Unstructured Anisotropic All-Quad Mesh Adaptation Based on a Continuous Local Error Model for the Discontinuous Galerkin Method

Keigan MacLean

Department of Mechanical Engineering

McGill University, Montreal

December, 2020

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of

Master of Engineering

Abstract

This thesis will present a method of unstructured anisotropic adaptation for all-quad mesh suitable for use with high-order methods. The overall goal is to minimize the computational cost needed to perform highly accurate Computational Fluid Dynamics (CFD) simulations. With adaptive methods this can be accomplished through modifying the underlying spatial discretization of the domain. Here, this will be tailored to the particular problem at hand through a continuous local error model. In this form, the physical mesh geometry is replaced by the notion of a frame field, allowing us to extend previous ideas from metric-based adaptation methods. This will serve as a means to restate the discrete error minimization problem in order to determine the target element size and shape based on an optimal distribution of degrees of freedom.

Furthermore, the goal of most CFD simulations is to measure some particular output of interest (e.g. such as lift or drag). Therefore, extensions to the method will be examined for goal-oriented adaptation. This involves using the adjoint problem solution and dual-weighted residuals to target mesh refinement in areas most crucial to the accuracy of these measurements.

Finally, this work will also consider the implementation of a fully unstructured and anisotropic all-quad mesh generator capable of conforming to the aforementioned target elements. Here, the procedure is based on an L_p-CVT energy minimization problem used to determine updated node positions. Overall, this work will combine each of these aspects in order to establish a complete adaptive framework for use in solving high-order problems with the Discontinuous Galerkin method.

Résumé

Cette thèse présentera une méthode d'adaptation anisotropique non structurée pour les maillages tous quadrilatères, à utiliser avec les méthodes d'ordre élevé. L'objectif est de minimiser le coût de calcul nécessaire pour réaliser des simulations de dynamique des fluides (CFD) de haute précision. Avec les méthodes adaptatives, cela peut être réalisé en modifiant la discrétisation spatiale du domaine. Ici, cela sera adapté au problème particulier en question grâce à un modèle d'erreur locale continue. Sous cette forme, la géométrie physique du maillage est remplacée par la notion du "frame field", qui nous permet d'étendre les idées des méthodes d'adaptation basées sur la métrique. Cela nous permettra de reformuler le problème des erreurs discrètes afin de déterminer la taille et les formes des éléments en fonction d'une distribution optimale des degrés de liberté.

En outre, l'objectif de la majorité des simulations CFD est de mesurer un résultat d'intérêt (par exemple, la force du levage ou de la traînée). En conséquence, des extensions de la méthode seront examinées pour une adaptation orientée vers l'objectif. Cela implique l'utilisation de la solution du problème adjoint et des "dual-weighted residual" pour cibler le raffinement du maillage dans les domaines les plus cruciaux.

Enfin, ces travaux envisageront également la mise en œuvre d'un générateur de maillage entièrement non structuré et anisotrope, capable de se conformer aux éléments cibles susmentionnés. Ici, la procédure est basée sur un problème de minimisation de l'énergie L_p-CVT utilisé pour déterminer les positions actualisées des nœuds. Dans l'ensemble, ce travail combinera chacun de ces aspects afin d'établir un cadre adaptatif complet à utiliser pour résoudre les problèmes d'ordre élevé avec la méthode Galerkin discontinue.

Preface

The work of this Thesis involves several original contributions. While the topics of continuous error models and frame field based meshing have each been studied on their own, this work presents the first time they have been used in conjunction. This involved the modification of the error estimates making them suitable for use with quads in a new continuous form. Work conducted also involved the development of these features within the Computational Aerodynamic research group's open source CFD code PHiLiP, for use by other students and researchers alike. During this time, contributions were also made to the adjoint and flow solver modules, allowing for the results shown.

In addition, in the past minimal work has been conducted on the L_p-CVT mesh generation method. This work presents the first occasion where a discontinuous and unstructured background frame field has been used to define the anisotropic size targets guiding the all-quad remeshing process. It is also the first time this sort of generator has been used in a complete adaptive loop for solving numerical problems and in particular in conjunction with high-order methods. The hope of this work is that is sparks further interest into ties between these two areas and the ways in which recent independent developments could cumulatively prove especially powerful.

Acknowledgements

First, I would like to thank my supervisor, Professor Siva Nadarajah, for his constant insight and guidance throughout my Masters. His teachings helped enlighten me to a field filled with deep and interesting topics.

Next, I am grateful for my colleagues from the McGill Aerodynamics Group for the many discussions, late night study sessions, soccer games and coffee breaks that we shared. Academically, they have helped me gain insight well beyond the scope of the current project. Outside of school, I hope we remain in touch as we progress through our careers.

To my family, friends and girlfriend, thank you for the constant support and care you've shown over the past few years. The effort you've made to better understand my studies means the world to me. Thank you for letting me vent about "that one small bug" and sharing in the excitement with each major step along the way.

Finally, I gratefully acknowledge the financial support received through the Natural Sciences and Engineering Research Council (NSERC) CGS-M Scholarship, and from the department of Mechanical Engineering at McGill University.

Table of Contents

List of Figures

List of Tables

Nomenclature

Flow/primal problem

u	Exact flow solution
u_h	Discrete flow solution
u_h^k	Element-wise flow solution
$\mathcal{R}$	Residual
$\mathcal{F}$	Flux vector
$\hat{\mathcal{F}}$	Numerical flux vector
S	Source term

Discretization

Ω	Physical domain
Γ	Physical boundaries
Ω_h	Computational domain
Ω_k	k^{th} element
φ_i	Local shape functions
$\hat{\varphi}_i$	1D shape functions
$\mathcal{Q}_h$	Quadrangulation of the domain
p_k	Local polynomial order

$\mathcal{Q}_{hp}$ Discrete hp mesh

$\mathcal{V}_{hp}$ hp solution space

Discrete error minimization

N_k Number of local shape functions

N_{hp} Total degrees of freedom

Π_{hp} Projection operator

E_h Discrete error

C Maximum number of degrees of freedom

Adjoint/dual problem

$\mathcal{J}(u)$ Functional of interest

ψ Adjoint variable

I_h^H Prolongation operator

u_h^H Prolonged coarse to fine grid solution

ψ_h Discrete adjoint variable (from prolonged flow solution)

η_k Dual-weighted residual

Riemannian metric space

$\mathcal{M}$ Riemannian metric tensor

$\mathcal{B}_{\mathcal{M}}$ Metric ellipse

θ Ellipse orientation

h_1, h_2 Principle axes lengths

$\boldsymbol{e}$ Mesh edges

$\mathcal{T}_h$ Unit-mesh triangulation

Frame field

$\mathcal{F}$	Frame field
$f_{\boldsymbol{x}}$	Frame at point $\boldsymbol{x}$
$\boldsymbol{v}, \boldsymbol{w}$	Vector axes
$\mathcal{B}_f$	Representative parallelogram
V	Linear transformation from reference element
$\boldsymbol{e}_i$	Reference element axes
h_1, h_2	Principle axes lengths
θ_1, θ_2	Principle axes orientations
h	Size
θ	Orientation
ρ	Anisotropy

L_p-CVT energy

E_{L_p}	L_p-CVT energy
$M(\boldsymbol{y})$	Anisotropic metric
$\boldsymbol{x}_i$	Delaunay nodes
Ω_i	Voronoi cells
$\boldsymbol{c}_j$	Voronoi vertices
f_k	Voronoi cell facets
f'	Reference triangle
I_k	Local energy
L_M	Metric space edge length
$\boldsymbol{\gamma}(t)$	Parametric boundary

Continuous mesh model

$d(\boldsymbol{x})$ Local element density

$\mathcal{P}(\boldsymbol{x})$ Polynomial distribution

$w(\boldsymbol{x})$ Distribution for the number of shape functions

$\mathcal{N}_h$ Continuous number of mesh elements

$\mathcal{N}_{hp}$ Continuous mesh complexity

$\mathcal{C}$ Complexity target

Continuous error model

e^{int} Interpolation error

$\tilde{u}_h$ Reconstructed $p+1$ order solution on patch $D(k)$

A_1 Largest $p+1$ directional derivative

φ Orientation of A_1

A_2 $p+1$ directional derivative in perpendicular to A_1

$e(\boldsymbol{x})$ Continuous local error model

$\mathcal{E}$ Continuous global error model

Goal-oriented adaptation

I_k^c Current cell size

I_k Target cell size

α_k Scaling factor

ξ_k Quadratic fitting in logarithmic space

η_{ref} Reference dual-weighted residual value

Chapter 1

Introduction

Computational Fluid Dynamics (CFD) serves as a crucial tool to produce highly accurate engineering simulations. The results it provides play an important role in both the design and analysis in several key industries, such as: automotive, aerospace, turbomachinery, marine, nuclear energy, architecture and more [14, 31, 47, 61, 117, 129]. Recent developments in High-Performance Computing (HPC) have helped to mitigate limitations regarding the constant growth in the scale and complexity of problems considered. However, this is expensive (both computationally and monetarily) and on its own inadequate to keep up with current trends [116]. As a result, current developments to the modelling, discretizations and meshing procedures are crucial to reduce the associated computational cost needed to perform high-fidelity simulations.

In particular, high-order methods, such as the Discontinuous Galerkin (DG) Method, show potential in helping to achieve these goals. This is due in part to the higher-order accuracy they exhibit, allowing for more efficient computation per degree of freedom [129]. Furthermore, as they rely on arbitrary spatial discretizations of the domain (h) and choices of polynomial orders (p), adaptive methods can be used to effectively tailor the solution space to the particular problem at hand [113]. Together these quantities form the hp mesh. However, the results shown here will mainly concern the uniform (but not necessarily linear) polynomial case. Additionally, most engineering simulations are concerned with the

measurement of a particular quantity of interest (either for analysis or quantification of designs), for example lift and drag in aerodynamics. Therefore, this offers the potential for the adaptation to be targeted towards the areas most crucial for accurate measurement of this value. Thus, also preventing the computational cost increase associated with refining other regions of the flow. As will be examined later on, this can be achieved through goal-oriented adaptation methods using an adjoint-based error indicator [43].

To date, most research in this area has concentrated on the triangular meshing case (or more generally simplex mesh in nD). However, research into the use of quadrilaterals (or more generally tensor-product elements in nD) has been significantly more limited. This is despite the fact that these elements are often preferred in some anisotropic regions (such as shocks and boundary layers) due to their alignment with the flow behaviour. As a result, this thesis will consider the potential impact of recent developments in the field of unstructured anisotropic quad mesh generation. Namely, this will incorporate the notion of "frame fields" into a continuous representation of the all-quad mesh. However, as many of these works originate from the field of computer graphics, and are based on meshing closed manifold structures (e.g. 3D models) [15], they cannot be directly applied with the bounded domains used for CFD. Therefore, as a secondary aspect of this work, we will also consider the implementation of an L_p-CVT mesh generator, extending from previous works [10,40,41,74], to conform with anisotropic quad meshing targets.

Finally, the topic of mesh generation remains one of the most labour intensive steps of the CFD process. The NASA CFD Vision 2030 report emphasized the need for more automated analysis, and highlighted mesh adaptation in particular as one of the significant remaining bottlenecks of the simulation workflow [116]. Overall, this work will extend previous unstructured adaptation methods to the case of tensor-product elements and establish an adaptive framework for solving high-order problems on unstructured all-quad mesh for both feature-based and goal-oriented adaptations. While the work here is more fundamental, reflected in the smaller scale of problems, it represents an important foundation towards tackling more ground-breaking achievements to come.

1.1 Simplex Mesh Adaptation

Mesh generation is a key step in accurately discretizing the physical domain of interest in order to make it suitable for use with numerical schemes, such as the Finite Element Methods (FEM) and the Finite Volume Methods (FVM). There are many works outlining the fundamentals of this topic, notably, the book by Frey and George provides a good overview of both classical methods and aspects of the relevant modern methods [46]. Of particular interest here are the set of anisotropic simplex mesh generation methods based on Riemannian metric spaces. The specified metric field is used to modify the distance computation in adaptive "metric-based" mesh generators, resulting in local control over the size and stretching of anisotropic grids [1, 58, 79]. Original works in this area were based on the use of a 2D Delaunay triangulation [49,90]. More modern works have incorporated constrained Delaunay kernels [48, 58], hybrid-frontal approaches [68], and mesh modifications operators [45, 86] for use in both 2D and 3D. Recent works have also developed methods based on metric orthogonal point insertions schemes and the usage of triangulations in the L_∞ norm to produce right-angled triangular [78,109] and tetrahedral mesh [9].

Adaptive methods based on these techniques incorporate the duality between the Riemannian metric space and the discrete mesh [1]. Described as a "continuous mesh framework" [79, 80], this modification provides access to mathematical tools such as calculus of variations, providing a means to directly optimize the continuous mesh [1, 79]. This representation offers considerable benefits over the "discrete mesh" that is obtained from the generation process, as it avoids the combinatorial structure of decisions involved in the insertion/removal of nodes, changes in element connectivity, etc. The review paper by Alauzet and Loseille [1] offers an overview of past works in this area. This framework was originally introduced by the same authors in [79, 80] for the linear case. In this framework, the solution behaviour is related to the Riemannian metric through the continuous linear interpolate. This bounds the error associated with the class of isotropic

elements in the Riemannian space and leads to a continuous local error model. Here, as the error is dominated by the quadratic terms, the metric tensor is derived from the hessian of the flow and therefore it is commonly referred to as a Hessian-based method. Later, this work was extended to include goal-oriented adaptation [82], norm-oriented adaptation [83] and unsteady flows [11].

For the high-order case, methods for hp adaptation have also been developed based on such a continuous interpolation error model. This requires the model be extended to account for the dominant $p+1$ order error terms. Dolejší proposed a method based on an approximate quadratic form in 2D [34, 35], which was later also extended to 3D tetrahedral mesh adaptation [105]. More recent works have also considered the hp goal-oriented adaptation case [6, 7, 104]. For a more complete review on goal-oriented methods see [43]. Alternative metric-based high-order methods have also been studied using Lagrange finite elements [59], the level curves of high-order derivative tensors [22, 23], recasting in metric-logarithmic space [29], optimization from local surveying of split configurations [133] and within our research group based on a global hp-mesh optimization with analytic gradients [112, 113]. This work aims to extend the continuous mesh and error models for anisotropic unstructured quad meshing.

1.2 All-Quad Mesh Generation

For the purposes of our study, there also exists various forms of quad mesh generators. These include but are not limited to: automatic block decomposition methods [67, 127], packing [128], paving [13], advancing-front [109], parametrization-based [16], and energy minimization [127] methods. For a more detailed review of quad-meshing techniques see Bommes et al. [15]. Broadly speaking, these can be categorized based on several criteria: the level of structure in the mesh, the presence of anisotropic cells and whether the method is direct or indirect. For the case of indirect methods, the method first begins by generating a simplex based mesh (usually targeting right-angled cells) then combining

them based on a matching approach such as Blossom-Quad [110]. Other matching criteria also exist, based on a variety of mesh-quality metrics and ordering schemes [17,70,97].

Several of the previously mentioned methods incorporate the use of a particular type of orientation field, called a "cross field" (or 4-RoSy fields) [67, 98, 109, 127]. This can be specified by a set of orthogonal unit-length vectors prescribing the alignment of the mesh. In certain applications this is used to decompose the underlying mesh topology, determining mesh-singularities and separatrices to define the block structure [67,98,127]. As this does not include a scale, some methods couple this idea with a size-field [109]. Panozzo et al. later generalized this concept to "frame fields" (or 4-PolyVector fields), as the non-orthogonal and non-unit length case [32, 99]. Here, the local frame can be described by a coupled set of vector fields describing both the target alignment and size of the element axes. Together these methods are sometimes referred to as "field-guided", and the representations lumped together as "directional fields". These describe general coupled vector sets with rotational symmetry on manifold surfaces [123]. Attempts have also been made to extend this to 3D frame fields for hexahedral meshing, however, challenges remain in obtaining a suitable singularity graph [63, 64, 106].

We present a method which computes target frame fields based on the continuous error estimates discussed before, replacing the Riemannian metric field for quad mesh generation. Unfortunately, as these field-guided methods originate from the field of computer graphics, in the anisotropic case they are not currently suitable for use with bounded domains (to the authors knowledge, in the public domain). This introduces a significant number of additional constraints on the mesh, however, attempts are being made to solve this challenge [88]. Therefore, the second main contribution of this work will involve the design and implementation of an energy-based L_p-CVT mesh generator conforming to an input frame field and outputting an all-quad mesh. Originally proposed by Lévy and Liu in [74], this method extends a standard Centroidal Voronoi Tessellation (CVT) [36] to the anisotropic and higher-order norm case. It involves an energy minimization relative to a point distribution based on the L_p norm taken over the Voronoi

cells. It is observed that with a sufficiently large choice of p, these cells tend to align with the metric axes and form rectangular shapes. This results in a Delaunay triangulation with elements that are nearly right-angled, making them suitable for recombination into quads. Later, contributions by Baudouin et al. [10] introduced a new clipping procedure for the interior Voronoi cells and incorporated Blossom-Quad for recombination of the elements. However, it was limited in scope to the isotropic case and required starting from an initial size-field generated triangular mesh. In a set of works by Ekelschot et al. [40, 41], the anisotropic case was further explored and an alternative boundary treatment was proposed based on the reconstruction of the ghost Voronoi cells at the wall. In this work, we will consider extensions and challenges resulting from the inclusion of a frame field defined on an unstructured background mesh.

1.3 Thesis Overview

The remainder of the thesis will be organized as follows. First, in Chapter 2, we will review the underlying numerical discretization of the high-order method. We will also introduce the adjoint equations and corresponding dual-weighted residual to be used later with goal-oriented adaptation. Second, in Chapter 3, we will review various mathematical constructs needed for meshing. This chapter will also provide a mathematical background on the Riemannian metric field used in triangular mesh generation and the details of the frame fields substituted in its place for quad meshing. Chapter 4 will provide a description of the L_p-CVT method and the mesh generator that has been implemented. This relies on an energy minimization procedure used to distribute the mesh nodes in the anisotropic space. Additionally, we will discuss the various pre-processing and post-processing steps required to establish an interface with our in-house flow solver. Following this, Chapter 5 will introduce the continuous error and mesh models and discuss how they are used locally to determine the target frame field orientation and anisotropy. Using calculus of variations, a global continuous error problem will be solved to determine

the optimal size distribution of the mesh. Next, Chapter 6 will present results validating the mesh generator and using these two techniques together. Finally, concluding remarks and possible future works are discussed in Chapter 7.

Chapter 2

Adaptive High-Order Methods

To begin, CFD aims to solve problems based on physical flow phenomenon. In most cases, the underlying principles can be expressed in the form of a conservation law as

$$\mathcal{R}(u) = \frac{\partial u}{\partial t} + \nabla \cdot \mathcal{F}(u) - S(u) = 0, \tag{2.1}$$

where u is the flow solution, $\mathcal{R}(u)$ is the residual, $\mathcal{F}(u)$ is the flux vector and $S(u)$ is a source term. This representation includes cases such as advection-diffusion, Euler equations, the Navier-Stokes, etc. [3]. Computationally, this involves determining a discrete solution u_h corresponding to $\mathcal{R}_h(u_h) = 0$.

Of note are the class of high-order methods capable of solving these problems. The main benefits these methods offer is their ability to achieve higher-order accuracy, potentially providing reductions in the solution error per degree of freedom (or allowing a coarser grid to achieve equivalent results) [129]. There exist diverse high-order methods based on the Finite Difference Method (FDM), Finite Volume Method (FVM) and Finite Element Method (FEM) schemes for structured [39] and unstructured mesh [39, 130]. Of particular interest in the present work are the Discontinuous FEM such as the Discontinuous Galerkin (DG) method. In this Chapter, the DG method will be introduced along with some basic necessities for feature-based and goal-oriented adaptive methods. Namely, introducing the adjoint equations and dual-weighted residual (DWR).

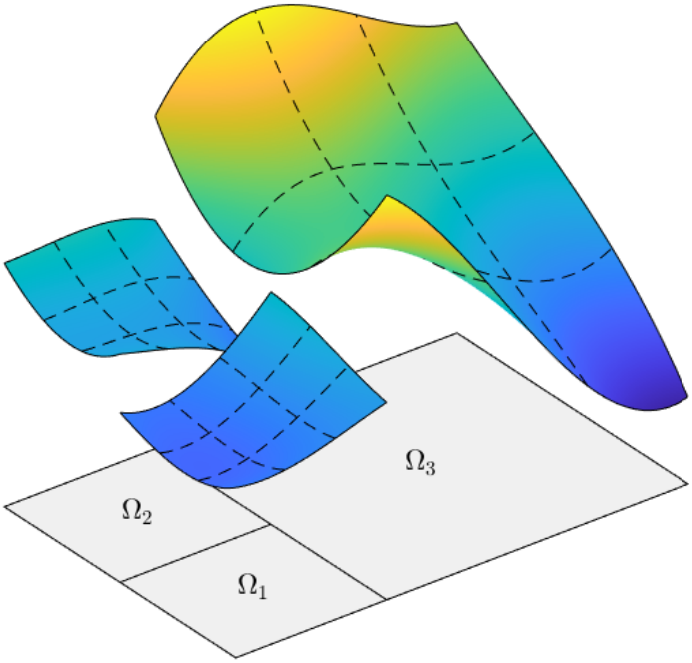


Figure 2.1: Example of function in discontinuous solution space, $\mathcal{V}_{hp}$.

2.1 Discontinuous Galerkin Method

The Discontinuous Galerkin (DG) method is one such class of high-order method. It combines an element-wise shape function discretization from Finite Element Methods (FEM) with the flexibility in allowing discontinuities at cell-interfaces from Finite Volume Methods (FVM) [60]. Overall, the combination of these aspects allow it to offer high-order accuracy on a compact stencil involving only direct neighbours [24]. This makes the method flexible, both in its ability to capture complex domains and through the use of varying polynomial orders (from the lack of conformality requirements).

The method was first presented by Reed and Hill in [108] as a method for solving the steady-state neutron transport equation. Since, it has been expanded to cover a wide range of fluid problems including advection-diffusion cases [62], the Euler equations [56] and the Navier-Stokes equations [8]. For a complete overview of the DG method, the book by Hesthaven and Warburton [60] provides an indepth guide.

Here, from [60], we seek a solution that is a piecewise sum of the shape functions over each element in the computational domain $\Omega_k \in \Omega_h$

$$u_h(x,t) = \bigoplus_{\Omega_k \in \Omega_h} u_h^k(x,t) \in \mathcal{V}_{hp}. \tag{2.2}$$

The discontinuous hp solution space is defined by the set of basis functions on each element

$$\mathcal{V}_{hp} = \bigoplus_{\Omega_k \in \Omega_h} \{\varphi_i(\Omega_k)\}_{i=1}^{N_k(p_k)}, \tag{2.3}$$

where $\varphi_i(\Omega_k)$ is the i^{th} shape function defined on element k. See Fig. 2.1 for an example of a function in this discontinuous space. Additionally, $N_k(p_k)$ is the number of shape functions for an element k of order p_k. For example,

$$N_k(p_k) = p_k + 1 \qquad \text{in 1D}, \tag{2.4}$$

$$N_k(p_k) = (p_k + 1)^2 \qquad \text{in 2D for quads}, \tag{2.5}$$

$$N_k(p_k) = \frac{1}{2}(p_k + 1)(p_k + 2) \qquad \text{in 2D for tris}. \tag{2.6}$$

For simplicity we will assume the steady-state solution case throughout. Here, the local element-wise solution can be written in the form

$$u_h^k(x) = \sum_{i=1}^{N_k(p_k)} u_i^k \varphi_i(x), \quad \forall x \in \Omega_k. \tag{2.7}$$

In particular for the case of tensor-product elements (e.g. lines, quads, hexes, etc.), these shape functions can be decomposed along each reference axis of the element. For example, in 2D as written in the reference coordinates

$$u_h^k(\xi,\eta) = \sum_{i=0}^{p_k} \sum_{i=0}^{p_k} u_{i,j}^k \hat{\varphi}_i(\xi) \hat{\varphi}_j(\eta), \tag{2.8}$$

where $\hat{\varphi}$ are the 1D shape functions. For our purposes, these will be the 1D Lagrange polynomials with Gauss-Lobatto support points [4]. This decomposition along the reference axes also provides a direct means to incorporate curved domains. The basic derivation of the scheme involves multiplying through Eq. 2.1 by a test function $\phi_h \in \mathcal{V}_{hp}$ and performing integration by parts

$$\begin{aligned}\int_{\Omega_k} \phi_h \frac{\partial u_h}{\partial t} d\Omega - \int_{\Omega_k} \nabla \phi_h \cdot \mathcal{F}(u_h) d\Omega - \int_{\Omega_k} \phi_h S(u_h) d\Omega \\ + \int_{\partial\Omega_k} \phi_h \hat{\mathcal{F}}(u_h^+, u_h^-, n) d\Gamma = 0, \quad \forall \phi_h \in \mathcal{V}_{hp},\end{aligned} \tag{2.9}$$

which results in the weak form of the DG method and will be used throughout the results. Here $\hat{\mathcal{F}}$ represents the numerical flux function describing the behaviour of the solution at the discontinuous interface between cells.

Given a smooth exact solution u, the DG method exhibits an optimal convergence rate of $\mathcal{O}(h^{p+1})$ in the L_2 norm (making it a high-order method) [73,111]. This assumes a quasi-uniform grid and that the edges are not aligned with the characteristic directions [60]. However, this result has also been demonstrated computationally in more general cases. On the other hand, this behaviour cannot be expected outside of the asymptotic regime, when relevant solution scales have been sufficiently resolved [80]. Many other factors, such as discontinuities, can also lead to sub-optimal convergence rates.

The current work has been implemented within the scope of our high-order research group in-house flow solver, PHiLiP, currently under development. It is based on the deal.II finite element library [4], which at its core relies on p4est [20] (or p8est in 3D) for the efficient parallel quad/octree data structures used in its computations. This restricts its usage to tensor-product elements, a limitation which becomes particularly cumbersome for unstructured meshing. As a result, this often causes the need to split a triangular mesh to obtain an acceptable input for complex domains. However, this leads to many poor quality elements and a large number of mesh singularities. This work hopes to partially address this challenge and help improve the flexibility of the solver in the future.

2.2 Discrete Error Minimization

As stated before, the goal of mesh adaptation is to increase the accuracy of the solution, here measured through the L^q norm of the discretization error. This should be done while simultaneously limiting the associated computational cost, measured by the total number of degrees of freedom (DOFs).

First, we define $k \in \mathcal{Q}_h$ as the set of quads forming the quadrangulation on the discrete domain Ω_h. Additionally, for the DG method, each element may have a different associated polynomial order p_k. In general, we can define the set of polynomial degrees as $\boldsymbol{p} = \{p_k, \forall k \in \mathcal{Q}_h\}$, however, for our purposes we will assume it to be uniform (but not necessarily linear). Together, these two sets describe the complete discretization of the functional space from Eq. 2.3. This provides both the geometric elements h, and the local polynomial orders p to form the complete hp mesh $\mathcal{Q}_{hp} = \{\mathcal{Q}_h, \boldsymbol{p}\}$. From this, we can measure the total degrees of freedom represented by the solution space

$$N_{hp}(\mathcal{Q}_{hp}) = \sum_{k \in \mathcal{Q}_h} N_k(p_k), \tag{2.10}$$

where $N_k(p_k)$ is known from Eq. 2.5 for the 2D quad case.

