 28, 62).

[82] P. Binev, W. Dahmen, and R. DeVore, "Adaptive finite element methods with convergence rates," *Numerische Mathematik*, Vol. 97, No. 2, pp. 219–268, 2004 (cit. on pp. 28, 62).

[83] J. M. Cascon, C. Kreuzer, R. H. Nochetto, and K. G. Siebert, "Quasi-optimal convergence rate for an adaptive finite element method," *SIAM Journal on Numerical Analysis*, Vol. 46, No. 5, pp. 2524–2550, 2008 (cit. on pp. 28, 62).

[84] C. Carstensen, M. Feischl, M. Page, and D. Praetorius, "Axioms of adaptivity," *Computers & Mathematics with Applications*, Vol. 67, No. 6, pp. 1195–1253, 2014 (cit. on pp. 28, 62, 78, 121).

[85] M. Feischl, D. Praetorius, and K. G. Van der Zee, "An abstract analysis of optimal goal-oriented adaptivity," *SIAM Journal on Numerical Analysis*, Vol. 54, No. 3, pp. 1423–1448, 2016 (cit. on pp. 28, 62, 121).

[86] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, "Adaptive Remeshing for Compressible Flow Computations," *J. Comput. Phys.*, Vol. 72, pp. 449–466, 1987 (cit. on p. 32).

[87] M. J. Castro-Díaz, F. Hecht, B. Mohammadi, and O. Pironneau, "Anisotropic unstructured mesh adaptation for flow simulations," *Internat. J. Numer. Methods Fluids*, Vol. 25, pp. 475–491, 1997 (cit. on p. 32).

[88] D. A. Venditti and D. L. Darmofal, "Anisotropic grid adaptation for functional outputs: Application to two-dimensional viscous flows," *J. Comput. Phys.*, Vol. 187, No. 1, pp. 22–46, 2003 (cit. on p. 32).

[89] K. J. Fidkowski and D. L. Darmofal, "A triangular cut-cell adaptive method for higher-order discretizations of the compressible Navier-Stokes equations," *J. Comput. Phys.*, Vol. 225, pp. 1653–1672, 2007 (cit. on p. 32).

[90] T. Leicht and R. Hartmann, "Anisotropic mesh refinement for discontinuous Galerkin methods in two-dimensional aerodynamic flow simulations," *Internat. J. Numer. Methods Fluids*, Vol. 56, pp. 2111–2138, 2008 (cit. on p. 32).

[91] L. Formaggia, S. Perotto, and P. Zunino, "An anisotropic a-posteriori error estimate for a convection-diffusion problem," *Comput. Visual Sci.*, Vol. 4, pp. 99–104, 2001 (cit. on p. 32).

[92] L. Formaggia, S. Micheletti, and S. Perotto, "Anisotropic mesh adaptation in computational fluid dynamics: Application to the advection–diffusion–reaction and the Stokes problems," *Applied Numerical Mathematics*, Vol. 51, No. 4, pp. 511–533, 2004 (cit. on p. 32).

[93] P. Houston, E. H. Georgoulis, and E. Hall, "Adaptivity and A Posteriori Error Estimation for DG Methods on Anisotropic Meshes," *International Conference on Boundary and Interior Layers*, 2006 (cit. on pp. 32, 62).

[94] T. Richter, "*A posteriori* error estimation and anisotropy detection with the dual-weighted residual method," *Internat. J. Numer. Methods Fluids*, Vol. 62, pp. 90–118, 2010 (cit. on p. 32).

[95] T. Leicht and R. Hartmann, "Error estimation and anisotropic mesh refinement for 3d laminar aerodynamic flow simulations," *J. Comput. Phys.*, Vol. 229, pp. 7344–7360, 2010 (cit. on pp. 32, 90).

[96] K. Fidkowski, "Output-Based Mesh Optimization for the Embedded Discontinuous Galerkin Method," in *AIAA Aviation 2019 Forum*, 2019, p. 2950 (cit. on p. 33).

[97] K. Svanberg, "A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations," *SIAM Journal on Optimization*, Vol. 12, No. 2, pp. 555–573, 2002 (cit. on pp. 41, 99).

[98] S. G. Johnson, The NLopt nonlinear-optimization package, available at: `http://ab-initio.mit.edu/nlopt` (cit. on pp. 41, 99).

[99] M. Ainsworth and J. T. Oden, "A posteriori error estimators for second order elliptic systems part 2. An optimal order process for calculating self-equilibrating fluxes," *Computers & Mathematics with Applications*, Vol. 26, No. 9, pp. 75–87, 1993 (cit. on p. 43).

[100] N. A. Pierce and M. B. Giles, "Adjoint and defect error bounding and correction for functional estimates," *J. Comput. Phys.*, Vol. 200, No. 2, pp. 769–794, 2004 (cit. on p. 55).

[101] M. B. Giles, M. C. Duta, J.-D. M-uacute, and N. A. Pierce, "Algorithm developments for discrete adjoint methods," *AIAA journal*, Vol. 41, No. 2, pp. 198–205, 2003 (cit. on p. 57).