Next, given an exact solution defined on the domain $u \in C^\infty(\Omega)$, the error in the discrete solution $u_h \in \mathcal{V}_{hp}$ measured in the L^q norm is

$$E_h(u_h) = \|u - u_h\|^q_{L^q(\Omega_h)} = \int_{\Omega_h} (u - u_h)^q \, \mathrm{d}\mathbf{x}. \tag{2.11}$$

Using this, the discretization error associated with a given hp mesh will be based on the projected discrete solution which minimizes the above quantity. Therefore, we can define the optimal projection operator for this solution space as

$$\Pi_{hp} u = \underset{u_h \in \mathcal{V}_{hp}}{\operatorname{argmin}} \, E_h(u_h). \tag{2.12}$$

Formally, we can state the global discrete error minimization problem as follows:

Problem 1 *Given the existence of the exact solution $u \in C^\infty(\Omega)$ and a maximum number of degrees of freedom C, solve*

$$\begin{aligned} \min_{\mathcal{Q}_{hp}} \quad & E_h(\Pi_{hp} u) = \|u - \Pi_{hp} u\|^q_{L^q(\Omega_h)} \\ s.t. \quad & N_{hp}(\mathcal{Q}_{hp}) \leq C \end{aligned} \tag{2.13}$$

However, practically speaking, this statement is intractable [35, 80]. Challenges result from the inclusion of discrete decisions (e.g. insertion/removal of nodes, element connectivity, splitting) along with otherwise continuous parameters (e.g. node coordinates, scale, anisotropy) in the problem of generating the mesh. Therefore, this problem will be reformulated in the continuous setting in the coming chapters.

2.3 Adjoint Equations

Accurately solving the flow problem is crucial for better understanding of the global flow physics. However, in real world applications this is typically not the end-goal and may not be feasible to fully capture. Instead, practical simulations aim to measure some observable quantity for design or analysis purposes, such as the lift/drag on an airfoil or the heat transfer through a cooling pipe. These quantities can be expressed as a functional of the flow solution. Written in integral form

$$\mathcal{J}(u) = \int_\Omega g_\Omega(u) d\Omega + \int_\Gamma g_\Gamma(u) d\Gamma. \tag{2.14}$$

The adjoint problem is based on the concept of duality between this functional and the primal (or flow) solution. Therefore, it is sometimes also referred to as the dual-problem. Here, a secondary set of equations are solved in order to determine the sensitivity of the functional of interest relative to the primal solution.

This method originates from the field of control theory [75]. It was originally introduced to aerospace applications for the purpose of aerodynamic design optimization by Jameson [65, 66]. This was later proposed as a method for error estimation and a means of attaining functional superconvergence including some early works by Giles and Pierce

using an adjoint error correction term [52,53,103]. Namely, this involved the introduction of the dual-weighted residual (DWR) that will be discussed in the next section. It was first used in goal-oriented mesh adaptation in a series of papers by Venditti and Darmofal [124–126]. Since these times, the adjoint has been extensively used throughout other goal-oriented adaptive methods. This is due to the significant benefits it offers towards accurately determining the functional value over a more traditional feature-based adaptation. It has also been incorporated in some norm-oriented adaptation methods, targeting both the flow and output errors in equal or weighted parts [83].

To derive these equations, we can start by introducing an auxiliary variable x, such as a design variable, to be used in the minimization of the functional of interest [101]

$$\begin{aligned} \min_{x} \quad & \mathcal{J}(u,x), \\ \text{s.t.} \quad & \mathcal{R}(u,x) = 0, \end{aligned} \tag{2.15}$$

where our flow residual from Eq. 2.1 acts as a constraint. Note, that this can also easily be modified for maximization or attaining a target value (e.g. specified C_L requirements). This results in the Lagrangian

$$\mathcal{L}(u,x,\psi) = \mathcal{J}(u,x) - \psi^T \mathcal{R}(u,x), \tag{2.16}$$

where ψ is the Lagrange multiplier or adjoint variable. Evaluating the differential with respect to the auxiliary variable

$$\frac{\partial \mathcal{L}}{\partial x} = \left[\frac{\partial \mathcal{J}}{\partial x} - \psi^T \frac{\partial \mathcal{R}}{\partial x}\right] + \left(\frac{\partial u}{\partial x}\right)^T \left[\frac{\partial \mathcal{J}}{\partial u} - \psi^T \frac{\partial \mathcal{R}}{\partial u}\right], \tag{2.17}$$

which must be set to 0 to find an optimal solution. Here, we notice that the second term can be eliminated by solving

$$\left[\frac{\partial \mathcal{R}}{\partial u}\right]^T \psi = \left(\frac{\partial \mathcal{J}}{\partial u}\right)^T, \tag{2.18}$$

which is the adjoint problem.

2.4 Dual Weighted Residual

Instead of adapting over all significant behaviours in the flow, we would like to target only the parts most relevant to the output measurement of interest (known as goal-oriented adaptation). This is primarily accomplished through the introduction of the dual-weighted residual (DWR). It can be derived by comparing two flow solutions, one on a coarse grid u_H and one on a fine grid u_h

$$\mathcal{R}_H(u_H) = 0, \tag{2.19}$$

$$\mathcal{R}_h(u_h) = 0, \tag{2.20}$$

where the fine grid can be the enriched space from either h or p refinements. Additionally, there must exist a prolongation operator $u_h^H = I_h^H u_H$ transferring the solution from the coarse grid to the fine grid. For the purposes of this research, a $p+1$ solution space was used for the fine grid solution with prolongation via direct injection. Then, we can approximate the projected residual equation using a Taylor series expansion of Eq. 2.20

$$\mathcal{R}_h(u_h) = 0 \approx \mathcal{R}_h(u_h^H) + \left[\left. \frac{\partial \mathcal{R}_h}{\partial u_h} \right|_{u_h^H} \right] \left(u_h - u_h^H \right), \tag{2.21}$$

where higher-order terms are ignored and the exact value is 0 (for a converged fine grid solution). This can be rewritten as

$$\left[\left. \frac{\partial \mathcal{R}_h}{\partial u_h} \right|_{u_h^H} \right] \left(u_h - u_h^H \right) = -\mathcal{R}_h(u_h^H). \tag{2.22}$$

Similarly, for the functional we obtain

$$\mathcal{J}_h(u_h) \approx \mathcal{J}_h(u_h^H) + \left(\left. \frac{\partial \mathcal{J}_h}{\partial u_h} \right|_{u_h^H} \right) \left(u_h - u_h^H \right). \tag{2.23}$$

Assuming the fine grid solution offers a sufficiently good approximation of the exact solution $\mathcal{J}_h(u_h) \approx \mathcal{J}(u)$ compared to the coarse grid value. Then, we can express the functional error

$$\mathcal{J}_h(u_h) - \mathcal{J}_h(u_h^H) \approx \mathcal{J}(u) - \mathcal{J}_h(u_h^H) = \left(\left. \frac{\partial \mathcal{J}_h}{\partial u_h} \right|_{u_h^H} \right) \left(u_h - u_h^H \right), \tag{2.24}$$

where noticing that

$$\left[\left. \frac{\partial \mathcal{R}_h}{\partial u_h} \right|_{u_h^H} \right]^T \psi_h = \left(\left. \frac{\partial \mathcal{J}_h}{\partial u_h} \right|_{u_h^H} \right)^T, \tag{2.25}$$

is the discrete fine grid adjoint problem with adjoint solution ψ_h computed based on the prolonged coarse grid solution. We can substitute Eq. 2.22 and rewrite the error as

$$\mathcal{J}(u) - \mathcal{J}_h(u_h^H) = -\psi_h^T \mathcal{R}_h(u_h^H). \tag{2.26}$$

To be conservative, taking the absolute value, the output error will be bounded by the sum of cell-wise DWR η_k values

$$|\mathcal{J}(u) - \mathcal{J}_h(u_h^H)| \leq \sum_k \underbrace{\left| (\psi_h)_k^T \left(\mathcal{R}_h(u_h^H) \right)_k \right|}_{\eta_k}. \tag{2.27}$$

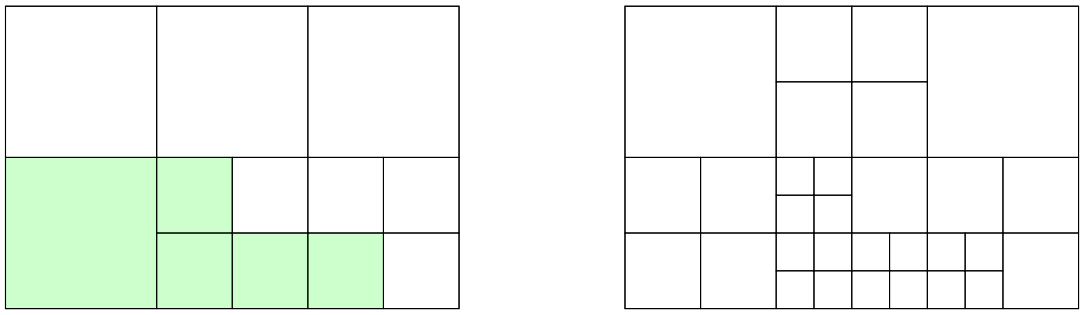
Therefore, targeting reductions locally for this cell-wise quantity will globally improve the goal-oriented output error bound. This will serve a key role in our adjoint-based adaptation method. For an adjoint-consistent method, both the solution and adjoint converge asymptotically at a rate of $\mathcal{O}(h^{p+1})$. Due to the presence of the divergence operator, the residual will convergence at $\mathcal{O}(h^p)$ [112]. Therefore, optimally the asymptotic convergence rate for both the error indicator and exact functional error is $\mathcal{O}(h^{2p+1})$.

2.5 Fixed-Fraction Refinement

One of the simplest and most commonly used approaches for mesh adaptation is based on a fixed-fraction refinement. In this case the mesh adaptation procedure begins by selecting some cell-wise error estimate which can be either feature-based (e.g. largest derivative values) or goal-oriented (such as the DWR). Then, after sorting the values of this quantity, the largest chosen fraction of cells are flagged. These cells are then isotropically split into sub-cells, 4 in the quad meshing case. As a result, a fraction of 30% is commonly used which provides an approximate doubling in the number of DOFs per iteration. Fig. 2.2 shows an example of this refinement on a patch of flagged cells. As will be used for the results, an additional split is introduced on the top center cell to prevent a difference of more than one level of refinement between neighbouring cells.

Note that this procedure introduces sub-faces to the cell interface, therefore, requiring a solver capable of handling non-conforming meshes. Fortunately, this is not an issue for the DG method due to the discontinuous nature of the cell boundaries. Otherwise, transition element templates can be used to ensure the mesh remains conformal between refinement areas for quad [2,114] and hex [55,100] meshes. Additionally, coarsening may also be used on cells with the lowest error, albeit, at a much lower rate. Other works in this area have considered the use of smoothness and jump indicators to incorporate decisions regarding anisotropic splitting [72] and hp adaptation [26].

This method is stable and generally performs well (in comparison to a uniform refinement). Therefore, it will be used in the results as a benchmark for the continuous methods presented herein.



(a) Initial mesh with flagged cells **(b)** Final mesh after splitting

Figure 2.2: Example of Fixed-fraction refinement from flagged cells (shown in green).

Chapter 3

Mesh Generation

This section will serve as a brief introduction to the fundamental mesh generation techniques and other essential mathematical concepts relevant to the remainder of our study. Here, we will start by discussing two of the most basic topics from computational geometry: the Delaunay triangulation, and its dual-graph the Voronoi diagram. We will then proceed with a discussion of the fields forming the basis for the continuous mesh models. Through their local modifications to the distance function, these serve an essential role in targeting anisotropic meshing behaviour. This will begin by looking at Riemannian metric fields used in the past with tri mesh generation and then progress to a discussion of frame fields which we will use in this work for the quad meshing case. In particular, we will examine the continuous mathematical tools these fields offer in order to model the discrete mesh behaviour.

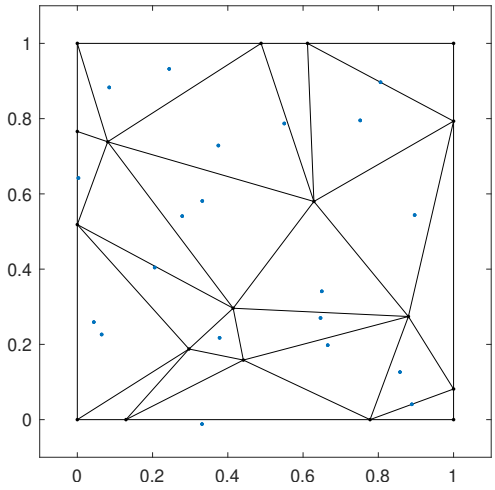
3.1 Delaunay Triangulation

The Delaunay Triangulation (DT) forms a key aspect of many basic mesh generation schemes. At its core, given a distribution of points $\boldsymbol{x}_i$ (the Delaunay nodes), it involves finding a triangulation of the domain such that no element contains exterior points in its circumcircle. This is known as the ”empty circumcircle property” as proposed by De-

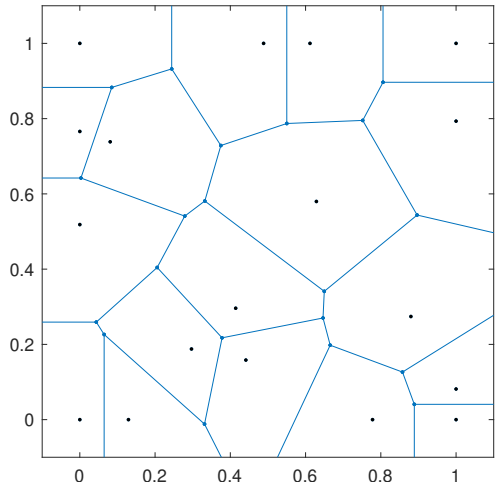
launay during his early work on the topic [30]. This also generalizes for use with nD spheres with simplices in higher dimensions [120]. There are several other properties resulting from this method of triangulation: the DT maximizes the minimum angles (in order) [120], it minimizes the circumcircle radius of each element [89], and it minimizes the interpolation error for an isotropic function $\|x\|_2^2$ among triangulations of the given vertices [27]. Additionally, the DT can be formed as the geometric dual of the Voronoi diagram [44, 107], which will be investigated in Section 3.2. Fig. 3.1a shows an example of the DT for a random set of points along with other constructs to be discussed.

Early computational works generated the DT starting from an empty domain by iteratively constructing valid triangles from a surface edge [91] or by inserting points in an ordered fashion (e.g. increasing x) and ensuring the result is Delaunay through a series of edge swaps [69]. Later, algorithms based on divide-and-conquer style approaches and splitting of the convex-hull improved the efficiency from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$ [71,115]. This mainly leaves the selection of node coordinates as the important step in generating a high-quality mesh. This is commonly achieved through node insertion either as a frontal approach [92] or by reconnection of a "cavity" by the Bowyer-Watson algorithm [18, 107, 131].

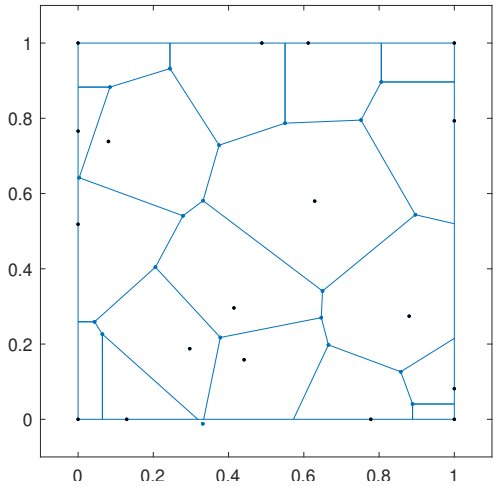
This has also been extended to Constrained Delaunay Triangulations (CDT) for meshing in the presence of prescribed boundary edges [5, 28]. In the case of a convex domain, the boundary edges will form the convex-hull and therefore no explicit step is needed. For anisotropic meshing, a modification of the circumcircle property to the case of a metric ellipse can be used. Here, methods have been developed by introduction of an anisotropic Delaunay kernel for both serial [33,46,86] and parallel [81,87] cases. Extensions have also been made to Optimal Delaunay Triangulations (ODT) based on a function dependent minimization of the interpolation error [27]. The ODT has also been considered for the case of curved mesh generation [42]. These concepts make the DT a crucial part of many triangular mesh generation and adaptation schemes.



(a) Delaunay Triangulation.



(b) Voronoi Diagram.



(c) Clipped Voronoi Diagram.

Figure 3.1: Comparison of various geometric constructs.

3.2 Voronoi Diagram

As described before, the Voronoi Diagram (VD) is the geometric dual of the DT, see Fig. 3.1b. Here, the circumcenter of each tri becomes a vertex of the VD and each edge results from the equidistant line of two Delaunay nodes, making it perpendicular to the corresponding Delaunay edge. The inverse statement is also true and the DT can be obtained by connecting edges across each face of the VD. Alternatively, the VD can be defined as a partitioning of the domain into regions Ω_i nearest to each particular Delaunay node $\boldsymbol{x}_i$ (sometimes called a site in this context) [54]. In this case, each area is repre-

sented by a polygon, with those nodes lying on the convex hull expanding infinitely. This representation can be efficiently computed using Fortune's sweep line algorithm of cost $\mathcal{O}(n \log n)$ [44]. However, for mesh generation, we are typically only concerned with a bounded domain Ω. As a result, this graph can be clipped to limit the area in each cell. For the convex case, this can be done directly by comparing each boundary cell with the corresponding edges of the domain. Methods have also been proposed for dealing with non-convex domains. For example in [132] for 2D, boundary cells are identified via a propagation queue. Then, the 2D domain is clipped by the Voronoi cell using the Sutherland-Hodgeman clipping algorithm [118]. Note here the direction of the application of the clipping is crucial as the clip lines must form convex domains (which the Voronoi cells are by definition). The remaining area on the domain after completion of the clipping results in the clipped cells on the boundary as shown in Fig. 3.1c. A procedure for clipping 3D surfaces based on a tessellated background mesh has also been studied [132]. This modification could be applied to the 2D case, however, it is not strictly necessary due to the significantly lower number of Voronoi cell faces compared to the 3D polytope.

As with the DT, the VD can also be used to target improvements in mesh quality through the dual space. The most common method is through the use of a Centroidal Voronoi Tessellation (CVT). This is a special class of VDs where the generating nodes or sites $\boldsymbol{x}_i$ coincide with the centroid of the cells $\boldsymbol{z}_i$ [36]. One of the simplest methods, known as Lloyd's algorithm, involves directly placing the site at the corresponding centroid on each iteration [36, 77]. From [36], the CVT can also be defined as an energy minimization problem with the functional

$$E_{CVT}(\boldsymbol{x}) = \sum_i \int_{\Omega_i \cap \Omega} \rho(\boldsymbol{y}) \, \|\boldsymbol{y} - \boldsymbol{x}_i\|_2^2 \, \mathrm{d}\boldsymbol{y}, \tag{3.1}$$

where Ω_i is the integral over corresponding Voronoi cells and $\rho(\boldsymbol{y})$ is the target density distribution. This expression can be differentiated [36] resulting in the gradient

$$\frac{\partial E_{CVT}}{\partial \boldsymbol{x}_i} = 2(\boldsymbol{x}_i - \boldsymbol{z}_i) \int_{\Omega_i \cap \Omega} \rho(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}. \tag{3.2}$$

The challenge shifts to solving for a stable critical points of this expression. It has also been shown that this functional is almost always C^2 continuous in the node positions [76], making it suitable for use with Newton solvers [95]. In particular, several methods have been able to use quasi-Newton methods such as the L-BFGS method [94] with success to effectively converge CVT problems [57,76]. This will form the basis of the L_p-CVT mesh generator to be introduced in Chapter 4.

3.3 Riemannian Metric Space

One of the most elegant methods for incorporating anisotropy into the mesh generation process is through the notion of the Riemannian metric. Here the anisotropy observed in the physical (or Euclidean) space is simply the result of an isotropic mesh generation procedure undergone in the Riemannian metric space [1,79]. We refer the reader to [12] for a more complete overview on the topic from the perspective of differential geometry. In the context of CFD mesh generation, early work incorporating the Riemannian metric field was done in [25,122]. This primarily involves a modification of the distance operator through the introduction of a generalized anisotropic 2 norm based on a globally varying metric field. Here we will give a brief overview of the constant field case. A more in depth approach requires integration along geodesics as described in [35,79,84] or can be approximated in an exponential space [85].

This metric space comes equipped with the inner-product and norm operators

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathcal{M}} = \sqrt{\boldsymbol{u}^T \mathcal{M} \boldsymbol{v}}, \tag{3.3}$$

$$\|\boldsymbol{u}\|_{\mathcal{M}} = \sqrt{\boldsymbol{u}^T \mathcal{M} \boldsymbol{u}}, \tag{3.4}$$

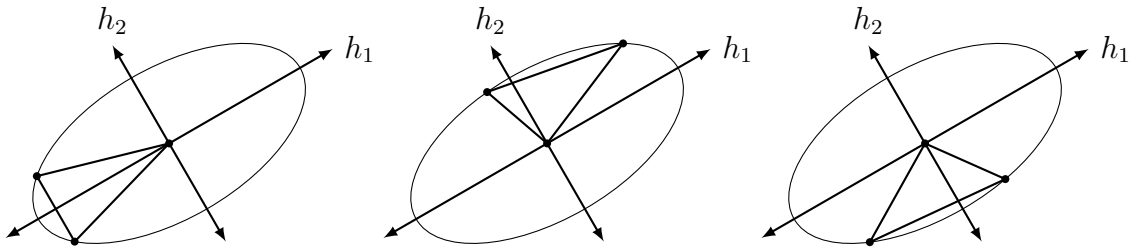


Figure 3.2: Equivalency class of unit-element triangles relative to metric $\mathcal{M}$.

where $\mathcal{M}$ is the local metric tensor. In $\mathbb{R}^d$ this can be expressed as a $d \times d$ symmetric positive definite matrix. As shown in Fig. 3.2, this defines the unit ball, $\mathcal{B}_{\mathcal{M}} = \{\|\boldsymbol{u}\|_{\mathcal{M}} = 1\}$ which takes the form of an ellipse often called the metric ellipse. In 2D the metric can be rewritten using an eigenvalue decomposition as

$$\mathcal{M} = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{bmatrix} = R(\theta) \underbrace{\begin{bmatrix} h_1^{-2} & 0 \\ 0 & h_2^{-2} \end{bmatrix}}_{\Lambda} R(\theta)^T, \tag{3.5}$$

where h_i represent the principle axes lengths and θ the orientation of the metric ellipse, as in Fig. 3.2. Based on this decomposition, we can rewrite the norm, Eq. 3.4, using $\mathcal{M}^{-\frac{1}{2}} = R(\theta)\Lambda^{-\frac{1}{2}}R(\theta)^T$ in terms of the 2 norm

$$\|\boldsymbol{u}\|_{\mathcal{M}} = \left\|\mathcal{M}^{-1/2}\boldsymbol{u}\right\|_2. \tag{3.6}$$

Targeting isotropic meshing in this norm results in the unit element definition

$$\|\boldsymbol{e}_i\|_{\mathcal{M}} = 1, \quad \forall \boldsymbol{e}_i \in \mathcal{T}_h, \tag{3.7}$$

where $\mathcal{T}_h$ is the triangulation and $\boldsymbol{e}_i$ corresponds to each edge of the local discrete triangular element. By considering the transformation $\tilde{\boldsymbol{e}}_i = \mathcal{M}^{-\frac{1}{2}}\boldsymbol{e}_i$ applied to rotated isotropic equilateral triangles, we see that this forms an equivalency class of unit mesh elements in the metric space (as shown in Fig 3.2). However, in general this expression can only be satisfied element-wise, not globally. Therefore, instead the unit mesh definition is relaxed

to a suitable (quasi-unit) range [79]. Globally, this can also be stated as an optimization problem [35]

$$\mathcal{T}_h = \operatorname{argmin} \sum_{\boldsymbol{e} \in \mathcal{T}_h} \left(\|\boldsymbol{e}\|_{\mathcal{M}} - C \right)^2, \tag{3.8}$$

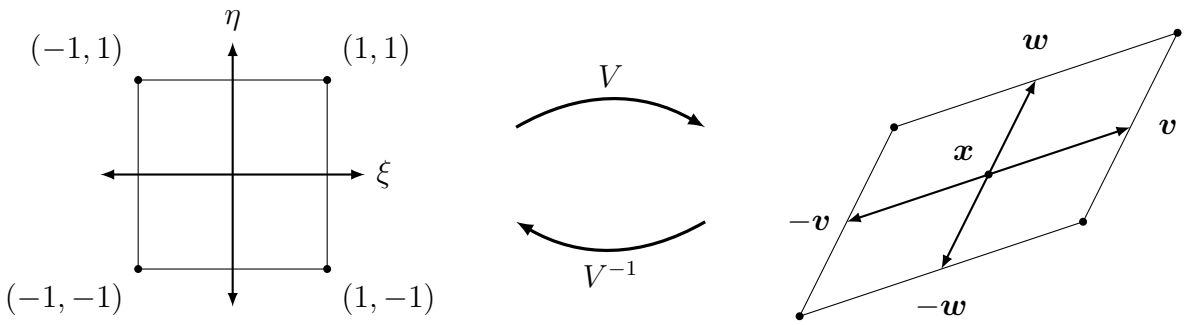
where C is a constant dependent on the choice of method. Fortunately there are already a number of mesh generators based on this concept readily available for prescribed metric meshing. One example making use of these concepts in 2D is BAMG [58], which has also been implemented in GMSH [51]. An adapted mesh is generated relative to a given background metric function (from an input file). This also performs metric smoothing to control variations in the mesh size near sharp features. In the results, this will be used along with Blossom-Quad [110] in an indirect method as a benchmark of our targeted all-quad mesh generator.

Overall, this establishes a duality between the continuous representation of the Riemannian metric field and the resultant discrete triangular mesh. This same idea will be exploited as a means of considering the error minimization problem in the well-posed continuous setting for the quad meshing case.

3.4 Frame Fields

Next, we will review the topic of frame fields. We will also draw parallels with the use of the Riemannian metric fields and preface their relation to the current work. For a more extensive introduction to the topic, see Panozzo et al. [99] who originally proposed the term. Alternatively, for a discussion of direction fields, which include cross fields, frame fields and other generalization, see also [123].

For a frame field $\mathcal{F}$, a frame $f_{\boldsymbol{x}} =< \boldsymbol{v}, \boldsymbol{w}, -\boldsymbol{v}, -\boldsymbol{w} >$ is defined at each point $\boldsymbol{x}$ on a surface in $\mathbb{R}^3$ by a coupled set of vector fields, $\boldsymbol{v}$ and $\boldsymbol{w}$, tangent to the plane [99]. Here, counterclockwise ordering is assumed and cyclic permutations of these axes represent the same frame. For the current work, we will limit our considerations to the planar 2D



(a) Reference Element **(b)** Representative Parallelogram

Figure 3.3: Reference element and frame field representative parallelogram.

meshing case. From an extension of parametrization based methods, these vectors can also be seen as the target reference element axes of a physical local quadrangle. Therefore, tracing along a smooth frame field provides the overall structure of the underlying quad mesh. Then, by parallel translation of the vector set, this can be used to form a representative parallelogram describing the target element for each point [15] as shown in Fig. 3.3.

From the representative parallelogram, we can define a unique linear mapping V from the reference element. Note that this is equivalent to the Jacobian matrix which varies spatially on general quad elements. This relation can be expressed as

$$f_{\boldsymbol{x}} =< \boldsymbol{v}, \boldsymbol{w}, -\boldsymbol{v}, -\boldsymbol{w} >= V < \boldsymbol{e}_1, \boldsymbol{e}_2, -\boldsymbol{e}_1, -\boldsymbol{e}_2 >, \tag{3.9}$$

where $\boldsymbol{e}_i$ are the unit vectors along the i^{th} coordinate axis forming the cross shown in Fig. 3.3a. One of the key advantages of this representation is that it allows for a polar decomposition of the frame. Here, we can rewrite the linear transformation using the polar form of each vector

$$V = \begin{bmatrix} \boldsymbol{v} & \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \cos(\theta_2) \\ \sin(\theta_1) & \sin(\theta_2) \end{bmatrix} \begin{bmatrix} h_1 & 0 \\ 0 & h_2 \end{bmatrix}, \tag{3.10}$$

where h_1, h_2 and θ_1, θ_2 are the lengths and angles associated with $\boldsymbol{v}$ and $\boldsymbol{w}$ respectively. Here, for the purpose of high-quality mesh generation, we will assume the orthogonal vector case. The non-orthogonal case has also been studied in [99] through decomposition into a Symmetric Positive Definite (SPD) matrix and a unit cross field. For this special case, using $\theta_1 = \theta$ and $\theta_2 = \theta + \frac{\pi}{2}$ for a counterclockwise ordering, this can be further simplified to

$$V = R(\theta) \operatorname{diag}(h_1, h_2), \tag{3.11}$$

where $R(\theta)$ is the counterclockwise rotation matrix. Alternatively, factoring out the average size $h = \sqrt{h_1 h_2} = \sqrt{\det(V)}$ and introducing the anisotropy $\rho = h_1/h_2$, we can rewrite the linear transformation in the decomposed form as

$$V = hR(\theta) \operatorname{diag}\left(\sqrt{\rho}, \frac{1}{\sqrt{\rho}}\right), \tag{3.12}$$

where in future sections we can replace the frame field definition with solving for h, θ and ρ directly. The inverse transformation for this case can also be expressed directly as

$$V^{-1} = \frac{1}{h} \operatorname{diag}\left(\frac{1}{\sqrt{\rho}}, \sqrt{\rho}\right) R(-\theta). \tag{3.13}$$

As we saw, the Riemannian metric field results in a modification of the 2 norm. Works in the past have introduced a similar idea based on L_∞ norm to produce right-angled triangles [109]. Therefore, for quad meshing, we consider the anisotropic modification of this norm based on the local frame field

$$\|\boldsymbol{u}\|_f = \left\|V^{-1}\boldsymbol{u}\right\|_\infty. \tag{3.14}$$

This definition serves two main purposes in the mesh generation process. First, this defines the unit ball $\mathcal{B}_f = \left\{\|\boldsymbol{u}\|_f = 1\right\}$ which corresponds exactly with the representative parallelogram of the frame. Unlike the case of triangular mesh generation based on the

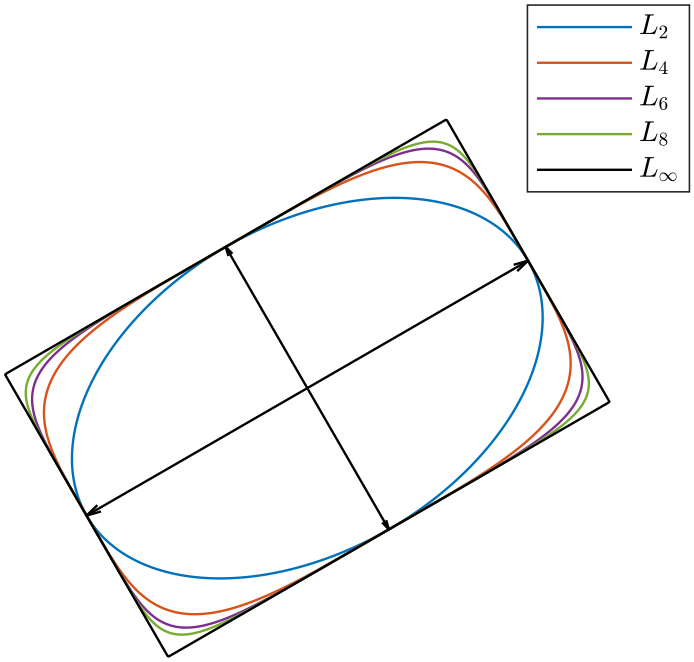


Figure 3.4: Comparison of unit ball of anisotropic L_p norm.

Riemannian metric where there was an equivalency class, this defines a unique target element for the mesh generator. This requires additional control over the orientation of the elements. Second, this matches the form used in the L_p-CVT energy by Lévy and Liu [74], which has been shown to generate aligned quad shaped Voronoi cells. This allows the method to produce right-angled triangles suitable for recombination.

However, as the L_∞ norm is non-differentiable, this behaviour is approximated using a sufficiently high L_p norm. From Fig. 3.4, it can be seen how the p value affects the shape of this unit ball. A value $p = 6$ was used a compromise for the L_p norm due to the diminishing returns beyond this point. In the case of $p = 2$ we return to a metric ellipse. Comparing Eq. 3.6 and 3.14, we notice a parallel between $\mathcal{M}^{-\frac{1}{2}}$ and V^{-1}. This will be exploited to form $\mathcal{M} = V^T V$ for use with BAMG as a benchmark for the L_p-CVT generator with use of our frame field estimates.

Chapter 4

L_p-CVT Mesh Generator

As described previously, there is a limited availability of unstructured quad mesh generators capable of meshing on a bounded domain. This issue is further amplified by the need for anisotropic remeshing based on the frame field concept, of which there are currently none available in the public domain.

As a result, a major portion of this work was dedicated to the development of an L_p-CVT mesh generator conforming to the discrete frame field targets produced by the error estimate defined in Chapter 5. Originally introduced by Lévy and Liu [74], this generalizes the Constrained Voronoi Tessellation (CVT) from Section 3.2 by introducing the L_p norm and anisotropic term. With a sufficiently high choice of p (here $p = 6$ was used), the isocontours of the norm approximate the representative parallelogram [74]. It has been observed that this leads to alignment of the nodes along the principal axis (or frame field) directions in stable critical points of the problem. Additionally, rather than using L_∞, this choice maintains the differentiability of the energy functional, allowing for the computation of analytic gradients to assist in the optimization process. Overall, the global energy minimization functional defined over the domain Ω can be expressed

$$E_{L_p}(\boldsymbol{x}) = \sum_i \int_{\Omega_i \cap \Omega} \|M(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{x}_i)\|_p^p \ \mathrm{d}\boldsymbol{y}, \tag{4.1}$$

where $\boldsymbol{x}_i$ are the node coordinates to be optimized and Ω_i are the corresponding Voronoi cells. Here we introduce the matrix function $M(\boldsymbol{y}) = (V(\boldsymbol{y}))^{-1}$ originating from the linear transformation of the local frame field for control of mesh anisotropy and orientation. This functional is minimized using a quasi-Newton method with an additional edge-splitting outer iteration for control of insertion/removal of nodes.

This work has been implemented as a standalone C++ code with dependencies on several existing libraries (GMSH, CGAL, Trilinos) [51,119,121]. While the code was designed to run on a single CPU, to improve computational speed, the numerical integration process described below was multi-threaded using OpenMP [96]. For consistency and convenience of access to Blossom-Quad [110], communication with external programs was performed via input and output in GMSH file format ".msh" files [51]. The remaining sections described the details of each step of the approach. This will be followed by an overview of the method implementation.

4.1 Background Metric Function

For our purposes, the metric function $M(\boldsymbol{y})$ is defined based on discrete estimates obtained from the flow solver in Chapter 5. As a result, a data structure must be established for querying values at the quadrature points from the background mesh. There are two main inputs to the mesh generator: the background quad-mesh $\mathcal{Q}_h$ used for computations in the previous adaptive iteration, and the discrete frame field defined element-wise $\mathcal{F}_h$. Note, during this procedure all size targets are doubled ($M(\boldsymbol{y})$ is scaled by 0.5) to allow for the splitting step discussed in Section 4.6. Here, CGAL [119] is used to establish an AABB tree for point location and frame field lookup. This is a constant cost lookup of $\mathcal{O}(\log n)$, however, due to the large number of evaluations this accounts for a significant portion of the overall computation time. Alternative methods for the standard CVT have proposed the use of traversal methods for point lookup with significant speed-ups [132]. Fortu-

nately, the number of full evaluations required can be greatly reduced by first checking if a point remains in the same background element between optimizer iterations.

4.2 Boundary Meshing

Next, parametrized boundaries of the domain are meshed by equidistribution of points in the frame field space. This leads to an energy minimization with fixed points on the boundary. For the standard CVT case other methods have also been studied, either including these parametrized points as part of the optimization or automatically placing boundary nodes after convergence [93]. From past methods using cross fields, elements at the boundary are assumed to align with the surface along one of their axes [67, 127]. In the case of the frame field, this would indicate integration along a single vector field, allowing the edge integral to equivalently take place in the L_2 norm. Therefore, we can evaluate the overall boundary edge length in this space as

$$L_M(\boldsymbol{\gamma}) = \int_0^1 \underbrace{\|M(\boldsymbol{\gamma}(t))\boldsymbol{\gamma}'(t)\|_2}_{dL(t)}\, dt, \tag{4.2}$$

where $\boldsymbol{\gamma}(t)$ is the equation for the curve in parametric space $t \in [0, 1]$ and $\boldsymbol{\gamma}'(t)$ is the corresponding tangent vector. Numerically, this is computed by the trapezoid rule

$$L_M(\boldsymbol{\gamma}) \approx \sum_{k=0}^{N-1} \underbrace{\frac{1}{2}\left(dL(t_{k+1}) + dl(t_k)\right)(t_{k+1} - t_k)}_{L_M^k}, \tag{4.3}$$

where $t_k = k/N, \forall k \in [0, N]$ are the $N+1$ equally distributed samples including endpoints and L_M^k are segment lengths. The parametric edge length is rounded up to determine the number of unit-length segments covering this distance in metric space $n = \lceil L_M(\gamma) \rceil$. To divide the boundary edge into these segments, this becomes a problem of determining points corresponding to the integer values of the incomplete integral

$$\boldsymbol{x}_i = \left\{ \boldsymbol{\gamma}(t_i) \;\middle|\; \frac{n}{L_M(\boldsymbol{\gamma})} \int_0^{t_i} \|M(\boldsymbol{\gamma}(t))\boldsymbol{\gamma}'(t)\|_2 \; dt = i \right\}, \quad \forall i \in [0, n]. \tag{4.4}$$

These points can be obtained by performing a partial sum until the element exceeding the next integer value

$$\boldsymbol{x}_i = \left\{ \boldsymbol{\gamma}(t_i) \;\middle|\; \sum_{k=0}^{j} dL_M^k + \frac{1}{2}\left(dL(t_i) + dL(t_j)\right)(t_i - t_j) = i, \quad i \leq \sum_{k=0}^{j+1} dL_M^k \right\}, \tag{4.5}$$

where, by linear interpolation

$$dL(t_i) = \left(dL(t_{j+1}) - dL(t_j)\right)\left(\frac{t_i - t_j}{t_{j+1} - t_j}\right) + dL(t_j) = at_i + b. \tag{4.6}$$

The problem reduces to a quadratic in t_i

$$at_i^2 + \left(b + dL(t_j) - at_j\right)t_i - \left(b + dL(t_j)\right)t_j - 2\left(i - \sum_{k=0}^{j} dL_M^k\right) = 0, \tag{4.7}$$

where the positive root $t_j < t_i \leq t_{j+1}$ is selected.

4.3 Clipped Voronoi Diagram

As described in Section 3.2, the Voronoi diagram is constructed using a sweep-line algorithm to determine the convex polygon nearest to each point in the L_2 norm. As used by Baudouin et al. [10], in order to limit the integration to the physical domain Ω, Voronoi cells intersecting the boundary are clipped. This introduces a new cell face along the domain boundary connecting two clip nodes. The clipped Voronoi diagram is then used to establish a facet-based data structure for use in the energy and derivative integrals. Each facet $f_k \in \Omega_i$ incorporates 1 Delaunay node $\boldsymbol{x}_i$ and 2 Voronoi vertices $\boldsymbol{c}_j$ (cell center or clip points) forming an ordered counterclockwise triangle. An example of this structure is shown below in Fig. 4.1.

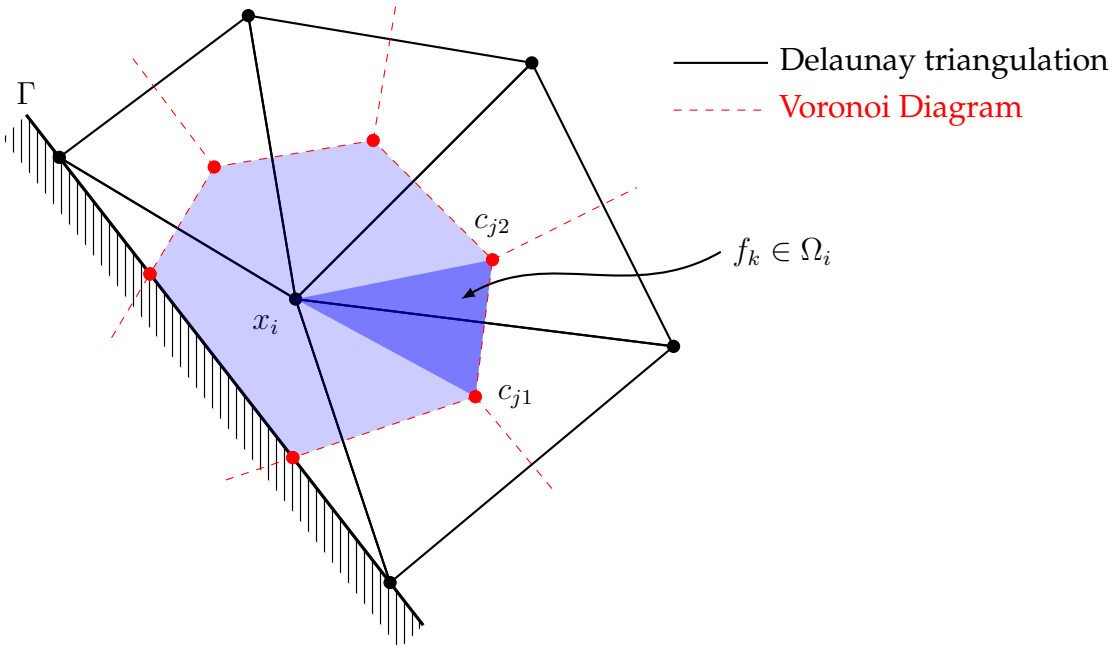


Figure 4.1: Triangles forming clipped Voronoi cell $f_k \in \Omega_i$.

4.4 Energy and Derivative Computation

The central and most computationally intensive step involves computing the integral of the energy function and its analytic derivatives. The energy associated with each volume node $\boldsymbol{x}_i$ (as boundary nodes are fixed) can be computed from the integral over its surrounding Voronoi cell, divided into facets $f_k \in \Omega_i$ from Fig. 4.1. Here, the integrals are obtained numerically using the quadrature rules on the reference triangle f' defined from [134]. On this element $\boldsymbol{T}(\boldsymbol{\xi})$ is used to define the mapping to the physical space and J is the corresponding Jacobian. The local energy value I_k from the facet f_k is

$$I_k = \int_{f_k} \|M(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{x}_i)\|_p^p \, \mathrm{d}\boldsymbol{y} = \int_{f'} \|M(\boldsymbol{T}(\boldsymbol{\xi}))(\boldsymbol{T}(\boldsymbol{\xi}) - \boldsymbol{x}_i)\|_p^p \, J \, \mathrm{d}\boldsymbol{\xi}, \tag{4.8}$$

where $E_{L_p} = \sum_i \sum_{f_k \in \Omega_i} I_k$ provides the global value. For simplicity, we denote the norm term as $\boldsymbol{F}(\boldsymbol{y}) = M(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{x})$. Next, computing the analytic derivative relative to the node coordinates using the facet-based data structure. The value can be separated into two terms. First, resulting directly from variations in $\boldsymbol{x}_i$ on its local cell changing the integrand $\boldsymbol{F}(\boldsymbol{y})$. Second, from the chain rule and the effect this has on the Voronoi vertex

coordinates which alter the integration facets f_k for both the current and neighbouring cells. Overall, assembly of the gradient can be written [10] as

$$\frac{\partial E_{L_p}}{\partial \boldsymbol{x}_i} = \sum_{f_k \in \Omega_i} \frac{\partial I_k}{\partial \boldsymbol{x}_i} + \sum_{\boldsymbol{c}_j \in \Omega_i} \sum_{f_k \ni \boldsymbol{c}_j} \frac{\partial I_k}{\partial \boldsymbol{c}_j} \frac{\partial \boldsymbol{c}_j}{\partial \boldsymbol{x}_i}. \tag{4.9}$$

For the Voronoi vertex derivatives $\frac{\partial \boldsymbol{c}_j}{\partial \boldsymbol{x}_i}$ the reader is advised to refer to [74] for the cell center and [10] for the clipped boundary vertex cases. This term is linked to the geometric construction of the clipped Voronoi diagram. The remaining terms are obtained from the partial derivatives of the integral in Eq. 4.8. These are computed facet-wise using Eqs. (8) and (9) from [40]

$$\frac{\partial I_k}{\partial \boldsymbol{x}_i} = \int_{f'} \frac{\partial \left\| \boldsymbol{F} \right\|_p^p}{\partial \boldsymbol{F}} \left[\frac{\partial M}{\partial \boldsymbol{T}} \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{x}_i} \left(\boldsymbol{T}(\boldsymbol{\xi}) - \boldsymbol{x}_i \right) + M(\boldsymbol{T}(\boldsymbol{\xi})) \left(\frac{\partial \boldsymbol{T}}{\partial \boldsymbol{x}_i} - 1 \right) \right] J + \left\| \boldsymbol{F} \right\|_p^p \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{x}_i} \, \mathrm{d}\boldsymbol{\xi}, \tag{4.10}$$

$$\frac{\partial I_k}{\partial \boldsymbol{c}_j} = \int_{f'} \frac{\partial \left\| \boldsymbol{F} \right\|_p^p}{\partial \boldsymbol{F}} \left[\frac{\partial M}{\partial \boldsymbol{T}} \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{c}_j} \left(\boldsymbol{T}(\boldsymbol{\xi}) - \boldsymbol{x}_i \right) + M(\boldsymbol{T}(\boldsymbol{\xi})) \left(\frac{\partial \boldsymbol{T}}{\partial \boldsymbol{c}_j} \right) \right] J + \left\| \boldsymbol{F} \right\|_p^p \frac{\partial J}{\partial \boldsymbol{c}_j} \, \mathrm{d}\boldsymbol{\xi}. \tag{4.11}$$

Together, these expressions form the analytic derivatives of the L_p-CVT energy functional. Using these energy gradients, the minimization problem $\min_{\boldsymbol{x}} E_{L_p}$ is solved with Trilinos ROL [121] using an L-BFGS quasi-Newton optimization with cubic line search. The strength of this line search is especially important in order to ensure suitable descent is achieved in the presence of discontinuities in the background metric. Convergence is reached once either the norm of the step size $\boldsymbol{\delta x}$ or the gradient ∇E_{L_p} fall below given tolerances. The assembly of gradient terms and the quasi-Newton iteration are summarized in Algorithm 1.

Algorithm 1: L_p-CVT energy minimization procedure

Input: Metric function $M(\boldsymbol{y})$, initial node positions $\boldsymbol{x}_0$
while $\|\boldsymbol{\delta x}\| > tol$ ***and*** $\|\nabla E\| > tol$ **do**
  Ω_i, f_k = Generate clipped Voronoi diagram, Section 4.3

  `// Looping over facets`
  foreach $f_k = [\boldsymbol{x}_i, \boldsymbol{c}_{j1}, \boldsymbol{c}_{j2}] \in \Omega$ **do**
    `// Cell energy`
    I_k = Eq. 4.8
    $E \mathrel{+}= I_k$

    `// Analytic derivatives`
    $\frac{\partial I_k}{\partial \boldsymbol{x}_i}$, $\frac{\partial I_k}{\partial \boldsymbol{c}_{j1}}$, and $\frac{\partial I_k}{\partial \boldsymbol{c}_{j2}}$ = Eqs. 4.10 and 4.11
    $\frac{\partial E}{\partial \boldsymbol{x}_i} \mathrel{+}= \frac{\partial I_k}{\partial \boldsymbol{x}_i}$

    `// Cross terms`
    foreach $f_m = [\boldsymbol{x}_m, \boldsymbol{c}_{n1}, \boldsymbol{c}_{n2}] \ni \boldsymbol{c}_j = [\boldsymbol{c}_{j1}, \boldsymbol{c}_{j2}]$ **do**
      $\frac{\partial E}{\partial \boldsymbol{x}_m} \mathrel{+}= \frac{\partial I_k}{\partial \boldsymbol{c}_j} \frac{\partial \boldsymbol{c}_j}{\partial \boldsymbol{x}_m}$ with $\frac{\partial \boldsymbol{c}_j}{\partial \boldsymbol{x}_m}$ from [10,74]
    end
  end

  $\boldsymbol{x} \mathrel{+}= \boldsymbol{\delta x}$, L-BFGS step from Trilinos ROL [121]
end
Output: Updated node positions $\boldsymbol{x}$

4.5 Node Insertion/Removal

However, as the energy minimization procedure discussed in the previous section only serves to equally distribute points across the domain in the frame field space. An additional control is needed to ensure the size of the elements match their target values by identifying regions for insertion and removal of nodes. Here, this is done through a check on the individual edge lengths after convergence is achieved as used by [40]. The length of each edge of the mesh are measured from the L_p line integral

$$L_M^p(\boldsymbol{e}) = \int_0^1 \|M(\boldsymbol{\gamma}(t))\boldsymbol{\gamma}'(t)\|_p \, \mathrm{d}t, \tag{4.12}$$

where the edge is a straight line $\boldsymbol{\gamma}(t) = (\boldsymbol{x}_2 - \boldsymbol{x}_1)t + \boldsymbol{x}_1$ and $\boldsymbol{\gamma}'(t) = \boldsymbol{x}_2 - \boldsymbol{x}_1$. Then, these lengths are sorted and compared against a prescribed range $L_M^p(\boldsymbol{e}) \in [L_{min}, L_{max}]$. Edges above the maximum length are split by inserting a new node at the midpoint. Those falling below the minimum length are merged to form a single point at the average coordinate (midpoint). During this process, nodes are flagged after use and their remaining edges ignored in an attempt to limit excessive local changes. The edge-splitting procedure is summarized in Algorithm 2.

Algorithm 2: L_p-CVT edge splitting procedure

Input: Metric function $M(\boldsymbol{y})$, node positions $\boldsymbol{x}$, connectivity $\mathcal{T}_h$
begin
  `// Looping over face-connectivity graph`
  foreach $\boldsymbol{e}_i \in \mathcal{T}_h$ **do**
    $L_M^p(\boldsymbol{e}_i)$ = Eq. 4.12
  end

  `// Looping over sorted edge list`
  Sort $L_M^p(\boldsymbol{e})$, $\boldsymbol{f}$ = `false`
  foreach $\boldsymbol{e}_i = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}] \in L_M^p(\boldsymbol{e})$ **do**
    if $L_M^p(\boldsymbol{e}_i) > L_{max}$ ***and*** $!f_{i1}$ ***and*** $!f_{i2}$ **then**
      Add $\boldsymbol{x}_n = \frac{1}{2}(\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2})$
      $[f_{i1}, f_{i2}]$ = `true`
      n_{split} ++
    else if $L_M^p(\boldsymbol{e}_i) < L_{min}$ ***and*** $!f_{i1}$ ***and*** $!f_{i2}$ **then**
      Add $\boldsymbol{x}_n = \frac{1}{2}(\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2})$
      Delete $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$
      $[f_{i1}, f_{i2}]$ = `true`
      n_{merge} ++
    end
  end
end
Output: Updated node list $\boldsymbol{x}$ and number of edges merged/split $[n_{merge}, n_{split}]$

This process was repeated as an outer iteration for the L-BFGS until the number of edges split and merged fell below a suitable tolerance on 2 consecutive iterations. Commonly, we used a value of 2% of the total number of edges for the limit. However, this was ramped to a maximum of 5% if overall convergence was still not achieved after many

steps (typically due to oscillatory split/merge patterns). For the edge length range, a conservative $L_M^p(\boldsymbol{e}) \in [0.25, 2.0]$ was used. As we started from relatively few nodes for each run, this mainly focused on splitting of long edges to result in approximately unit length segments. Merging was applied only as a last resort to prevent over-grouping of points.

4.6 All-Quad Output

Finally, CGAL [119] is used to generate a constrained Delaunay triangulation of the points and boundary edges. As mentioned before, with sufficiently high choice of p norm we expect this minimization to produce right-angled triangles. As a result, they are suitable to be merged into quads using the Blossom-Quad algorithm through GMSH [51, 110]. This incorporates the use of the minimum-weight perfect-matching Blossom algorithm originally developed by Edmonds [38] with a quality based cost function to identify neighbors for merging. However, a requirement for compatibility with the flow solver is that the output mesh is all-quad. In the general case, the mesh can only be guaranteed to be quad-dominant at this stage, therefore, an additional subdivision step is required. To complete the algorithm, a light angle-based smoothing step is applied to the final all-quad mesh to help mitigate any mesh quality issues [135].

4.7 Communication

Overall, as this will form part of an adaptive loop with our flow solver, communication between the two programs is required. As an input, both a background mesh and frame field are needed. For this purpose, the GMSH ".msh" v4.1 file format was used [51]. This allowed us to pass the frame field written in the form $M(\boldsymbol{y}) = (V(\boldsymbol{y}))^{-1}$ as matrix valued data stored in the "$ElementData" field for each corresponding quad. See GMSH reference manual for details of formatting [50]. Once meshing was completed, the output mesh used the same format. However, here the matrix value data was no longer

needed but boundary information tags were used to define different boundary conditions or identification of the target surface for functional output evaluation.

4.8 Overview

Overall, the various stages of the program are summarized in Algorithm 3 and in the flowchart, Fig. 4.2.