[102] R. Stenberg, "On some techniques for approximating boundary conditions in the finite element method," *Journal of Computational and applied Mathematics*, Vol. 63, No. 1-3, pp. 139–148, 1995 (cit. on p. 57).

[103] R. Verfürth, *A Posteriori Error Estimation Techniques for Finite Element Methods*. Oxford University Press, 2013 (cit. on p. 62).

[104] M. Ceze and K. J. Fidkowski, "Anisotropic hp-adaptation framework for functional prediction," *AIAA journal*, Vol. 51, No. 2, pp. 492–509, 2012 (cit. on p. 63).

[105] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Geometric means in a novel vector space structure on symmetric positive-definite matrices," *SIAM journal on matrix analysis and applications*, Vol. 29, No. 1, pp. 328–347, 2007 (cit. on pp. 68, 75, 80).

[106] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004 (cit. on p. 71).

[107] R. H. Nochetto, A. Veeser, and M. Verani, "A safeguarded dual weighted residual method," *IMA J. Numer. Anal.*, Vol. 29, pp. 126–140, 2009 (cit. on p. 78).

[108] C. Schwab, *p- and hp- Finite Element Methods*. Great Clarendon Street, Oxford, UK: Oxford Science Publications, 1998 (cit. on p. 78).

[109] G. Strang and G. J. Fix, *An Analysis of the Finite Element Method*. Wellesly, MA: Wellesley-Cambridge Press, 1988 (cit. on p. 82).

[110] D. Ibanez, N. Barral, J. Krakos, A. Loseille, T. Michal, and M. Park, "First Benchmark of the Unstructured Grid Adaptation Working Group," *Procedia Engineering*, Vol. 203, pp. 154–166, 2017, 26th International Meshing Roundtable (cit. on p. 82).

[111] E. C. Cyr, J. Shadid, and T. Wildey, "Approaches for adjoint-based a posteriori analysis of stabilized finite element methods," *SIAM Journal on Scientific Computing*, Vol. 36, No. 2, A766–A791, 2014 (cit. on p. 87).

[112] L. P. Franca, T. J. Hughes, and R. Stenberg, *Stabilized finite element methods for the Stokes problem*. Teknillinen korkeakoulu, 1991 (cit. on p. 87).

[113] L. P. Franca, S. L. Frey, and T. J. R. Hughes, "Stabilized finite element methods: I. Application to the advective-diffusive model," *Comput. Methods Appl. Mech. Engrg.*, Vol. 95, pp. 253–276, 1992 (cit. on p. 87).

[114] J. Douglas Jr. and T. Dupont, "Interior Penalty Procedures for Elliptic and Parabolic Galerkin Meethods," *Comput. Methods Appl. Sciences*, 1976 (cit. on p. 88).

[115] A. C. Huang, "An Adaptive Variational Mutltiscale Method with Discontinuous Subscales for Aerodynamics Flows," PhD thesis, Mass. Inst. of Tech., Department of Aeronautics and Astronautics, November 2019 (cit. on pp. 88, 90, 105).

[116] C. M. Klaij, J. J. W. van der Vegt, and H. van der Ven, "Space-Time Discontinuous Galerkin Method for the Compressible Navier-Stokes Equations," *J. Comput. Phys.*, Vol. 217, pp. 589–611, 2 2006 (cit. on p. 90).

[117] A. Riley and M. Lowson, "Development of a three-dimensional free shear layer," *Journal of Fluid Mechanics*, Vol. 369, pp. 49–89, 1998 (cit. on pp. 90, 93).

[118] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, "PETSc Users Manual," ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004 (cit. on p. 90).

[119] F. Bassi and S. Rebay, "High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations," *J. Comput. Phys.*, Vol. 138, No. 2, pp. 251–285, 1997 (cit. on p. 111).

[120] P. C. Caplan, R. Haimes, D. L. Darmofal, and M. C. Galbraith, "Anisotropic Geometry-Conforming *d*-simplicial Meshing via Isometric Embeddings," *Procedia Engineering*, Vol. 203, pp. 141–153, 2017, 26th International Meshing Roundtable (cit. on p. 111).

[121] P. C. Caplan, R. Haimes, and X. Roca, "Isometric Embedding of Curvilinear Meshes Defined on Riemannian Metric Spaces," in *Proceedings of the 27th International Meshing Roundtable*, 2018 (cit. on p. 111).

[122] A. Loseille, F. Alauzet, and V. Menier, “Unique Cavity-Based Operator and Hierarchical Domain Partitioning for Fast Parallel Generation of Anisotropic Meshes,” *Computer-Aided Design*, Vol. 85, pp. 53–67, 2017 (cit. on p. 111).

[123] M. Ceze and K. J. Fidkowski, “Constrained pseudo-transient continuation,” *Internat. J. Numer. Methods Heat Fluid Flow*, Vol. 102, pp. 1683–1703, 2015 (cit. on p. 119).