Algorithm 3: L_p-CVT Mesh Generation Procedure

Input: Background quad mesh $\mathcal{Q}_n$, discrete frame field $\mathcal{F}_h$
begin
`// Pre-processing`
$M(\boldsymbol{y}) = (V(\boldsymbol{y}))^{-1}$, define metric from Eq. 3.13 and Section 4.1
$\overline{\mathcal{T}}$ = mesh domain boundaries, Eq. 4.4 in Section 4.2
$\boldsymbol{x}_0$ = distribute volume nodes

`// Splitting iterations, Section 4.5`
while $i_{split} < i_{max}$ ***and*** $i_{good} < 2$ **do**
`// Quasi-Newton iterations, Section 4.4`
$\boldsymbol{x}$ = Call Algorithm 1

`// Edge length checks, Section 4.5`
$[\boldsymbol{x}, n_{merge}, n_{split}]$ = Call Algorithm 2
if $n_{merge} < n_{tol}$ ***and*** $n_{split} < n_{tol}$ **then**
i_{good} ++
else
$i_{good} = 0$
end
end

`// Post-processing, Section 4.6`
$\mathcal{T}_{n+1}$ = generate CDT from CGAL [119]
$\mathcal{Q}/\mathcal{T}_{n+1}$ = merge to quad-dominant from Blossom-Quad [110]
$\mathcal{Q}_{n+1}$ = split to all-quad
θ = smooth cell angles based on [135]
end
Output: Adapted quad mesh $\mathcal{Q}_{n+1}$

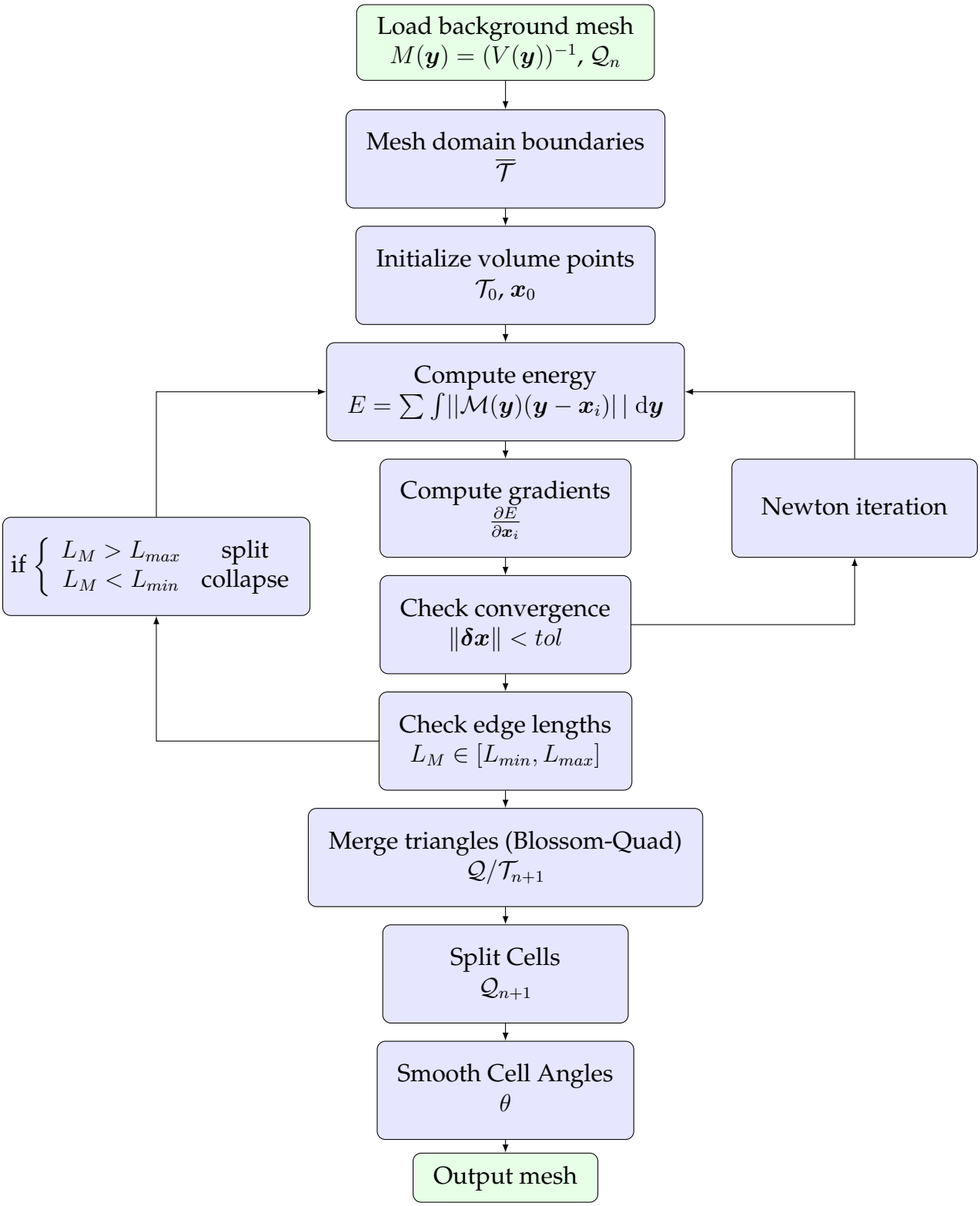


Figure 4.2: Flowchart outlining steps implemented in the L_p-CVT mesh generator.

Chapter 5

Continuous Mesh Adaptation

As Problem 1 cannot be solved directly, in this chapter it will be approximated by a continuous error model and a continuous mesh model. Extending on past works for high-order anisotropic error estimation in the triangular case, these models will serve to replace the objective function and constraint respectively.

First, we will examine the continuous mesh modelling the behaviour of the discrete hp mesh $\mathcal{Q}_{hp}$. For generality, this will involve a frame field $\mathcal{F}$ and a polynomial field $\mathcal{P}$ defined over the domain Ω. However, in our case we will assume a uniform polynomial distribution throughout for high-order h only adaptation. Second, the continuous error model will be divided into two parts. Locally the optimal shape (orientation and anisotropy) of the frame field will be determined, leaving the size or density to be solved as a global problem. Here, we will first consider the case of feature-based adaptation derived from a global error bound minimization using calculus of variations. Next, making use of the dual-weighted residual, we will introduce extensions to the goal-oriented adaptation. Finally, we provide a summary and brief overview of the complete adaptive iteration procedure.

5.1 Continuous Mesh Model

Here, based on Section 3.4, the frame field describes the target element size and shape at each point via the representative parallelogram. Based on Eq. 3.12, in the orthogonal case we can decompose the frame by the size h, orientation θ and anisotropy ρ of the local element. Using, this, we can define the element density throughout the domain

$$d(\boldsymbol{x}) = \frac{1}{4} \det\left(V(\boldsymbol{x})\right)^{-1}, \tag{5.1}$$

where $V(\boldsymbol{x})$ is the linear transformation associated with the frame $f_{\boldsymbol{x}}$ from Eq. 3.9. For our purposes, this could also be rewritten using $\det\left(V(\boldsymbol{x})\right)^{-1} = h(\boldsymbol{x})^{-2}$. Integrating this quantity over the domain, we can estimate the number of elements in the continuous mesh

$$\mathcal{N}_h(\mathcal{F}) = \int_\Omega d(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \frac{1}{4} \int_\Omega \det(V(\boldsymbol{x}))^{-1} \, \mathrm{d}\boldsymbol{x}. \tag{5.2}$$

Additionally, the degrees of freedom per element is known from Eq. 2.5 (based on the number of local shape functions). This can be written in the continuous sense based on the polynomial field at a given point $\boldsymbol{x} \in \Omega$

$$w(\boldsymbol{x}) = \left(\mathcal{P}(\boldsymbol{x}) + 1\right)^2, \tag{5.3}$$

for the 2D tensor-product element case. Integrating Eq. 5.1 over the domain again, this time weighted by the number of local shape functions from Eq. 5.3, we can approximate the total degrees of freedom for the continuous hp mesh

$$\mathcal{N}_{hp}(\mathcal{F}, \mathcal{P}) = \int_\Omega d(\boldsymbol{x}) w(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \frac{1}{4} \int_\Omega \det(V(\boldsymbol{x}))^{-1} \left(\mathcal{P}(\boldsymbol{x}) + 1\right)^2 \, \mathrm{d}\boldsymbol{x}, \tag{5.4}$$

which is often also called the continuous complexity. This provides a suitable substitute for the constraint of Problem 1 by $N_{hp}(\mathcal{Q}_{hp}) \approx \mathcal{N}_{hp}(\mathcal{F}, \mathcal{P}) \leq \mathcal{C}$ for some chosen complexity bound.

5.2 Local Continuous Error Model

The continuous error model used is based on previous works for the triangular mesh adaptation case [34, 35, 105]. Here, this results in a frame definition split into two parts from Eq. 3.12. First, in this section, we will introduce the continuous interpolation error. This will serve as a replacement for the integrand of the discrete error in Eq. 2.11 determined by the local solution behaviour and frame field. Additionally, this local error can then be bounded and minimized relative to an optimal choice of frame orientation θ and anisotropy ρ. In the next section, the global error will be minimized relative to this bound to determine the size distribution h of the frame field.

The error in the polynomial solution u of order p will be dominated by the $p+1$ order terms. Writing this as a Taylor series expansion about the point $\bar{\boldsymbol{x}}$

$$u_{\bar{\boldsymbol{x}},p}(\boldsymbol{x}) = \sum_{|\boldsymbol{\alpha}|\leq p} \frac{\partial^{\boldsymbol{\alpha}} u(\bar{\boldsymbol{x}})}{\boldsymbol{\alpha}!}(\boldsymbol{x}-\bar{\boldsymbol{x}})^{\boldsymbol{\alpha}} \quad (+\mathcal{O}(h^{p+1})), \tag{5.5}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$, $|\boldsymbol{\alpha}| = \alpha_1 + \alpha_2$ is the multi index of monomial exponents. Taking the difference with the $p+1$ expansion, this results in a homogeneous error polynomial

$$e^{int}_{\bar{\boldsymbol{x}},p}(\boldsymbol{x}) = u_{\bar{\boldsymbol{x}},p+1}(\boldsymbol{x}) - u_{\bar{\boldsymbol{x}},p}(\boldsymbol{x}) = \sum_{i=0}^{p+1} \frac{1}{i!(p+1-i)!} \frac{\partial^{p+1} u(\bar{\boldsymbol{x}})}{\partial x^i \partial y^{p+1-i}} (x-\bar{x})^i (y-\bar{y})^{p+1-i}. \tag{5.6}$$

This approximates the local discretization error $e^{int}_{\bar{\boldsymbol{x}},p}(\boldsymbol{x}) \approx (u - u_h)$. Alternatively, this can be written in terms of the directional derivatives with the unit vector $\boldsymbol{\xi} = \frac{\boldsymbol{x}-\bar{\boldsymbol{x}}}{|\boldsymbol{x}-\bar{\boldsymbol{x}}|}$

$$e^{int}_{\bar{\boldsymbol{x}},p}(\bar{\boldsymbol{x}} + h\boldsymbol{\xi}) = D^{p+1}_{\boldsymbol{\xi}} u(\bar{\boldsymbol{x}}) h^{p+1}, \quad \|\boldsymbol{\xi}\|_2 = 1. \tag{5.7}$$

In order to obtain these high order terms, the enriched solution $\tilde{u} \in \mathbb{P}^{p+1}(D(k))$ is approximated through the H^1 patchwise reconstruction on the set of nearest neighbours (shown in Fig. 5.1)

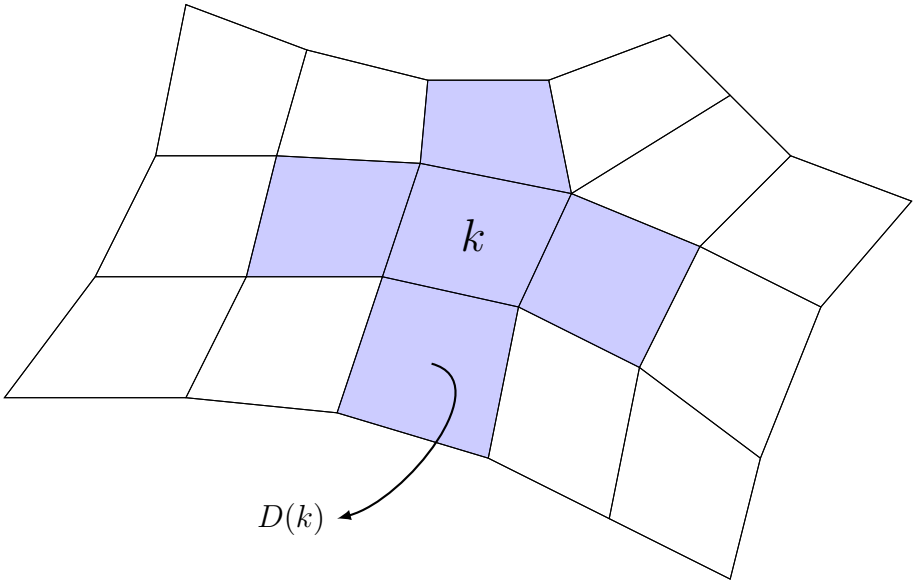


Figure 5.1: Patch of nearest neighbours $D(k)$ of element k for use in reconstruction.

$$\langle \tilde{u}, \phi \rangle_{H^1(D(k))} = \langle u_h, \phi \rangle_{H^1(D(k))}, \quad \forall \phi \in \mathbb{P}^{p+1}(D(k)). \tag{5.8}$$

From [35], Eq. 5.7 can be approximately bounded by the high-order quadratic form

$$|e^{int}_{\bar{\boldsymbol{x}},p}|(\boldsymbol{x}) \leq \left((\boldsymbol{x} - \bar{\boldsymbol{x}})^T R(\varphi) \operatorname{diag}(A_1, A_2)^{\frac{2}{p+1}} R(-\varphi)(\boldsymbol{x} - \bar{\boldsymbol{x}}) \right)^{\frac{p+1}{2}}, \tag{5.9}$$

where $R(\theta)$ is the counterclockwise rotation matrix. From the enriched solution we denote the largest order $p+1$ directional derivative A_1 and its direction $\boldsymbol{\xi}_1$. The directional derivative in the perpendicular direction A_2 is also used. These can be summarized

$$A_1(\bar{\boldsymbol{x}}, p) = \max_{\|\boldsymbol{\xi}\|_2 = 1} |D^p_{\boldsymbol{\xi}} u(\bar{\boldsymbol{x}})|, \tag{5.10}$$

$$\boldsymbol{\xi}_1(\bar{\boldsymbol{x}}, p) = \operatorname*{argmax}_{\|\boldsymbol{\xi}\|_2 = 1} |D^p_{\boldsymbol{\xi}} u(\bar{\boldsymbol{x}})|, \tag{5.11}$$

$$\varphi(\bar{\boldsymbol{x}}, p) \in [0, 2\pi) \quad s.t.\ \boldsymbol{\xi}_1 = (cos(\varphi), sin(\varphi)), \tag{5.12}$$

$$A_2(\bar{\boldsymbol{x}}, p) = |D^p_{\boldsymbol{\xi}_2} u(\bar{\boldsymbol{x}})|, \quad \text{where } \boldsymbol{\xi}_1 \cdot \boldsymbol{\xi}_2 = 0. \tag{5.13}$$

Note that the error bound from Eq. 5.9 is approximate as the perpendicular direction value is not representative of the high-order error level sets. This can be modified as discussed in [34] to ensure the bound is sharp. However, similar performance is achieved from these bounds in practice.

With the choice of quadratic form, the level-sets of the error are approximated by concentric ellipses. Therefore, from the metric ellipse derived in [34], this can be transformed into an isotropic error function minimizing the bound on the unit-ball. Thus, we obtain the anisotropy $\rho = (A_1/A_2)^{\frac{-1}{p+1}}$ and orientation $\theta = \varphi - \frac{\pi}{2}$ for the frame field. This result also bounds the cell-wise interpolation error with an added constant to ensure the metric ellipse circumscribes the representative parallelogram. Treating this as a piecewise constant function over the domain, we introduce the global continuous error estimate

$$\mathcal{E}(d, \mathcal{P}) = \sum_{k \in \mathcal{Q}_h} \int_{\Omega_k} e(\bar{\boldsymbol{x}}_k, d_k, p_k) \, \mathrm{d}\boldsymbol{x} = \sum_{k \in \mathcal{Q}_h} \left\| e^{int}_{\bar{\boldsymbol{x}},p} \right\|^q_{L^q(\Sigma)} \geq \sum_{k \in \mathcal{Q}_h} \left\| e^{int}_{\bar{\boldsymbol{x}},p} \right\|^q_{L^q(k)} \approx E_h, \tag{5.14}$$

which also provides an upper bound for the discrete error from Problem 1. Based on this, the local continuous error estimate is given by

$$e(\boldsymbol{x}, d, p) = B(\boldsymbol{x}, p) d(\boldsymbol{x})^{-\frac{q(p+1)}{2}}, \tag{5.15}$$

where $B(\boldsymbol{x}, p)$ is the local constant determined by the solution behaviour and cell shape, independent of the element density

$$B(\boldsymbol{x}, p) = 2^{\frac{q(p+1)+2}{2}} \left(\frac{2\pi}{q(p+1)+2} \right) \left(A_1(\boldsymbol{x}, p) A_2(\boldsymbol{x}, p) \right)^{\frac{q}{2}}. \tag{5.16}$$

5.3 Global Continuous Error Minimization

After solving the local problem from Eq. 5.15, the remaining continuous global error function $\mathcal{E}$ depends only on the choice of element density $d(\boldsymbol{x})$ and polynomial distribution

$\mathcal{P}(\boldsymbol{x})$. Assuming a constant but arbitrary order polynomial distribution, we can introduce the global continuous error minimization problem:

Problem 2 *Given a polynomial distribution $\mathcal{P}(\boldsymbol{x})$ and hp mesh complexity limit $\mathcal{C}$, find the density distribution solving*

$$\begin{aligned} \min_{d} \quad & \mathcal{E}(d,\mathcal{P}) = \int_{\Omega} B(\boldsymbol{x},\mathcal{P}(\boldsymbol{x}))d(\boldsymbol{x})^{-\frac{q(\mathcal{P}(\boldsymbol{x})+1)}{2}}\,\mathrm{d}\boldsymbol{x}, \\ \textit{s.t.} \quad & \mathcal{N}_{hp}(d,\mathcal{P}) = \int_{\Omega} d(\boldsymbol{x})(\mathcal{P}(\boldsymbol{x}))+1)^2\,\mathrm{d}\boldsymbol{x} \leq \mathcal{C}. \end{aligned} \tag{5.17}$$

Through the substitution of both the objective and constraint functions we obtain a suitable approximation to Problem 1 in the continuous framework. Here, these are offered through the analogous continuous error and continuous mesh models respectively. Overall, these provide integral forms for the expressions and access to mathematical optimization tools such as calculus of variations which can be used to solve this problem. First, taking variations in objective and constraint functionals with respect to $d(\boldsymbol{x})$

$$\delta_d \mathcal{E}(d,\mathcal{P}) = -\int_{\Omega} \frac{q(\mathcal{P}(\boldsymbol{x})+1)}{2} B(\boldsymbol{x},\mathcal{P}(\boldsymbol{x}))d(\boldsymbol{x})^{\frac{-q(\mathcal{P}(\boldsymbol{x}+1)+2)}{2}} \delta d(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \tag{5.18}$$

$$\delta_d \mathcal{N}_{hp}(d,\mathcal{P}) = \int_{\Omega} (\mathcal{P}(\boldsymbol{x})+1)^2 \delta d(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{5.19}$$

Introducing the Lagrangian $\mathcal{L}(d,\mathcal{P}) = \mathcal{E}(d,\mathcal{P}) + \lambda\mathcal{N}_{hp}(d,\mathcal{P})$. With a fixed choice of $\mathcal{P}$ these variations must equal 0 to obtain a minima

$$\begin{aligned} \delta_d \mathcal{L}(d,\mathcal{P}) = 0 &= \delta_d \mathcal{E}(d,\mathcal{P}) + \lambda \delta_d \mathcal{N}_{hp}(d,\mathcal{P}), \\ &= \int_{\Omega} \left[\lambda(\mathcal{P}(\boldsymbol{x})+1)^2 - \frac{q(\mathcal{P}(\boldsymbol{x})+1)}{2} B(\boldsymbol{x},\mathcal{P}(\boldsymbol{x}))d(\boldsymbol{x})^{\frac{-q(\mathcal{P}(\boldsymbol{x}+1)+2)}{2}} \right] \delta d(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \end{aligned} \tag{5.20}$$

Due to the arbitrariness of the variations in $d(\boldsymbol{x})$, the integrand term in square brackets must be 0 for a valid solution. As a result

$$d(\boldsymbol{x}) = \left(\frac{2\lambda(\mathcal{P}(\boldsymbol{x})+1)}{qB(\boldsymbol{x},\mathcal{P}(\boldsymbol{x}))} \right)^{\frac{2}{-q(\mathcal{P}(\boldsymbol{x})+1)+2}}, \qquad \forall \boldsymbol{x} \in \Omega. \tag{5.21}$$

which provides the optimal distribution of DOFs dependent on the Lagrange multiplier λ. Problem 2 can then be solved via Bisection conducted on λ until the constraint is sufficiently satisfied.

Overall, this defines the size field and completes the global definition of the target frame field. These targets are then provided to the L_p-CVT mesh generator in order to produce a conforming adapted mesh (see Chapter 4). Therefore, through the duality of the continuous framework, we have targeted direct error minimization for the discrete meshing problem (Problem 1).

5.4 Goal-Oriented Adaptation

Alternatively, as discussed in Section 2.3, many simulations are performed with the primary goal being to measure some functional of interest. Therefore, we consider an alternative approach for goal-oriented adaptation targeting the error associated with this functional computation. This is achieved through the dual-weighted residual (DWR) η_k introduced in Section 2.4. Based on the implementation for tris by Balan et al. [6], this involves determining a cell-wise area scaling factor α_k such that

$$I_k = \alpha_k I_k^c, \tag{5.22}$$

where I_k^c is the current cell area measured from the mesh and I_k is the new target area. Based on the frame field definition, the new frame size will be

$$h = \sqrt{\frac{1}{4}\alpha_k I_k^c}. \tag{5.23}$$

This shifts the problem to determining a suitable distribution of α_k to highlight refinements in areas with large DWR values and coarsen excessively refined areas where the DWR is small to free up DOFs. From [6], this is achieved through a quadratic fit of α_k to the DWR in the logarithmic space between $\eta_{min} = \min_{k\in\Omega_k} \eta_k$ and $\eta_{max} = \max_{k\in\Omega_k} \eta_k$

$$\alpha_k = \begin{cases} \left((r_{max}-1)\,\xi_k^2+1\right)^{-1}, & \eta_k \geq \eta_{ref}, \\ \left((c_{max}-1)\,\xi_k^2+1\right), & \eta_k < \eta_{ref}, \end{cases} \tag{5.24}$$

where

$$\xi_k = \begin{cases} \frac{\log(\eta_k)-\log\left(\eta_{ref}\right)}{\log(\eta_{max})-\log\left(\eta_{ref}\right)}, & \eta_k \geq \eta_{ref}, \\ \frac{\log(\eta_k)-\log\left(\eta_{ref}\right)}{\log(\eta_{min})-\log\left(\eta_{ref}\right)}, & \eta_k < \eta_{ref}. \end{cases} \tag{5.25}$$

Here, there are 3 control parameters to choose: the maximum refinement factor r_{max}, the maximum coarsening factor c_{max} and a reference DWR value η_{ref}. With suitable refinement and coarsening ranges selected, bisection is performed on the η_{ref} value to control the complexity growth. Overall, this produces an updated set of size field targets for the frame definition. Once again, the local minimization based on the primal solution from Section 5.2 will be used to determine the cell orientation and anisotropy.

Note that unlike in the feature-based adaptation of Section 5.3, this procedure seeks an iterative improvement in the functional estimate and not a globally optimal distribution as derived from Problem 2. As a result, there is a significant dependence on the sequence of previous adaptation steps. In the case of goal-oriented adaptation, this was used to prevent oscillations between refinement and coarsening of some key regions of the flow. Here, particular care must be taken to ensure the chosen values of r_{max}, c_{max} and $\mathcal{C}$ are well suited for the previous mesh. For the examples shown, we found the choice $[r_{max}, c_{max}] = [20, 4]$ to give good results throughout. These values were also commonly used in the examples of [6], however, with a different procedure for selecting η_{ref} based on the fraction of cells to be refined (without direct control of the subsequent total DOFs). Additionally, care must be taken in the mesh generator to avoid "creep" in the resultant complexity relative to the target value as this could lead to requiring a net coarsening on the following step. In extreme cases, the target complexity may fall outside of the range of maximum refinement/coarsening ($\eta_{ref} = \eta_{min}$ or $\eta_{ref} = \eta_{max}$ respectively) preventing the continuation of the adaptation procedure.

5.5 Overview

The following flowchart, Fig. 5.2, and Algorithm 4, provide a summary of the feature-based and goal-oriented error estimation and mesh adaptation steps. This outline shows the main components that have been implemented in the flow solver code (PHiLiP) to be able to use the unstructured adaptation techniques discussed here. Note that each adaptive iteration involves a call to either the L_p-CVT mesh generator (discussed in Chapter 4) or to GMSH for use of BAMG with Blossom-Quad [51, 110] (with metric term discussed in Section 3.4) in order to generate a new quadrangulation.

Algorithm 4: Continuous error estimation and mesh adaptation process

Input: Initial Mesh $\mathcal{Q}_0$, initial solution u_0
while $N_{hp} < N_{max}$ **do**
  `// Primal and dual solve`
  u_h = solve flow, Eq. 2.9
  ψ_h = solve adjoint, Eq. 2.18 (goal-oriented only)

  `// Local error minimization`
  for $k \in \mathcal{Q}_h$ **do**
    $\tilde{u}_h$ = reconstruct $p+1$ solution, Eq. 5.8
    A_1, A_2, φ = determine directional derivatives, Eqs. 5.10-5.13
    $\rho = (A_1/A_2)^{\frac{-1}{p+1}}$, target anisotropy
    $\theta = \varphi - \frac{\pi}{2}$, target orientation

    η_k = DWR, Eq. 2.27 (goal-oriented only)
  end

  `// Global error minimization`
  Find λ (feature-based) or η_{ref} (goal-oriented) by bisection for $\mathcal{C} - \mathcal{N}_{hp} = 0$
  $d(\boldsymbol{x})$ = set cell sizes, Eq. 5.21 (feature-based) or Eq. 5.23 (goal-oriented)

  `// Adapt mesh`
  $V(\boldsymbol{x})^{-1}$ = assemble target frame field, Eq. 3.13
  $\mathcal{Q}_{n+1}$ = call L_p-CVT (Algorithm 3) or BAMG [58]
end

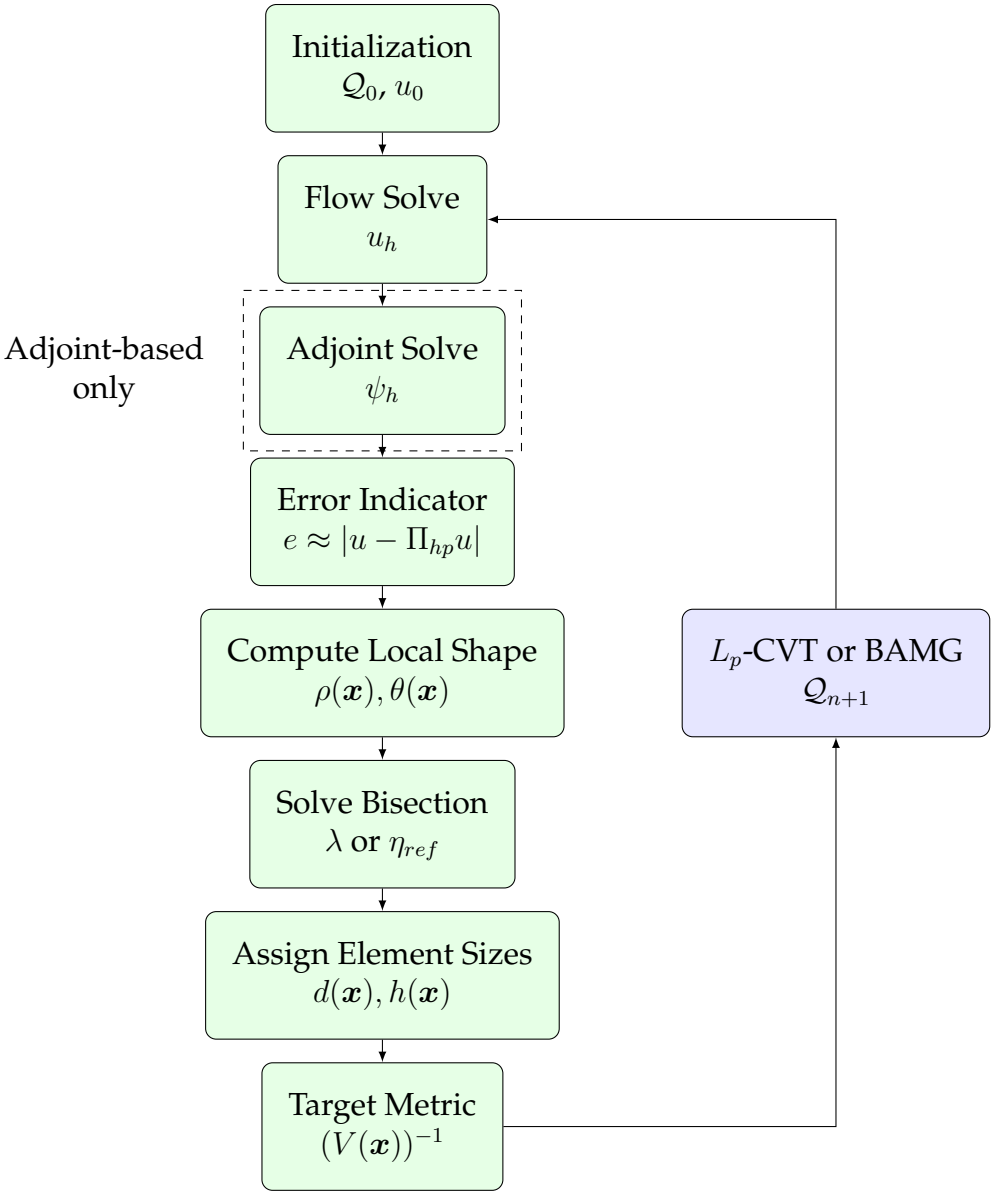


Figure 5.2: Flowchart outlining steps implemented in the PHiLiP flow solver.

Chapter 6

Results

In the following section, we will examine a selection of test cases. First, this will begin with a validation of the L_p-CVT mesh generator. We will also examine the impact of the discrete background mesh on convergence and final all-quad output mesh. After, we will consider two feature-based and two goal-oriented adaptive iterations. These serve to study the behaviour of the overall procedure on solutions modelling common flow behaviours (shocks and boundary layers). Overall, this will provide an initial showcase of the potential for the method and help identify future areas for development.

6.1 Validation of All-Quad L_p-CVT Mesh Generator

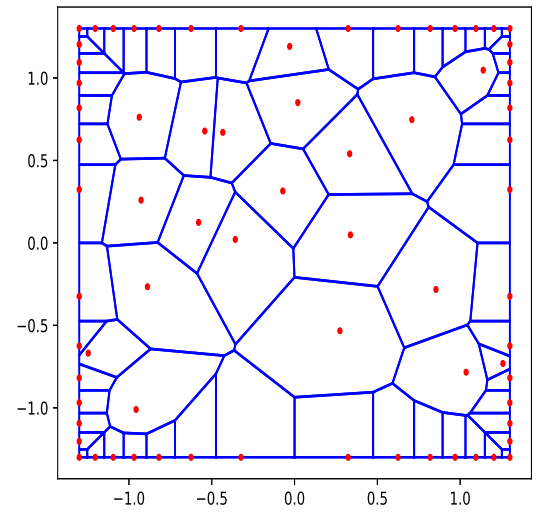
First, before considering the overall adaptive procedure described in Chapter 5, we will validate the capabilities of the implemented L_p-CVT mesh generator outlined in Chapter 4. Here we will examine the energy minimization based on an analytic metric function. This was originally defined in Section III.B of [40] based on the Hessian of the saddle point function $f(x, y) = x^2 - y^2$ on the domain $\Omega = [-1.3, 1.3]^2$

$$M(x, y) = \begin{bmatrix} 1 + 4x^2 & -4xy \\ -4xy & 1 + 4y^2 \end{bmatrix}. \tag{6.1}$$

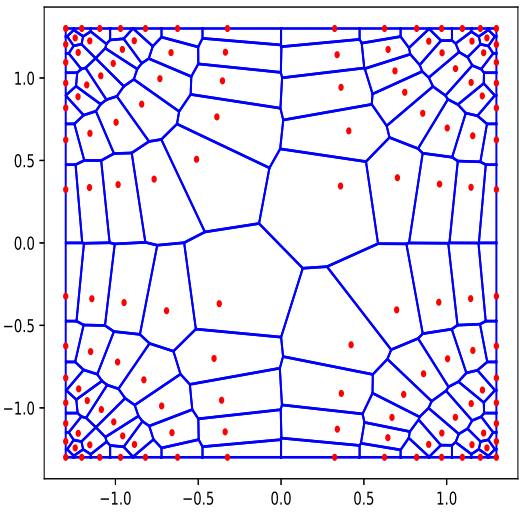
Note that this matrix appears to be derived from the triangular metric case. For example, comparing Eq. 3.6 and 3.13 for $\mathcal{M}^{-\frac{1}{2}}$ and V^{-1} respectively, the two are equivalent aside from an additional rotation matrix the left-hand side causing symmetry (and matching the form here). As a result, it is different from of our frame field estimates and results in a non-orthogonal set of basis vectors for the representative parallelogram (or respectively the unit-ball of the norm). In the 2-norm case, this difference does not matter as $\|Q\boldsymbol{y}\|_2 = \|\boldsymbol{y}\|_2$ for an orthogonal/rotation matrix. However, the distinction is important for the general L_p case. Nonetheless, it will be used here due to the availability of results for comparison.

Starting from the boundary mesh (see Section 4.2) and $n_0 = 20$ randomly distributed points, we will follow the procedure outlined in Chapter 4. Here, and for all subsequent cases, the $p = 6$ norm was chosen for use in the L_p-CVT energy as a trade-off between accuracy and smoothness of the gradients. Additionally, a valid length range of $L_M \in [0, \sqrt{2}]$ was used for the edge splitting procedure. Overall, Fig. 6.1 shows the initial and final Voronoi diagrams as well as the energy and gradient convergence.

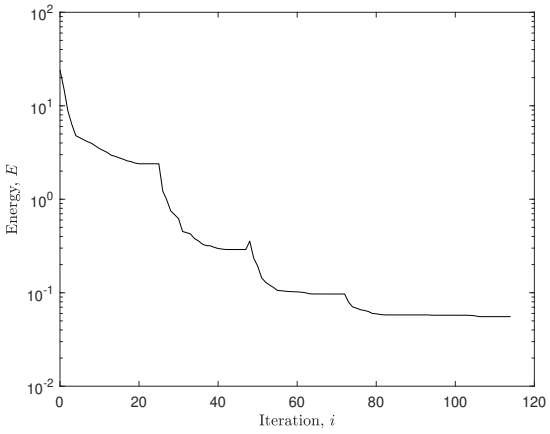
From here, we can assess that the solver is indeed behaving correctly. Comparing the initial and final Voronoi diagrams, we see that the Voronoi cells have taken the corresponding shape of the local metric function. Additionally, due to the high-order p norm, they are close to rectangular and align well along the principle directions. This includes the cells originating from the boundary nodes, that were fixed during this process. New nodes have been inserted in the correct regions near the corners to obtain the proper size distribution. In general, this behaviour matches previous results [40], however, some slight differences can be noted (e.g. missing node at $(0, 0)$). These are to be expected due to a change in starting conditions and the fact that this method converges to a local minima. The L_p-CVT energy decreased significantly through the minimization process, with the gradient converging to a tolerance of around $\mathcal{O}(1e{-}9)$ throughout all stages. Note that the gradient jumps (seen around iteration 25, 50, etc.) are due to the insertion of



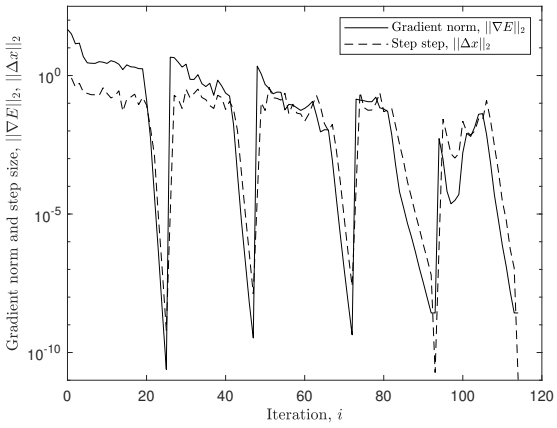
(a) Initial Voronoi Diagram



(b) Final Voronoi Diagram



(c) Energy Convergence



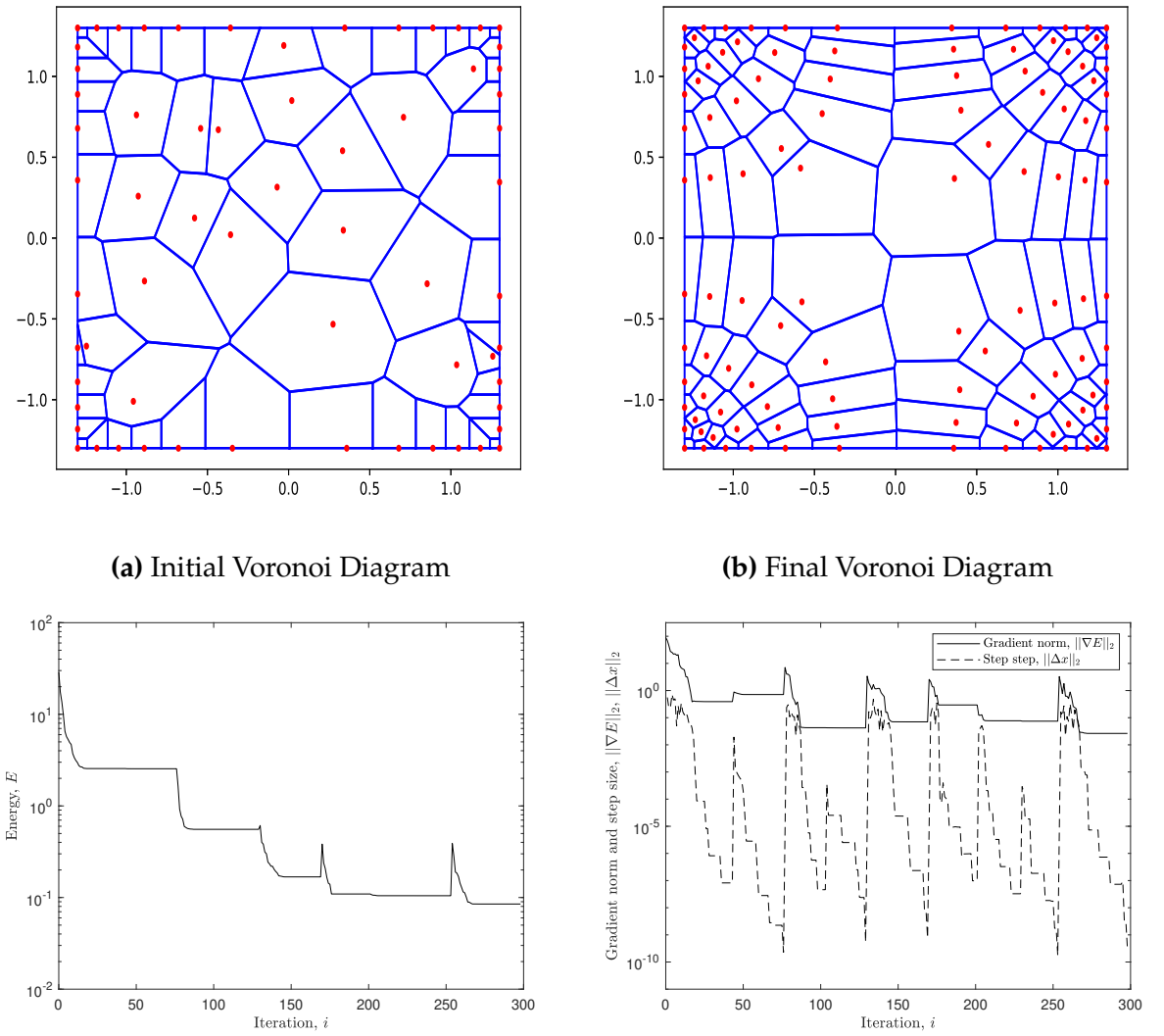
(d) Gradient norm and step size convergence

Figure 6.1: Summary of L_p-CVT on analytic quadratic metric function.

new nodes requiring the L-BFGS solver to restart. Overall, for this simple case the entire procedure takes less than 120 iterations over 5 convergence cycles.

Next, in subsequent sections, we will examine adaptive iterations with the frame field and corresponding metric derived from Chapter 5. In these cases, only discrete information is available on a background mesh for L_p-CVT solver. This results in a discontinuous metric function and requires point location for each metric evaluation $M(x, y)$. Therefore, to test the capabilities of the solver under these conditions, we repeat the previous convergence using a 16×16 background grid with metrics evaluated from the cell-center coordinate in Eq. 6.1. For comparison the same set of summary plots are shown in Fig. 6.2 (initial and final Voronoi diagrams as well as energy and gradient convergence). Notably, this change results in a lack of gradient convergence. Unlike before, we only see a 2 order drop before stalling. This is expected due to the jumps introduced in the second term of Eqs. 4.10 and 4.11 by the discontinuous metric function. In the present work we have forgone any interpolation of the metric between background mesh elements. See [41] for a discussion of some possible choices that have been proposed for use with L_p-CVT, for example, linear interpolation, a method in the exponential space from Pennec et al. [102] and a spline based approach. However, each of these methods has drawbacks making them unsuitable or impossible to use with the unstructured mesh and non-symmetric matrix case. For example, special care must be taken to avoid introducing size field singularities where $\det(M(\boldsymbol{y})) = 0$ or a very small value. Additionally, some research has been conducted on frame field interpolation, see [123].

Despite this limitation, considerable reductions in the energy are achieved through the strength of the cubic line search. Overall, we see from Fig 6.1d that process is stopped when the step size reaches the given tolerance. Additionally, looking at the final Voronoi diagram, we see similar behaviour to before. Even then, this is partially caused by the differences in the boundary mesh due to the change in metric sizes along the edge (and rounding to an integer number of unit-edge segments). Therefore, this method achieves the desired behaviour at the cost of requiring more iterations to achieve convergence.



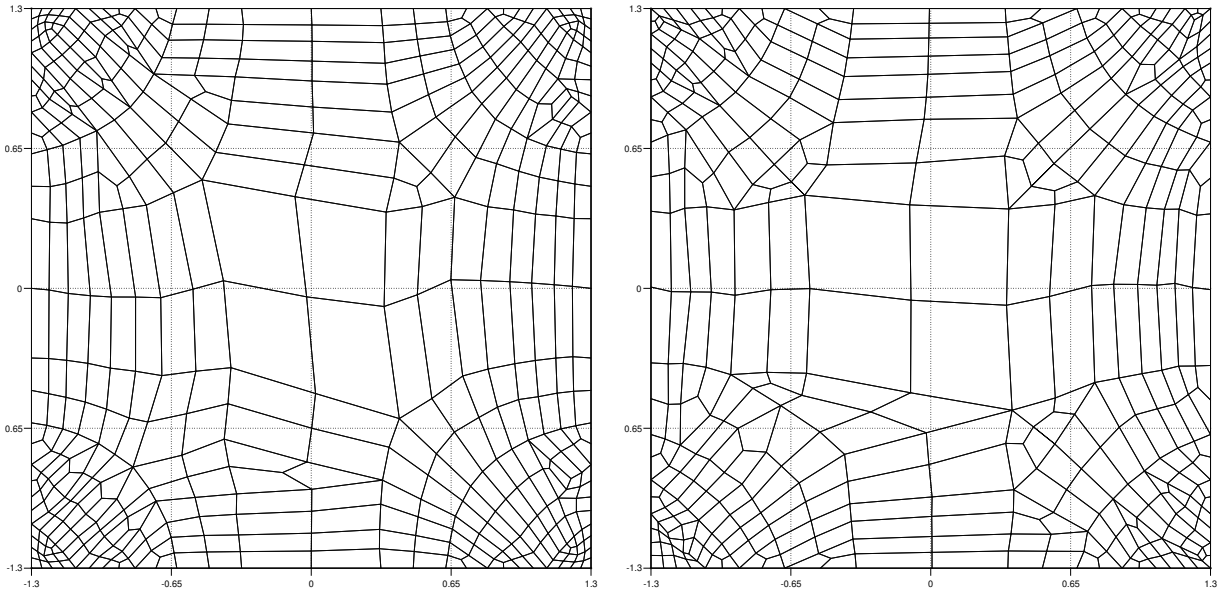
(a) Initial Voronoi Diagram

(b) Final Voronoi Diagram

(c) Energy Convergence

(d) Gradient norm and step size convergence

Figure 6.2: Summary of L_p-CVT on background quadratic metric function.



(a) Analytic metric **(b)** Background metric

Figure 6.3: Comparison of final all-quad mesh on quadratic metric function.

After conducting the post processing steps, these similarities are also reflected in the final all-quad mesh (compared in Fig. 6.3). Here, the final Voronoi diagrams have been triangulated (with a Constrained Delaunay Triangulation), merged to form a quad-dominant mesh, then split and smoothed to create the all-quad mesh shown. Overall, this produces aligned and anisotropic quad elements which match the local solution behaviour. Merging has also lead to a reduction in the number of small angles introduced by splitting anisotropic triangles. Therefore, through this initial result, we justify the use of this method to conform to our computed discrete frame field targets in the sections to come.

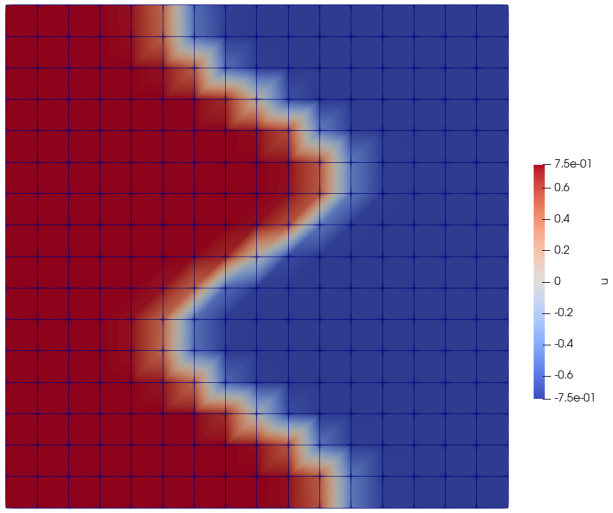


Figure 6.4: Initial S-Shock solution ($p = 2$).

6.2 Feature-Based S-Shock Adaptation

For the first complete example, we will examine the behaviour of this method for feature-based adaptation of a weak shock. This case is an anisotropic advection-diffusion problem with a manufactured solution

$$\nabla \cdot (D\nabla u) - C \cdot \nabla u = S, \tag{6.2}$$

where

$$D = \frac{0.01\pi}{e}\begin{bmatrix} 12 & 3 \\ 3 & 20 \end{bmatrix}, \qquad C = \left[1.1, -\frac{\pi}{e}\right]^T, \tag{6.3}$$

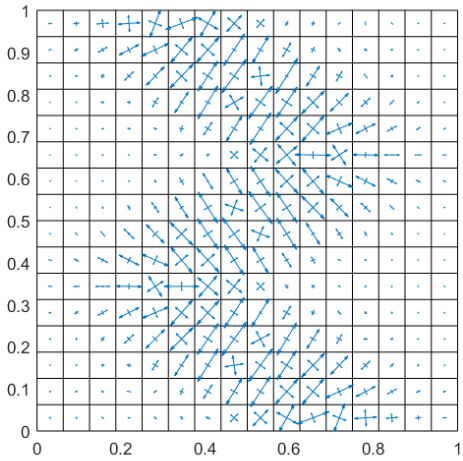
and S is the manufactured source term corresponding to the exact solution

$$u(x, y) = 0.75 \tanh\left(4 \sin\left(10y - 5\right) - 24x + 12\right), \tag{6.4}$$

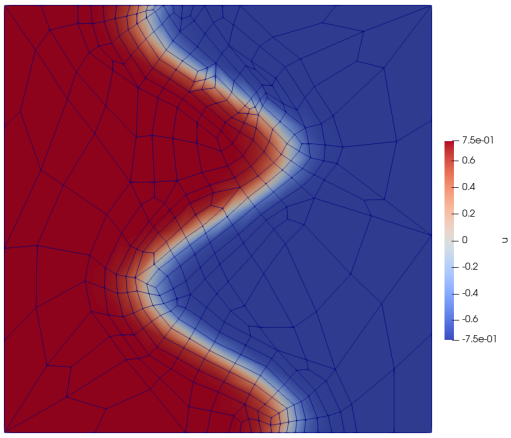
defined on $\Omega = [0, 1]^2$. Convergence of the solution in the L_2 norm was studied starting from an initial 16x16 grid (see Fig. 6.4) with uniform polynomial orders (ranging from 1-3). For reference, the behaviour was compared with uniform refinement and a fixed-

fraction approach with 30% splitting based on the largest order $p+1$ directional derivative of the reconstructed solution. To evaluate the effectiveness of the L_p-CVT mesh generator implemented here, we will also consider using the target frame field estimates with a BAMG mesh generated by GMSH [51, 58] (merged with Blossom-Quad [110] to produce an all-quad mesh). For this case, the Riemannian metric field is assembled via $\mathcal{M} = M^T M$ (based on the discussion in section 3.4). For each of the continuous methods (BAMG and L_p-CVT), we use a $1.5\times$ target complexity growth for each step (or equivalently increase in total DOFs). For the L_p-CVT mesh generator, we use an edge splitting tolerance of $L_M \in [0.25, 2.0]$ and terminate the procedure when less than 2% of edges are flagged on consecutive iterations. This range was chosen conservatively to prevent oscillations in the adding and removal of nodes while mainly targeting the coarse initial edges for splitting. Note that unlike the previous example, this case makes full use of the error estimation, target frame generation and unstructured background metric function (requiring point location) to iteratively adapt the mesh.

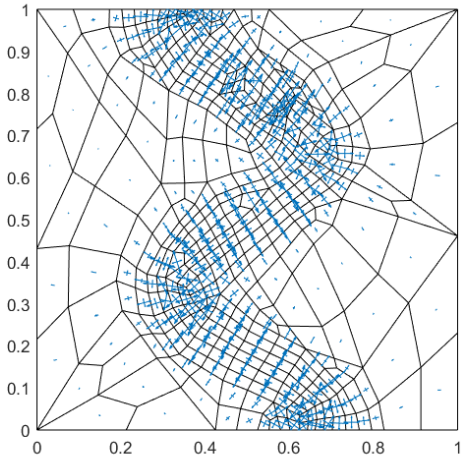
To clarify the details of this procedure before looking at the overall results, the sequence of target frame fields and corresponding discrete mesh are shown in Fig. 6.5 (for the $p = 2$, L_p-CVT case). Starting from the initial solution on the coarse grid from Fig. 6.4, the continuous error estimation process of Chapter 5 is used to derive the target frame field shown in 6.5a. Here the frames are plotted as the underlying vector set, however, with the axes lengths inverted to better highlight areas of refinement. It can be seen that even on the first iteration, the refinement targets clearly outline the shock shape (with coarsening away from this area) and produce frame axes aligned with the natural curvature. In particular, it can be observed that greater refinement is requested along the transverse shock direction as would be expected to capture the steep gradients. Fig. 6.5b shows the adapted grid resulting from this first target frame field. The grid already follows the S-Shape in semi-structured patches, greatly improving the resolution of the shock front. However, the process does not stop here and is repeated on the new (now unstructured) background mesh from this flow solution. Gradually throughout the iterations, it can be



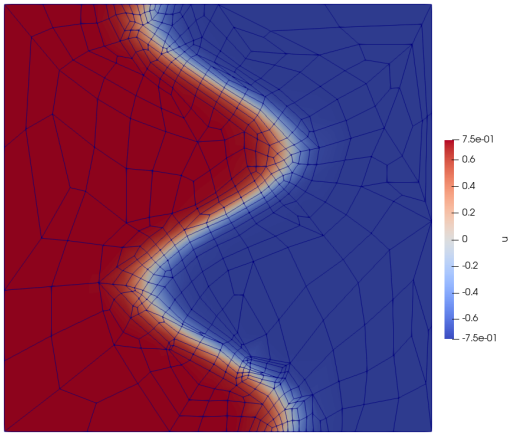
(a) Input frame field (Adaptation #1)



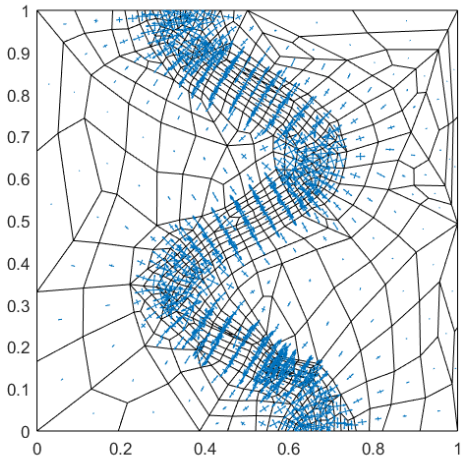
(b) Output mesh solution (Adaptation #1)



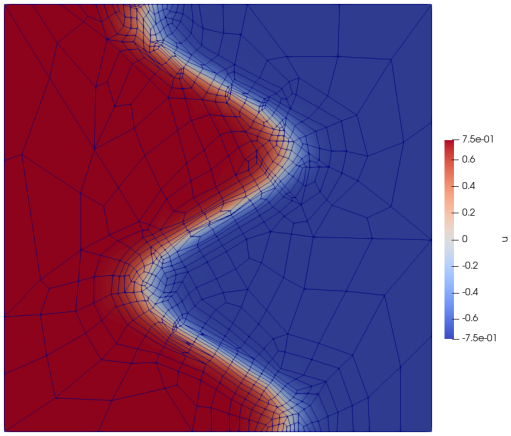
(c) Input frame field (Adaptation #2)



(d) Output mesh solution (Adaptation #2)

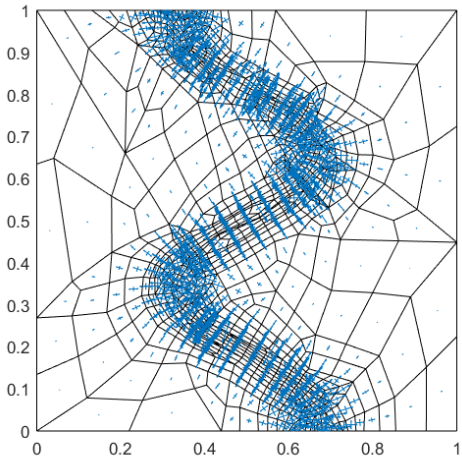


(e) Input frame field (Adaptation #3)

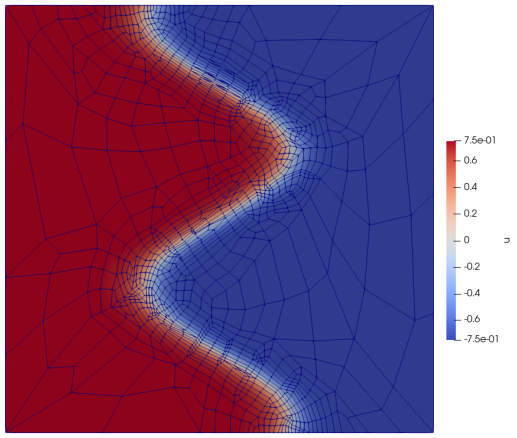


(f) Output mesh solution (Adaptation #3)

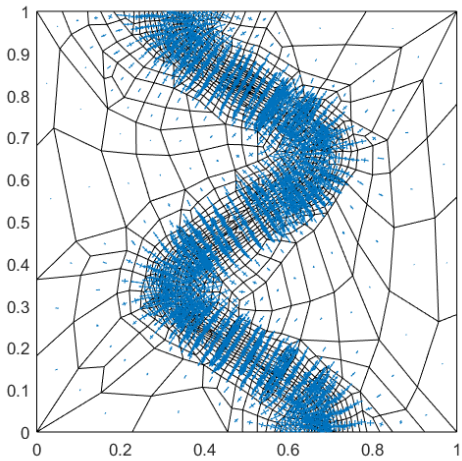
Figure 6.5: Sequence of target frame fields $(1/h)$ and L_p-CVT adapted grids, feature-based S-Shock ($p = 2$).



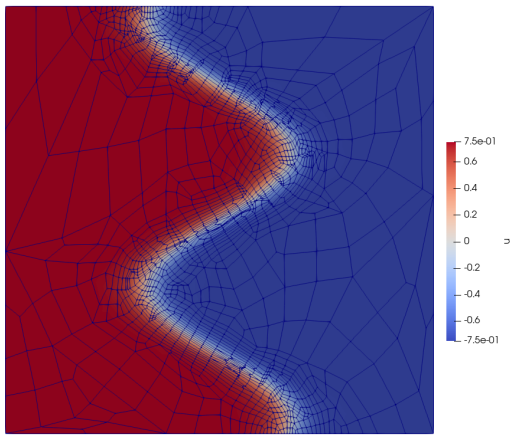
(g) Input frame field (Adaptation #4)



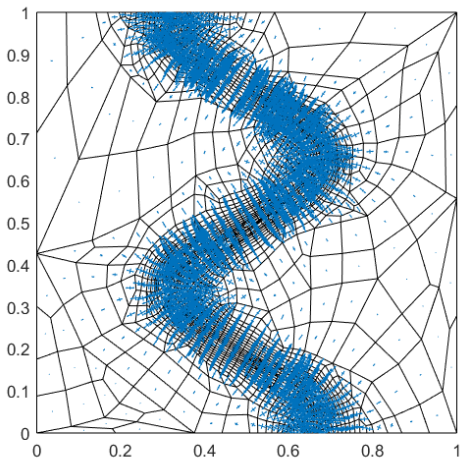
(h) Output mesh solution (Adaptation #4)



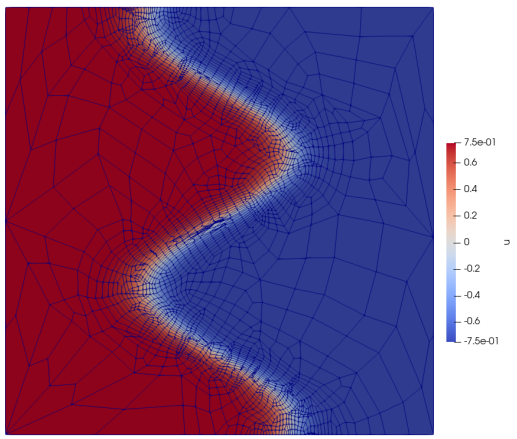
(i) Input frame field (Adaptation #5)



(j) Output mesh solution (Adaptation #5)



(k) Input frame field (Adaptation #6)



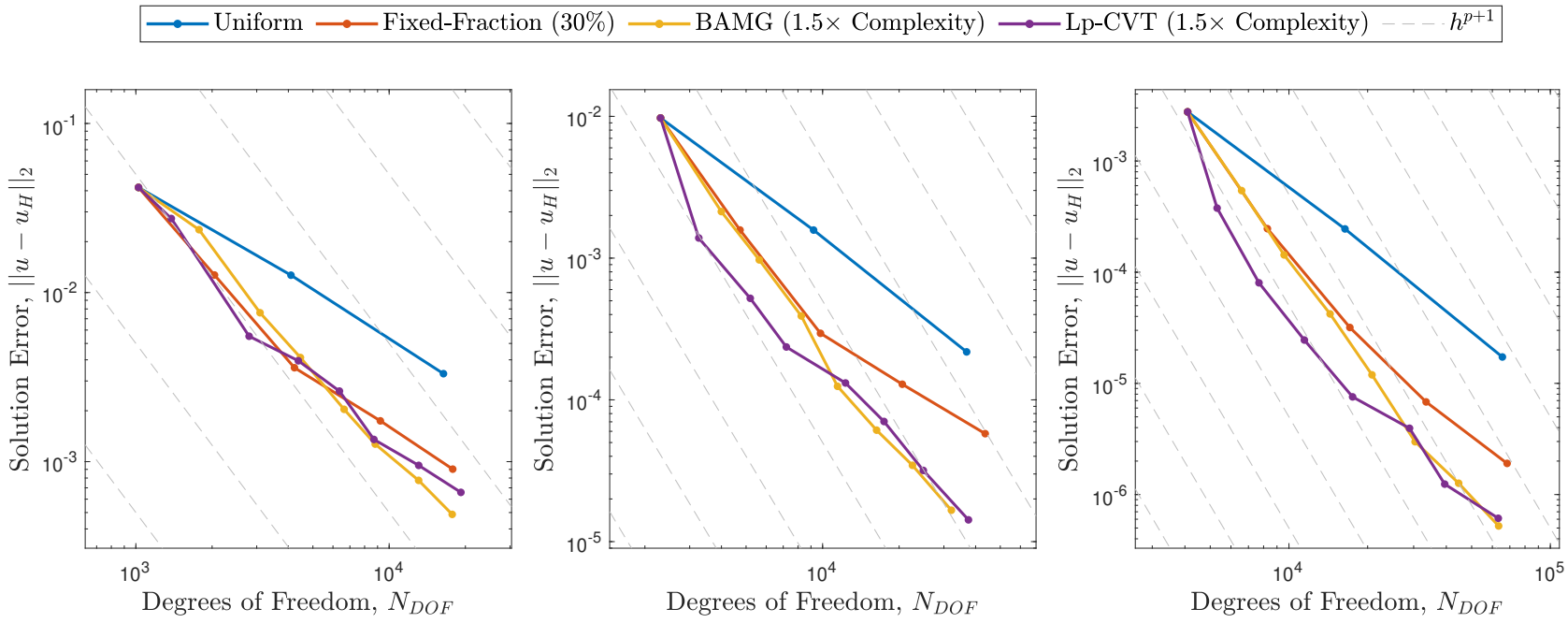
(l) Output mesh solution (Adaptation #6)

Figure 6.5: Sequence of frame fields $(1/h)$ and L_p-CVT adapted grids, feature-based S-Shock ($p = 2$). Continued...

observed how further refinement is targeted toward this crucial area of the solution. As the flow solution becomes more accurate during this process, the frame field estimates also see improvements and maintain good alignment (across the shock) with their neighbouring cells. This can be observed through the consistent rows of vectors from the frame field across the straight regions of the shock. Finally, the process terminates either at some desired error tolerance, or as in our case here, a maximum number of DOFs is reached.

Next, the overall error convergence results are shown in Fig. 6.6. In general, the continuous error estimates performed favorably in all cases. Towards the final iterations of the $p = 1$ case, the BAMG mesh was able to outperform L_p-CVT with both offering slight improvements over a fixed fraction approach. However, the benefits of the L_p-CVT mesh generator are particularly noticeable at the beginning of the $p = 2$ and $p = 3$ cases. Towards the end, the continuous methods evened out, with both offering a sizeable improvement over a typical fixed fraction approach. Here, the dashed grey lines indicate the optimal asymptotic convergence rate for the DG method, $\mathcal{O}(h^{p+1})$. Note, this may not be attained due to the need to sufficiently resolve all flow features. In this case, the adaptive methods allow convergence to meet or exceed this rate starting from a uniform mesh which is not yet in the asymptotic regime.

Additionally, Table 6.1 compares the wall clock time required for solving the advection diffusion problem on each final mesh. In general, similar computational effort is needed for all methods in reaching the final accuracy shown. Note that although the uniform case ran slightly faster, it has been omitted due to its significantly higher error levels. Although there are variations between the other methods, some of the discrepancies can be explained by differences in the number of degrees of freedom for each mesh.



(a) $p = 1$ **(b)** $p = 2$ **(c)** $p = 3$

Figure 6.6: Convergence of L_2 error for the feature-based S-Shock case.

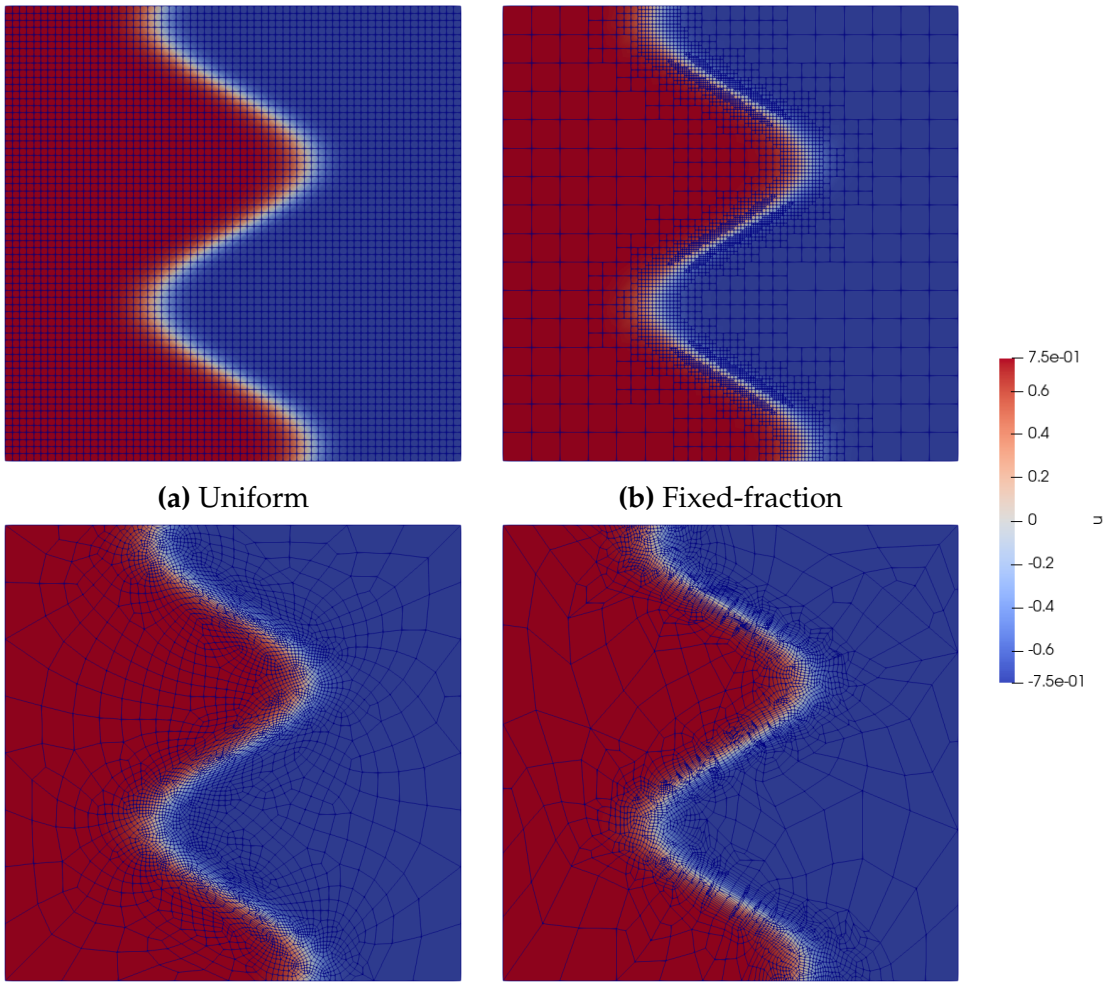
		N_{DOF}	$\|u - u_H\|_2$	t
$p = 1$	Fixed-Fraction	1.78×10^4	9.03×10^{-4}	8.74
	BAMG	1.77×10^4	4.88×10^{-4}	6.34
	L_p-CVT	1.92×10^4	6.58×10^{-4}	7.00
$p = 2$	Fixed-Fraction	4.36×10^4	5.77×10^{-4}	57.89
	BAMG	3.21×10^4	1.66×10^{-5}	28.66
	L_p-CVT	3.75×10^4	1.42×10^{-5}	42.46
$p = 3$	Fixed-Fraction	6.83×10^4	1.91×10^{-6}	196.22
	BAMG	6.34×10^4	5.22×10^{-7}	151.79
	L_p-CVT	6.32×10^4	6.12×10^{-7}	194.01

Table 6.1: Comparison of final mesh solution wall clock times (in seconds) for the feature-based S-Shock case.

Error trends can be justified by examining the final mesh for each case in Figs. 6.7 ($p = 1$), 6.9 ($p = 2$) and 6.11 ($p = 3$). From here, we can see differences in the areas highlighted by the fixed-fraction method. This corresponds with the regions of largest directional derivatives, which in particular for these cases, lie along the straight diagonals of the shock. For the $p = 1$ case, there appears to be an inflection point for the quadratic error estimate. However, for the $p = 2$ and $p = 3$ cases, the behaviour appears to be continuous across the shock.

Moving on to the continuous methods, as examined from the iterative procedure before (based on the $p = 2$, L_p-CVT case), these are able to completely re-mesh the domain between iterations. As a result, this reduces the dependency of these methods on the initial mesh configuration and allows them to retarget the initial DOFs towards the shock. This is one of the reasons the L_p-CVT method was able to perform noticeably better in the early iterations of the $p = 2$ and $p = 3$ cases. Both methods refine along the shock, introducing anisotropy in the straight diagonal segments and coarsening other areas of the domain. One noticeable difference for both orders is the smoother transition of mesh sizes in the BAMG case away from this region. This mainly results from the inclusion of metric smoothing in the method.

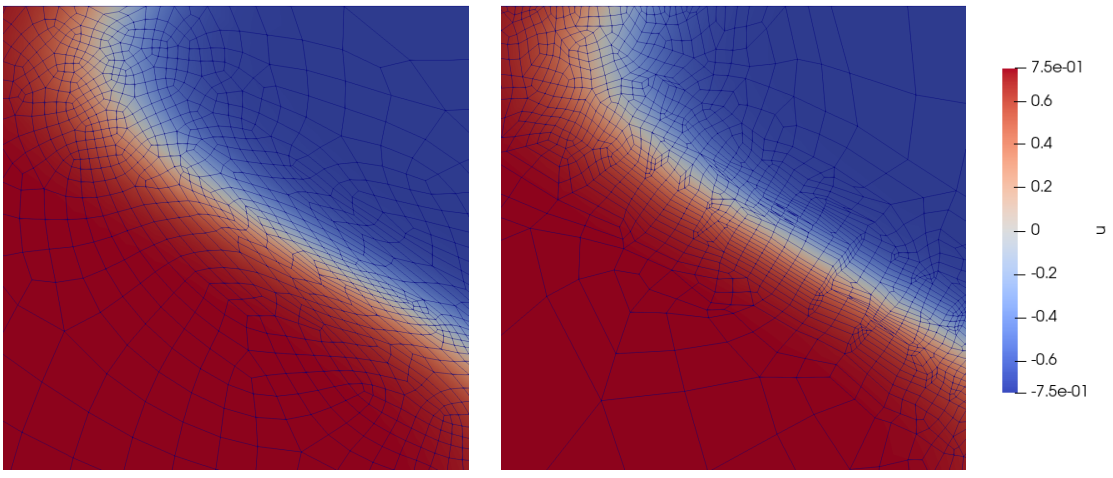
To better distinguish the details between the meshing behaviour along the shock, Figs. 6.8 ($p = 1$), 6.10 ($p = 2$) and 6.12 ($p = 3$) show a closeup of the top-most shock region intersecting the boundary. First, in the $p = 1$ case, we notice the alignment of the cells along the shock forming somewhat regular and aligned rectangular shapes. On the other hand, the cells produced by BAMG are more random as the triangular based mesh generator offers no control over cell orientation. Additionally, examining closely, the L_p-CVT produces slightly coarser cells in the center of the shock near the inflection point, as desired from the observations of the fixed-fraction method. On the other hand, this behaviour is smoothed out for the BAMG mesh. In the $p = 2$ case, changes across the shock are more gradual. Here, the row of quad elements for the L_p-CVT mesh curve naturally originate from the boundary along the shock shape. Once again, this produces an aligned quad structure instead of the irregular orientations present in the BAMG mesh. Similar trends were also observed for the $p = 3$ case. Overall, both methods systematically introduce anisotropy, helping to accurately capture the solution behaviour and highlighting the usefulness of the frame field targets.



(a) Uniform **(b)** Fixed-fraction

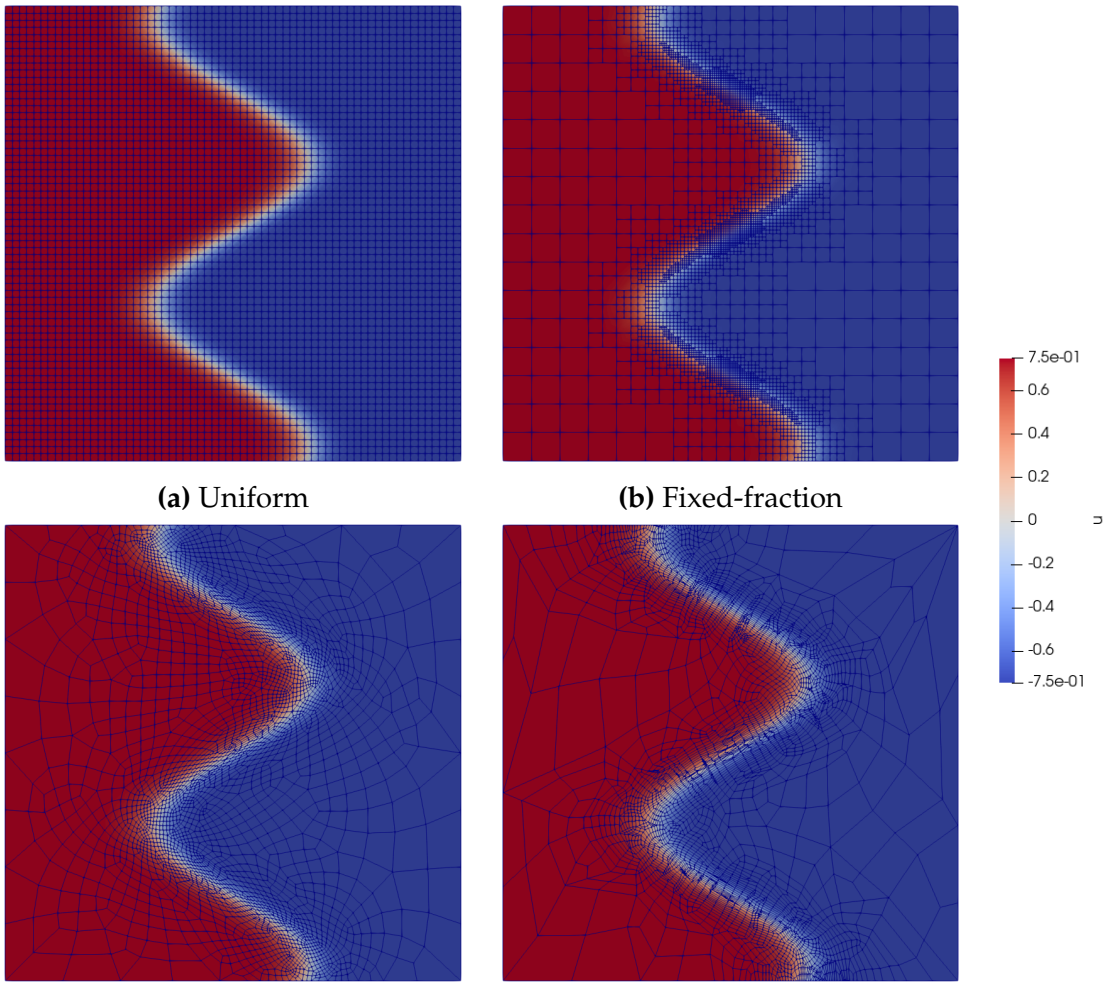
(c) BAMG **(d)** L_p-CVT

Figure 6.7: Final mesh obtained for the feature-based S-Shock ($p = 1$).



(a) BAMG **(b)** L_p-CVT

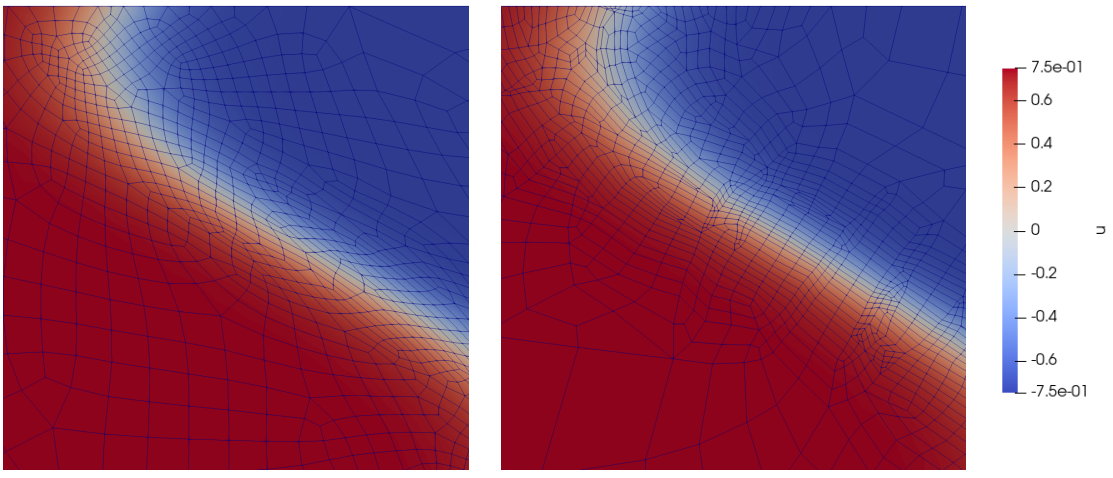
Figure 6.8: Closeup of final mesh obtained for the feature-based S-Shock ($p = 1$).



(a) Uniform **(b)** Fixed-fraction

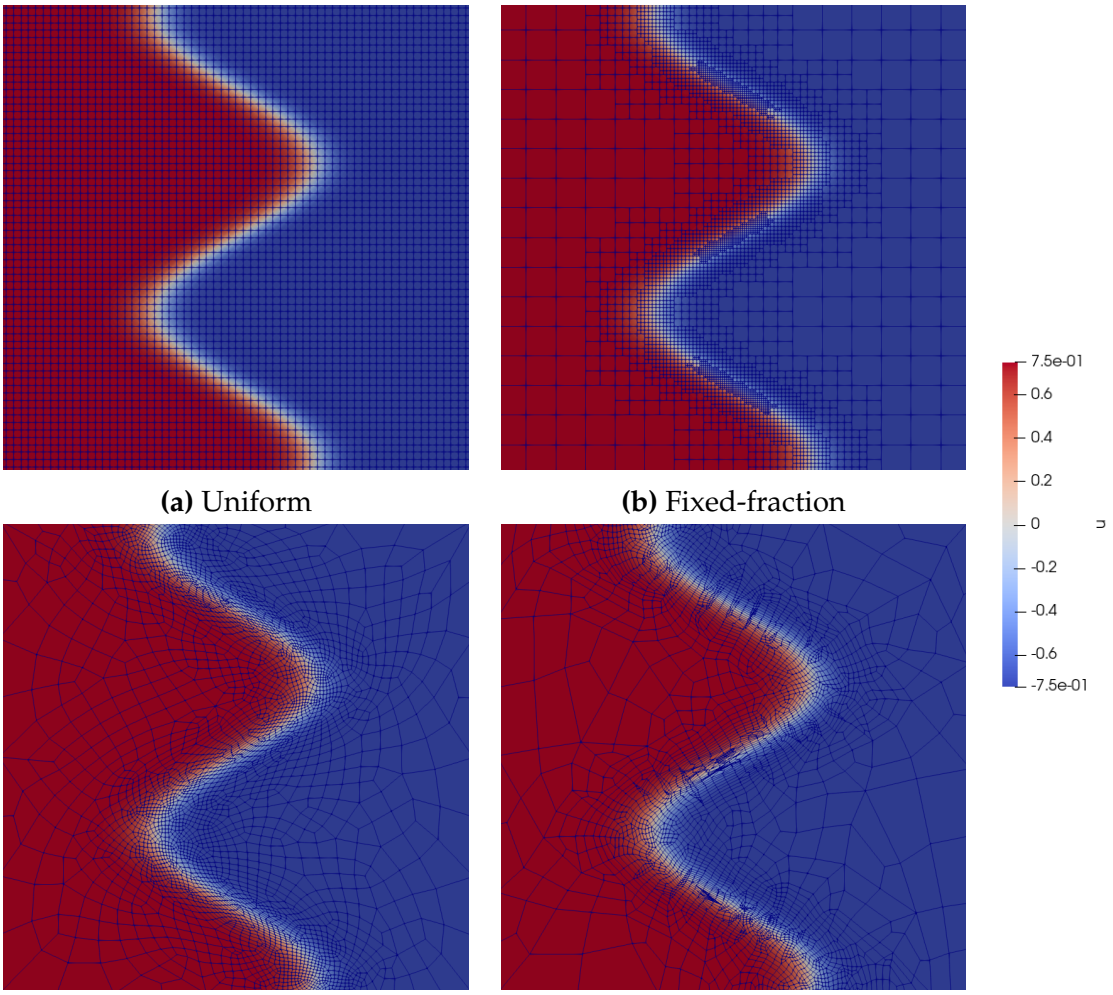
(c) BAMG **(d)** L_p-CVT

Figure 6.9: Final mesh obtained for the feature-based S-Shock ($p = 2$).



(a) BAMG **(b)** L_p-CVT

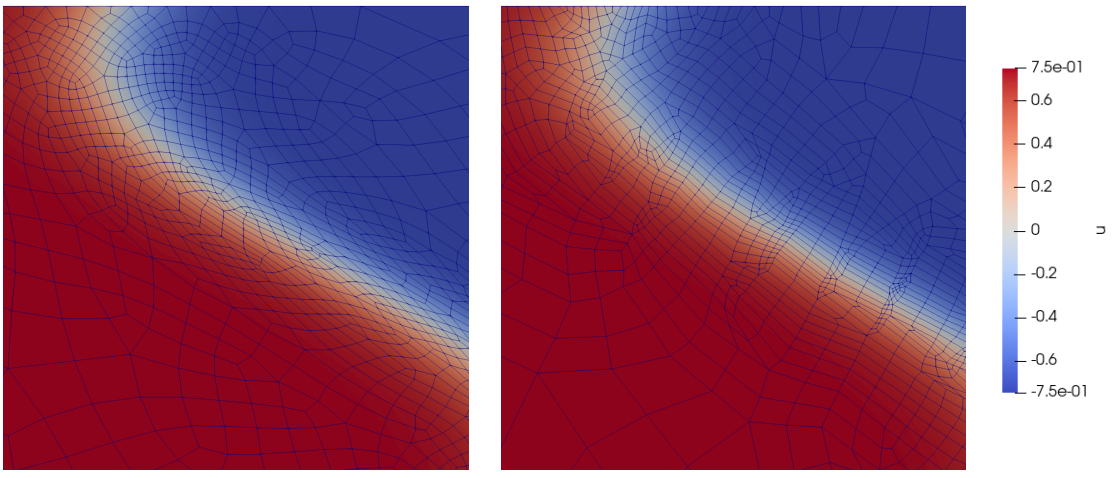
Figure 6.10: Closeup of final mesh obtained for the feature-based S-Shock ($p = 2$).



(a) Uniform **(b)** Fixed-fraction

(c) BAMG **(d)** L_p-CVT

Figure 6.11: Final mesh obtained for the feature-based S-Shock ($p = 3$).



(a) BAMG **(b)** L_p-CVT

Figure 6.12: Closeup of final mesh obtained for the feature-based S-Shock ($p = 3$).

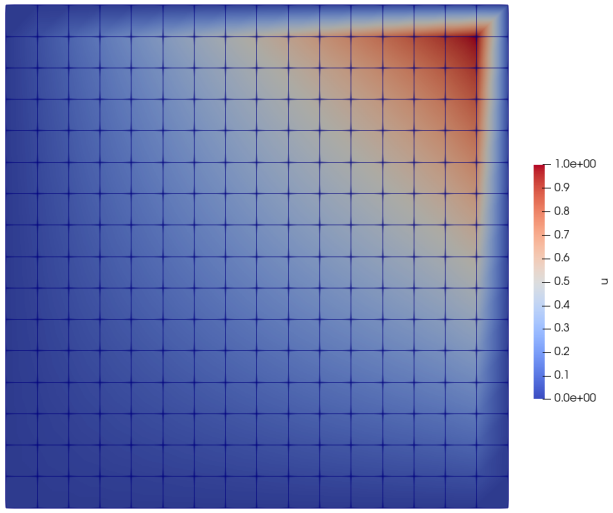


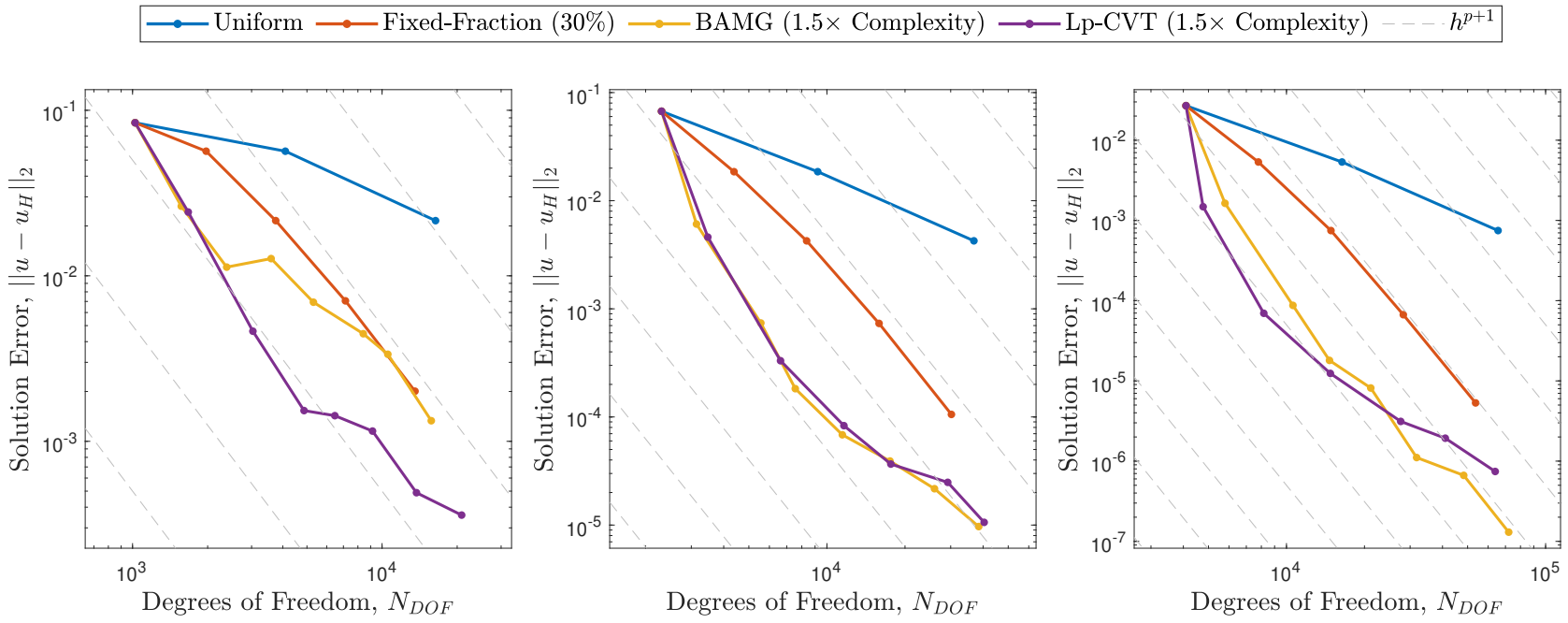
Figure 6.13: Initial boundary layer solution ($p = 2$).

6.3 Feature-Based Boundary Layer Adaptation

Next, looking at another feature-based advection-diffusion case. This time to examine the behaviour of this method in the presence of an artificial boundary layer. For this case, the manufactured solution is given by

$$u(x, y) = \left(x + \frac{e^{x/\varepsilon} - 1}{1 - e^{1/\varepsilon}}\right)\left(y + \frac{e^{y/\varepsilon} - 1}{1 - e^{1/\varepsilon}}\right), \tag{6.5}$$

where $\varepsilon = 0.005$ (as used in [35] for their steeper boundary layer). This results in a quadratic behaviour that quickly decays in the upper and right sides as shown on the coarse initial grid in Fig. 6.13. The success of the various methods will depend heavily on their ability to incorporate small cell sizes along the steep and rapidly changing gradients in this region. For the diffusion and advection constants in Eq. 6.2, $D = 0.1I$ and $C = [1, 1]^T$ were used respectively. Again, the 4 methods described before (uniform, fixed-fraction, and continuous estimates with BAMG and L_p-CVT mesh generators) were used with polynomial orders 1 to 3 and the error convergence in the L_2 norm was plotted in Fig. 6.14. Once again, Table 6.2 shows the wall clock times needed to compute the solution on each final mesh for the various methods and polynomial orders. In general, this case performed slightly faster with similar trends throughout.



(a) $p = 1$ **(b)** $p = 2$ **(c)** $p = 3$

Figure 6.14: Convergence of L_2 error for the feature-based boundary layer case.

		N_{DOF}	$\|u - u_H\|_2$	t
$p = 1$	Fixed-Fraction	1.36×10^4	2.01×10^{-3}	3.82
	BAMG	1.57×10^4	1.33×10^{-3}	3.81
	L_p-CVT	2.09×10^4	3.58×10^{-4}	6.17
$p = 2$	Fixed-Fraction	3.02×10^4	1.06×10^{-4}	20.12
	BAMG	3.85×10^4	9.68×10^{-6}	30.17
	L_p-CVT	4.04×10^4	1.06×10^{-5}	28.46
$p = 3$	Fixed-Fraction	5.37×10^4	5.32×10^{-6}	87.26
	BAMG	7.20×10^4	1.30×10^{-7}	153.14
	L_p-CVT	6.39×10^4	7.43×10^{-7}	113.11

Table 6.2: Comparison of final mesh solution wall clock times (in seconds) for the feature-based boundary layer case.

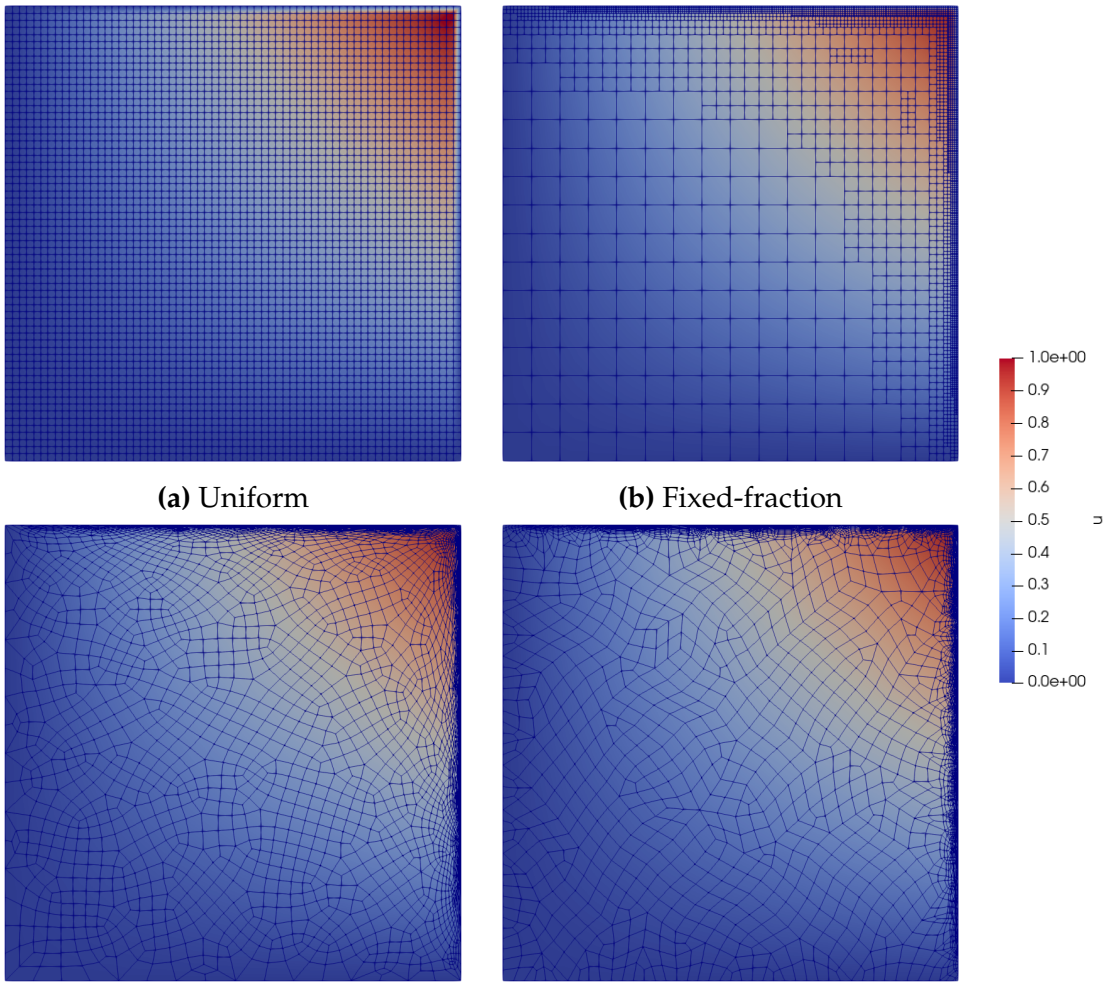
For this case, the use of the continuous methods offered significant benefits across all cases compared to the uniform and fixed-fraction refinements. This is particularly noticeable in the early iterations when these methods can re-purpose the initial DOFs to the highly anisotropic boundary layer region. In particular for the $p = 1$ case, the L_p-CVT method was able to maintain this improvement despite stalling of the BAMG convergence after a few iterations (making it more comparable with the error levels of the fixed-fraction method). For $p = 2$, both continuous methods saw nearly equivalent convergence, with both offering solid benefits. However, we will see below that the exact characteristics of the mesh used to attain these levels are rather distinct. Finally for the $p = 3$ case, the L_p-CVT method started off the best, but, began to stall at later iterations due to reaching limitations in its anisotropy. Unfortunately, this is to be expected at a certain point with the L_p-CVT method as the hessian has been observed to be ill-conditioned in highly anisotropic regions [74]. This problem is further exacerbated by the use of a discrete background metric function. Overall, once the error is sufficiently small and equidistributed over the domain, the refinement is expected to be uniform throughout with error convergence in the asymptotic range.

Once again, we will examine the final mesh to justify these trends. First, Fig. 6.15 shows the set of final $p = 1$ mesh. Here, unlike the previous test case, there is uniform

refinement throughout the domain (due to the non-constant function behaviour). For the most part the continuous methods produced visually similar results, however, the metric smoothing of the BAMG resulted in patches of larger cells directly on the boundary (for example, at the top center). As a result, a large portion of the overall error came from these relatively few cells. This resulted in the decreased error convergence relative to the L_p-CVT method. Fig. 6.16 shows a closeup of one of these problematic areas near the center on the top edge of the mesh. Looking at the boundary region, there is a noticeable difference in the size of elements closest to the wall. On the other hand, the L_p-CVT method produced a more regular and well aligned mesh in this region.

For the $p = 2$ and $p = 3$ cases, the final mesh are quite different as shown in Figs. 6.17 and 6.19 respectively. As the lower left region of domain is nearly quadratic, it can be accurately captured with a minimal number of elements. This allowed for coarsening in this region for the continuous methods, leaving the steep boundary layer as the main feature to be refined. As a result, the element size nearest to the wall is crucial to accurately resolve the solution. Additionally, as there is minimal behaviour in the transverse direction, anisotropic cells help limit the added DOFs and corresponding complexity.

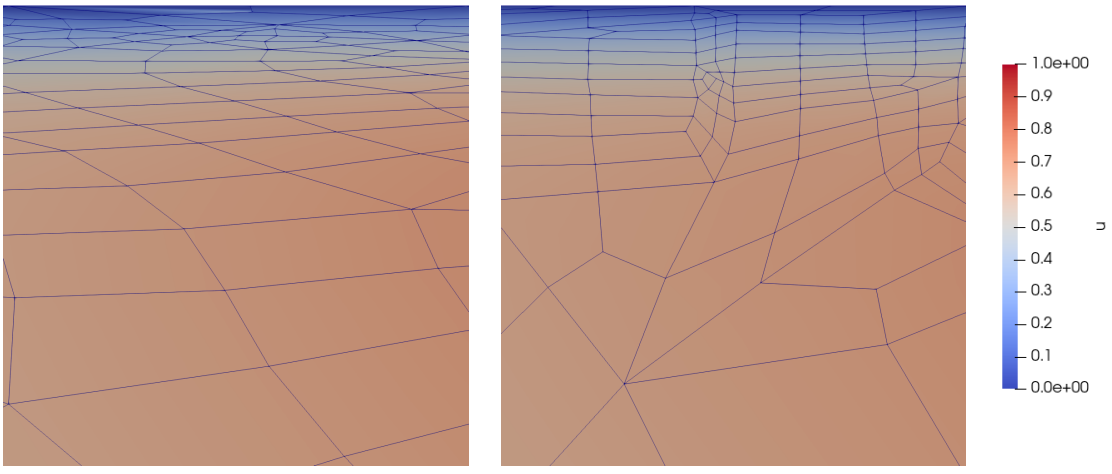
This can be better observed from Figs. 6.18 and 6.20 showing a closeup of the top-right corner of the mesh for the BAMG and L_p-CVT methods. Here, both mesh generators successfully incorporate small isotropic elements directly in the corner with anisotropy along the sides. However, there is a clear trade-off between the two methods in this region. First, BAMG is able to produce more anisotropic grids with a smoother transition from the refined areas (owing to the metric smoothing). However, this produces diagonal or otherwise misaligned cells with small minimum angles. On the other hand, L_p-CVT produces better aligned quads but at the cost of decreased anisotropy near the wall. The L_p-CVT minimization in this region also becomes more challenging as the boundary points are constrained by the pre-processing step.



(a) Uniform **(b)** Fixed-fraction

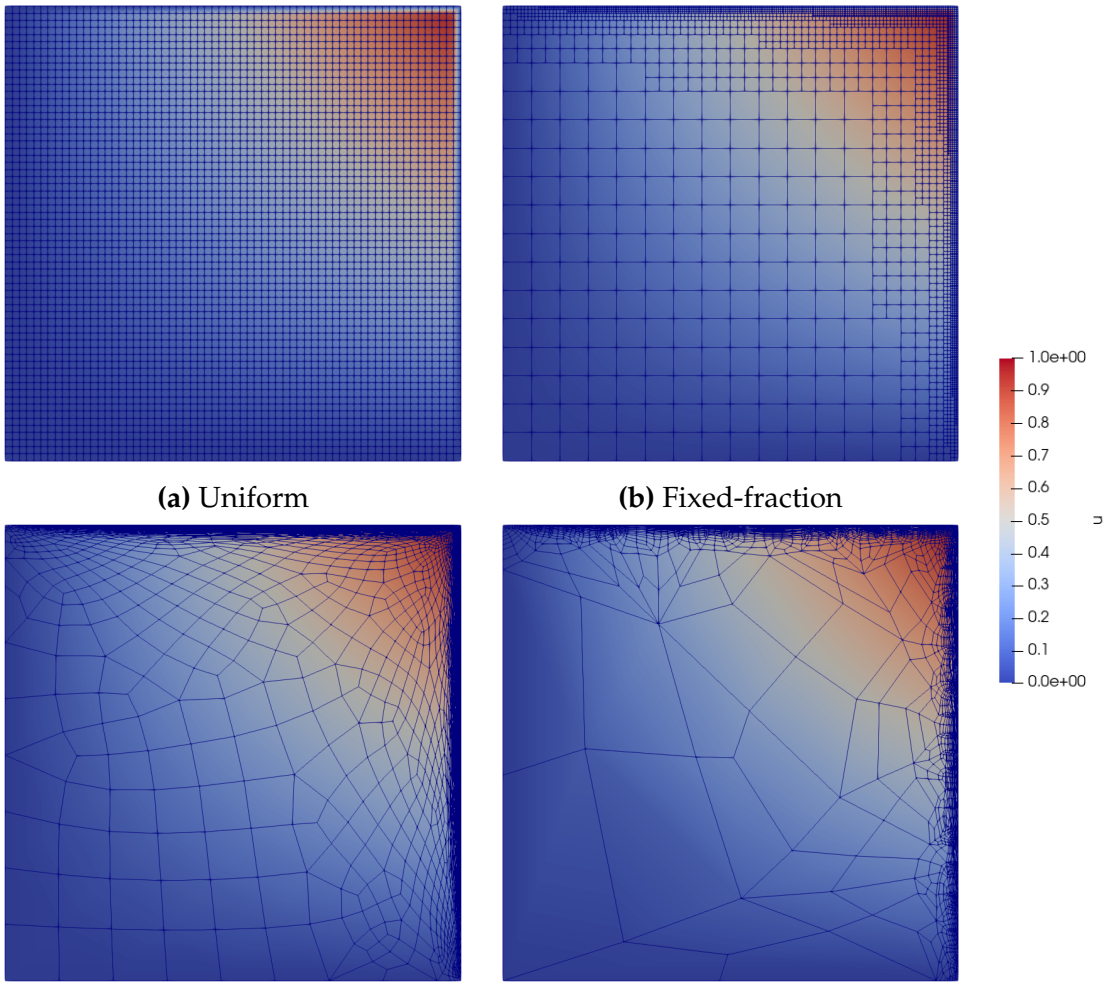
(c) BAMG **(d)** L_p-CVT

Figure 6.15: Final mesh obtained for the feature-based boundary layer ($p = 1$).



(a) BAMG **(b)** L_p-CVT

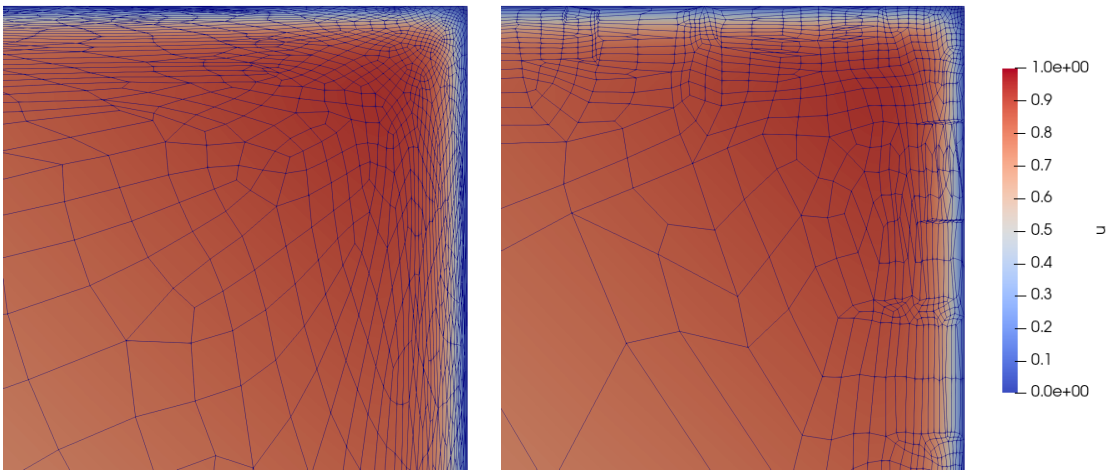
Figure 6.16: Closeup of final mesh obtained for the feature-based boundary layer ($p = 1$).



(a) Uniform **(b)** Fixed-fraction

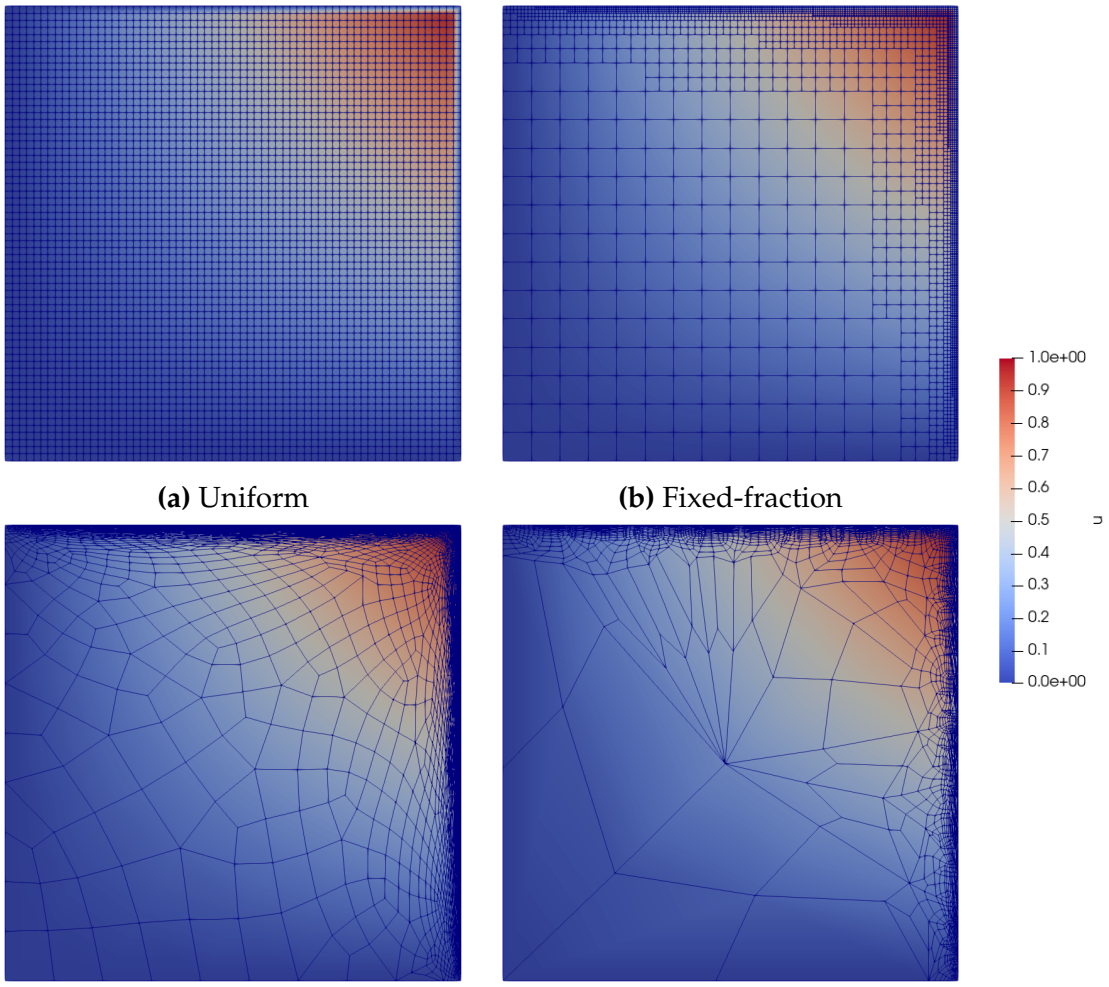
(c) BAMG **(d)** L_p-CVT

Figure 6.17: Final mesh obtained for the feature-based boundary layer ($p = 2$).



(a) BAMG **(b)** L_p-CVT

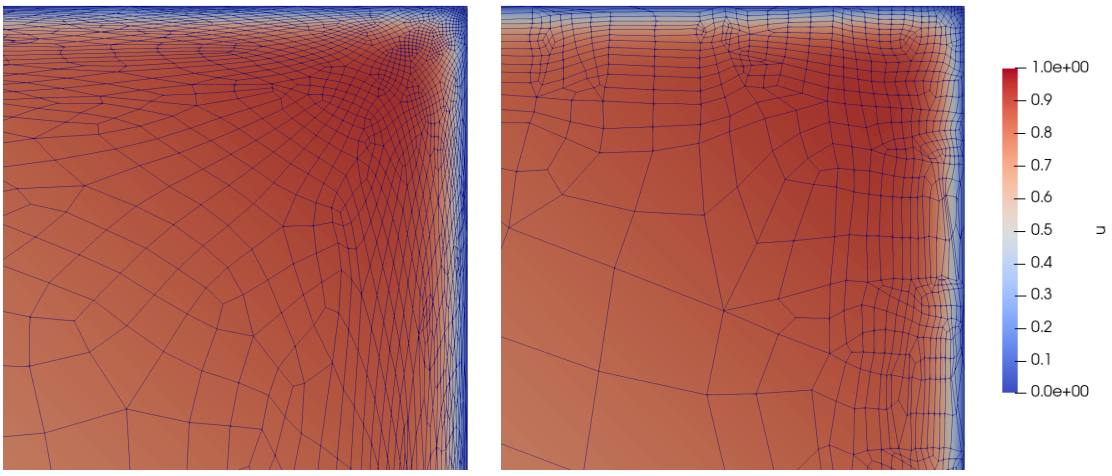
Figure 6.18: Closeup of final mesh obtained for the feature-based boundary layer ($p = 2$).



(a) Uniform **(b)** Fixed-fraction

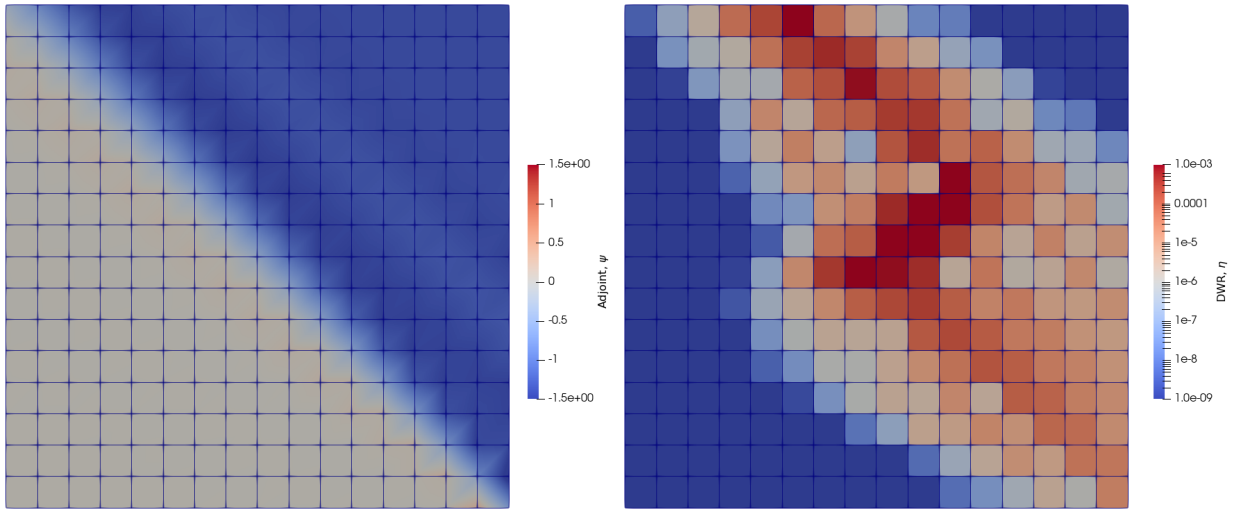
(c) BAMG **(d)** L_p-CVT

Figure 6.19: Final mesh obtained for the feature-based boundary layer ($p = 3$).



(a) BAMG **(b)** L_p-CVT

Figure 6.20: Closeup of final mesh obtained for the feature-based boundary layer ($p = 3$).



(a) Adjoint, ψ **(b)** DWR, η

Figure 6.21: Initial S-Shock adjoint and logarithmic DWR ($p = 1$).

6.4 Goal-Oriented S-Shock Adaptation

Moving on to the adjoint-based error indicator and goal-adaptation procedure from Section 5.4. We return to the S-Shock case from Section 6.2 for an advection-only problem (with the same flow constants as before). Here, the new goal is to accurately predict the functional corresponding to the 2 norm squared over the right boundary of the domain

$$\mathcal{J}(u) = \int_{\Gamma_{right}} \|u\|_2^2 \, d\Gamma. \tag{6.6}$$

As the behaviour is less intuitive for this case, Fig. 6.21 shows the initial $p = 1$ adjoint solution and corresponding DWR (on a logarithmic scale). This highlights the areas with largest contribution to the functional error.

For goal-oriented cases, the DWR will be used as an indicator for the 30% fixed-fraction flagging. For the continuous methods, the orientation and anisotropy of the target frame are derived from the primal solution behaviour. However, for the size updates, we use the quadratic fit of the logarithmic DWR from Eq. 5.24. For the two goal-oriented test cases, $[r_{max}, c_{max}] = [20, 4]$ are used for the size update range with the same complexity step of $1.5\times$. We will be comparing the 4 convergence rates on polynomial orders

		N_{DOF}	$\lvert\mathcal{J}(u) - \mathcal{J}_H(u_H)\rvert$	t (sol.)	t (adj.)
$p = 1$	Fixed-Fraction	3.00×10^4	5.17×10^{-8}	14.69	19.57
	BAMG	2.08×10^4	6.15×10^{-8}	8.05	13.31
	L_p-CVT	2.50×10^4	2.76×10^{-7}	10.20	15.72
$p = 2$	Fixed-Fraction	3.50×10^4	3.80×10^{-9}	41.92	48.21
	BAMG	2.48×10^4	1.12×10^{-9}	22.40	23.65
	L_p-CVT	2.38×10^4	5.93×10^{-9}	25.87	28.93
$p = 3$	Fixed-Fraction	6.09×10^4	2.16×10^{-12}	221.96	136.58
	BAMG	7.72×10^4	1.69×10^{-11}	222.92	120.13
	L_p-CVT	6.19×10^4	1.67×10^{-10}	172.88	134.42

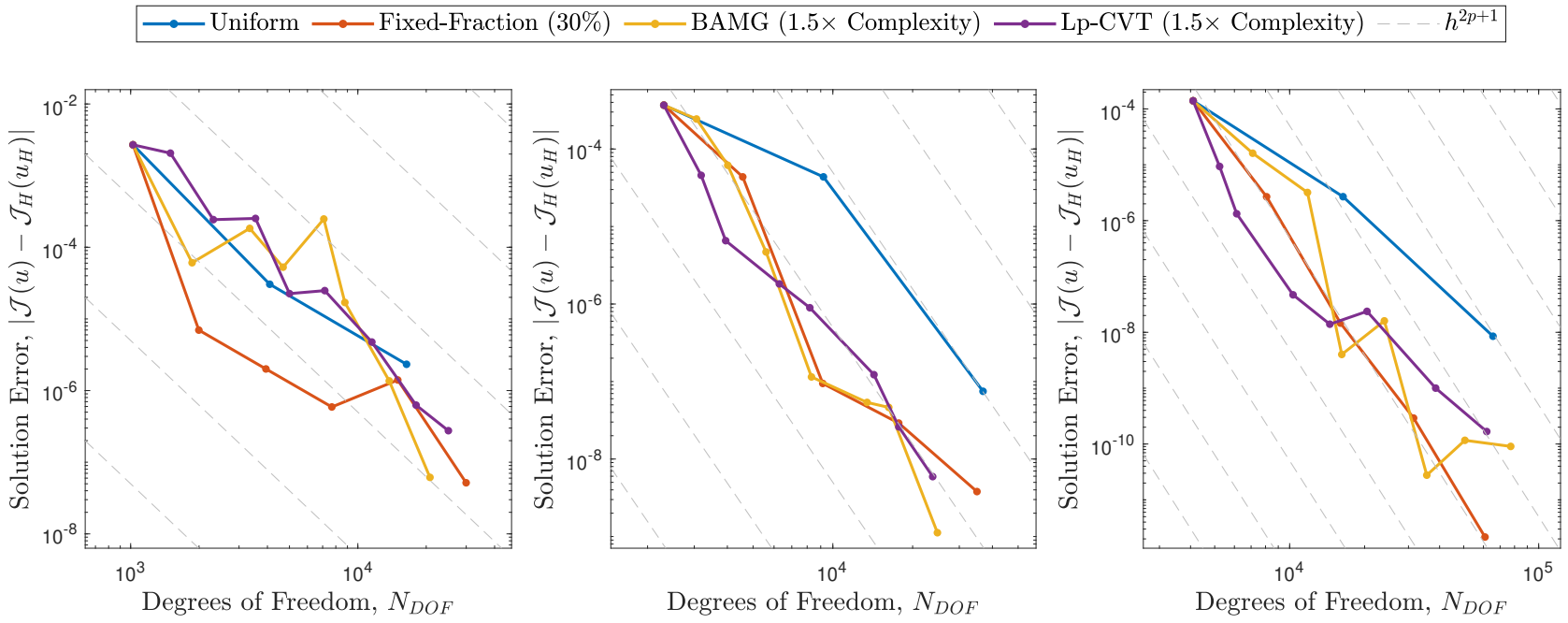
Table 6.3: Comparison of final mesh solution and adjoint problem wall clock times (in seconds) for the goal-oriented S-Shock case.

ranging from 1 to 3, see Fig. 6.22. Note that these convergence plots correspond to the error in the functional (relative to the exact value). Asymptotically, a convergence rate of $\mathcal{O}(h^{2p+1})$ can be achieved for adjoint-consistent schemes (as is the case here, see Section 2.4). This is indicated by the dashed grey lines on the figures. In table 6.3, the time required for solving both the primal (flow) and dual (adjoint) problems are listed. Here, the flow solution time was decreased due to removal of diffusion. Here, in comparison the adjoint started off as the larger fraction time, but, decreased as the scale of problems grew with polynomial order. In this case, this step includes projection of u_H to the fine grid (with enriched polynomial orders) and the subsequent solution of the adjoint problem.

Right away with the $p = 1$ case, the goal-oriented adaptation proved noticeably more challenging for the continuous methods. From here, it appears the fixed-fraction was more stable when working with under-resolved flow features. Only general target areas are needed rather than the complete set of new size targets. This issue is exacerbated by the quadratic fitting to the DWR, resulting in large target size variations between neighbouring cells. As a result, it took multiple iterations before stronger convergence began. However, in the $p = 2$ and $p = 3$ cases where the accuracy of the initial solution is sufficient, the convergence is better behaved with the continuous methods. This is particularly beneficial with the L_p-CVT method at early iterations for these cases. However, later on

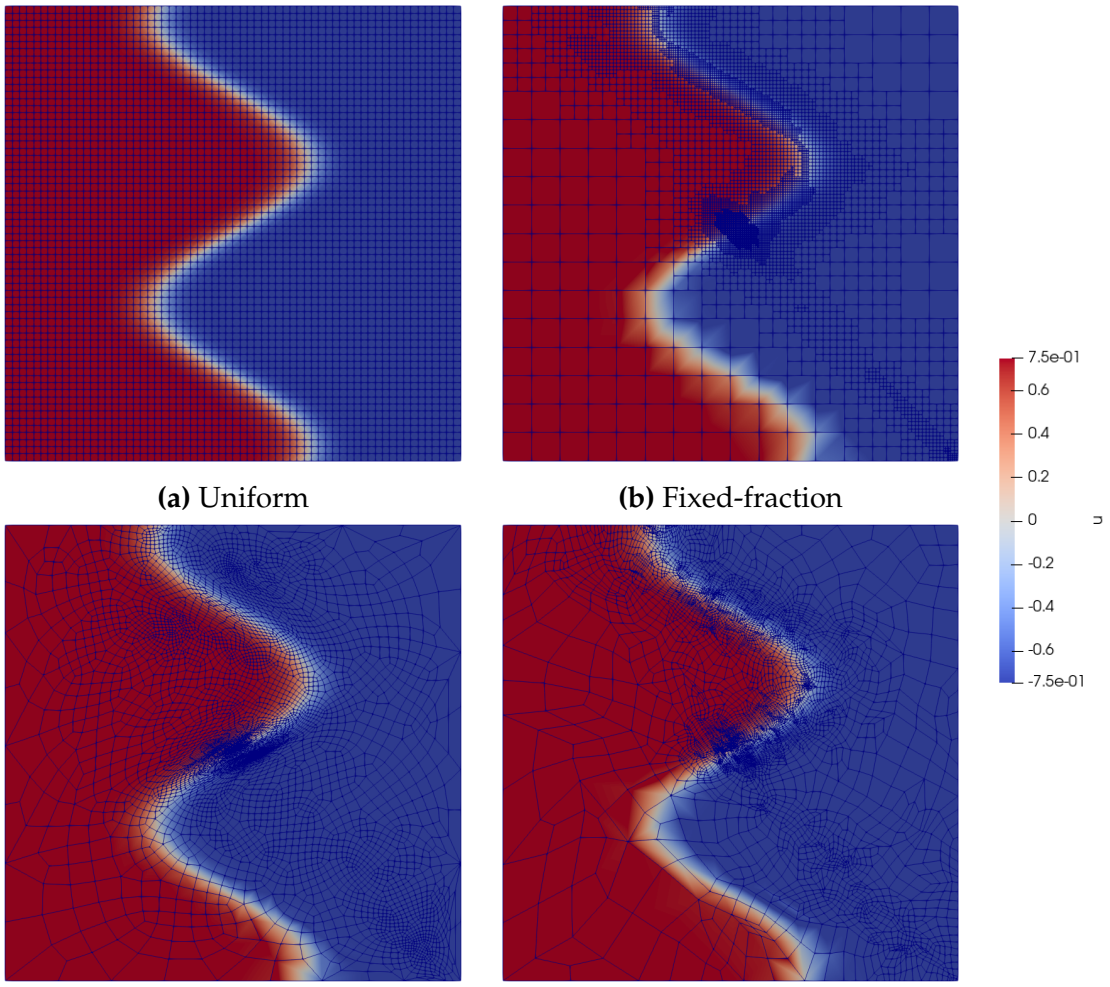
this method suffered due to difficulties meshing in the presence of large discontinuities in the background metric (without the use of any smoothing).

Examining the final mesh produced for the $p = 1$ case in Fig. 6.23, all adaptive methods concentrated refinement in the areas expected (from the initial DWR graph). This was mainly around the upper half of the shock feature corresponding to the areas that will be advected onto the right boundary. In particular, the central shock region was most targeted with a line of refinement leading to the corner. A closeup of this region is shown in Fig. 6.24 for the continuous methods. Here we see the additional smoothness from the BAMG mesh generator and the targeting of the downstream side of the shock. On the other hand, the L_p-CVT method still maintained some alignment of the cells with the shock direction. However, the sharper variations in frame size resulted in more out of place points and affected the quad structure in this region. It also exhibited a more sudden coarsening towards the bottom left away from the area of interest. For $p = 2$ (Figs. 6.25 and 6.26) and $p = 3$ (Figs. 6.27 and 6.28), these variations were smoother, helping to improve the convergence behaviour. Overall, more work is needed to offer consistent benefits over the fixed-fraction method for this case.



(a) $p = 1$ **(b)** $p = 2$ **(c)** $p = 3$

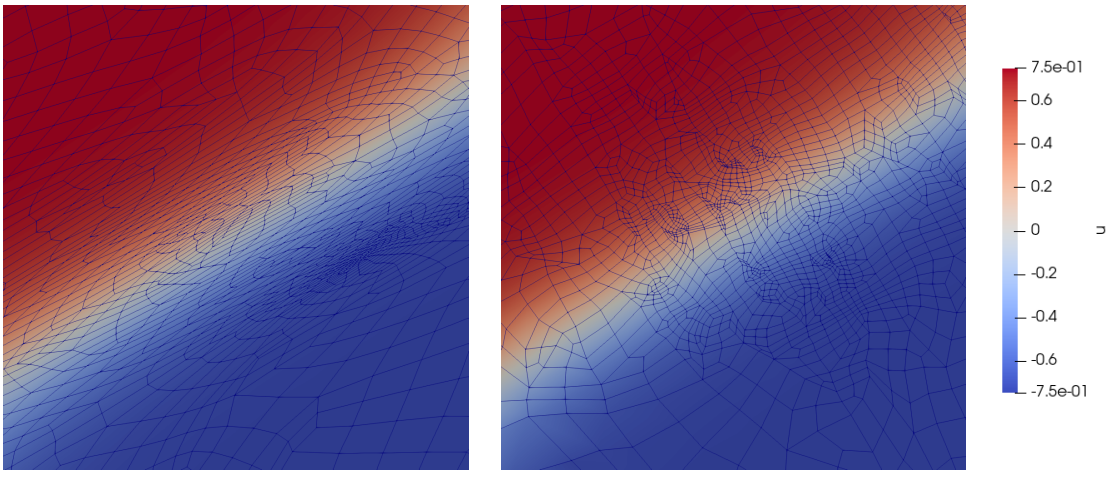
Figure 6.22: Convergence of functional error for the goal-oriented S-Shock case.



(a) Uniform **(b)** Fixed-fraction

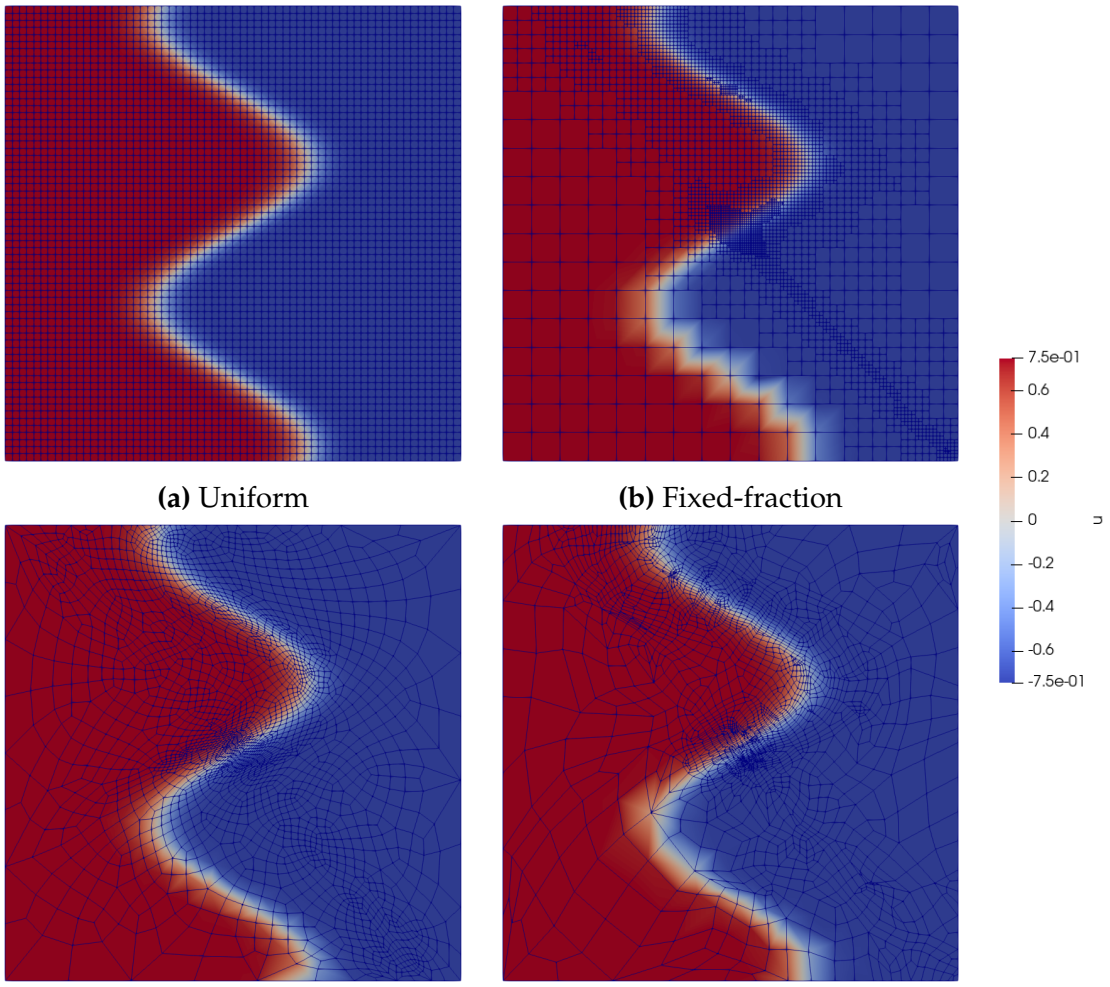
(c) BAMG **(d)** L_p-CVT

Figure 6.23: Final mesh obtained for the goal-oriented S-Shock ($p = 1$).



(a) BAMG **(b)** L_p-CVT

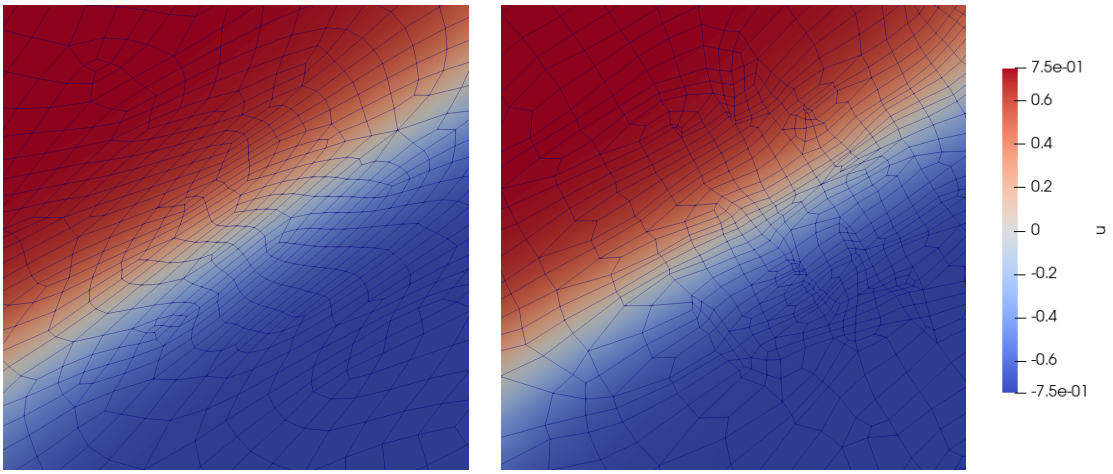
Figure 6.24: Closeup of final mesh obtained for the goal-oriented S-Shock ($p = 1$).



(a) Uniform **(b)** Fixed-fraction

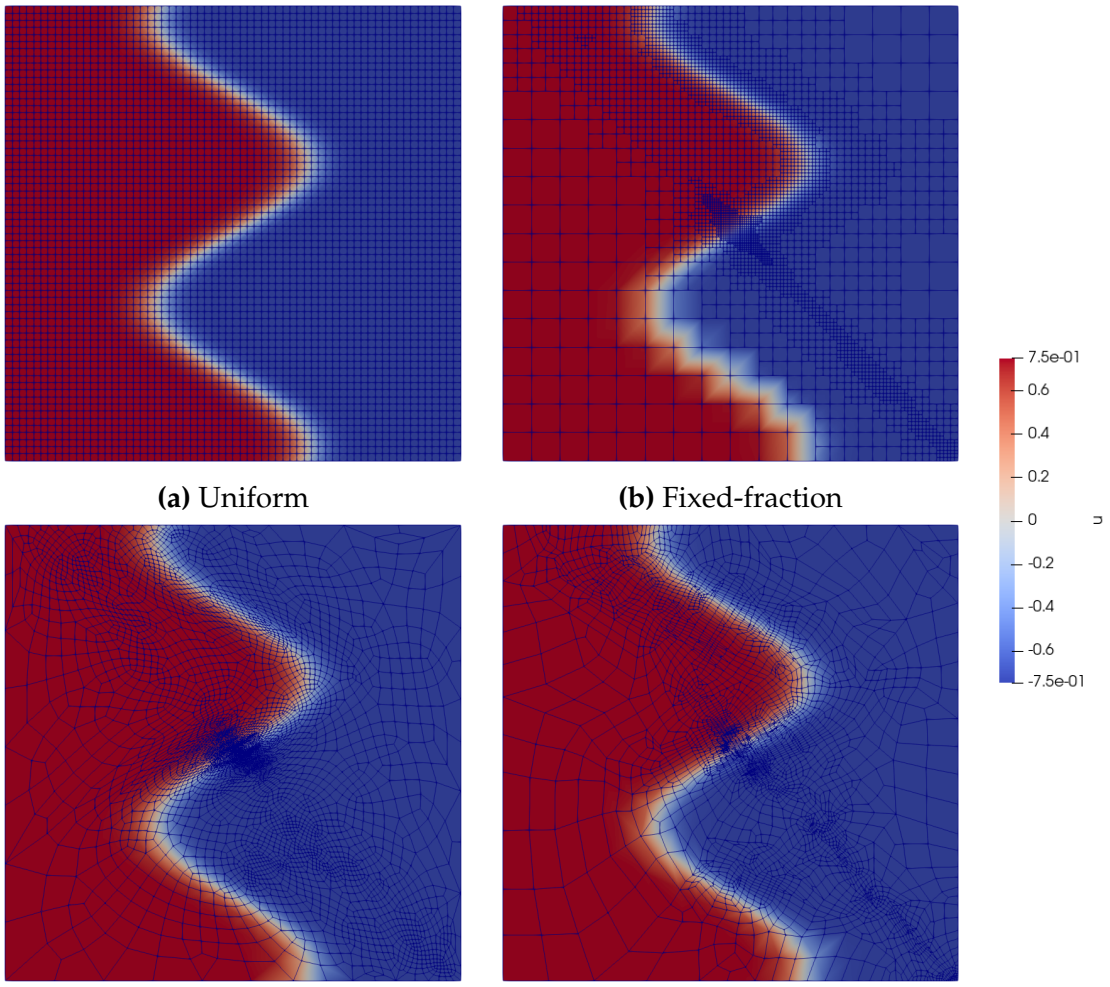
(c) BAMG **(d)** L_p-CVT

Figure 6.25: Final mesh obtained for the goal-oriented S-Shock ($p = 2$).



(a) BAMG **(b)** L_p-CVT

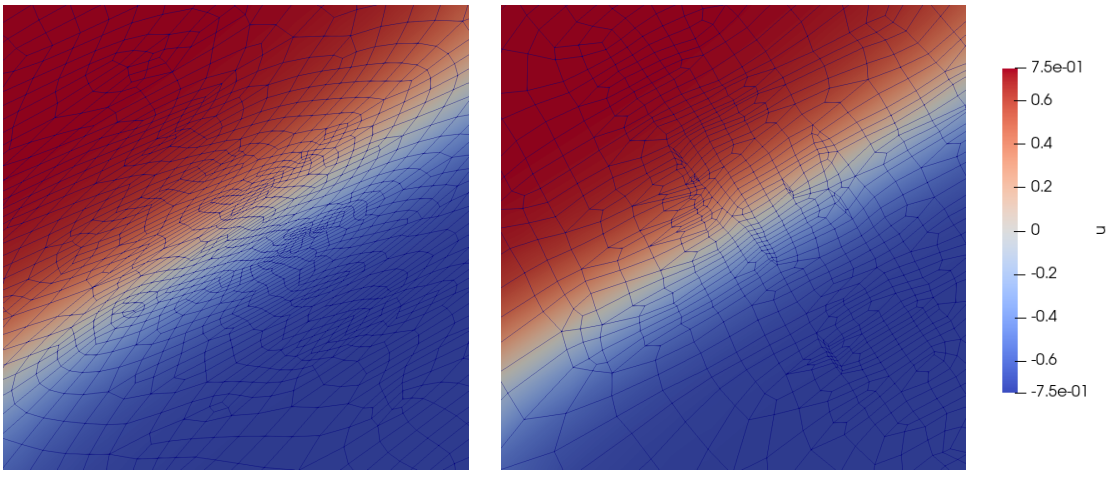
Figure 6.26: Closeup of final mesh obtained for the goal-oriented S-Shock ($p = 2$).



(a) Uniform **(b)** Fixed-fraction

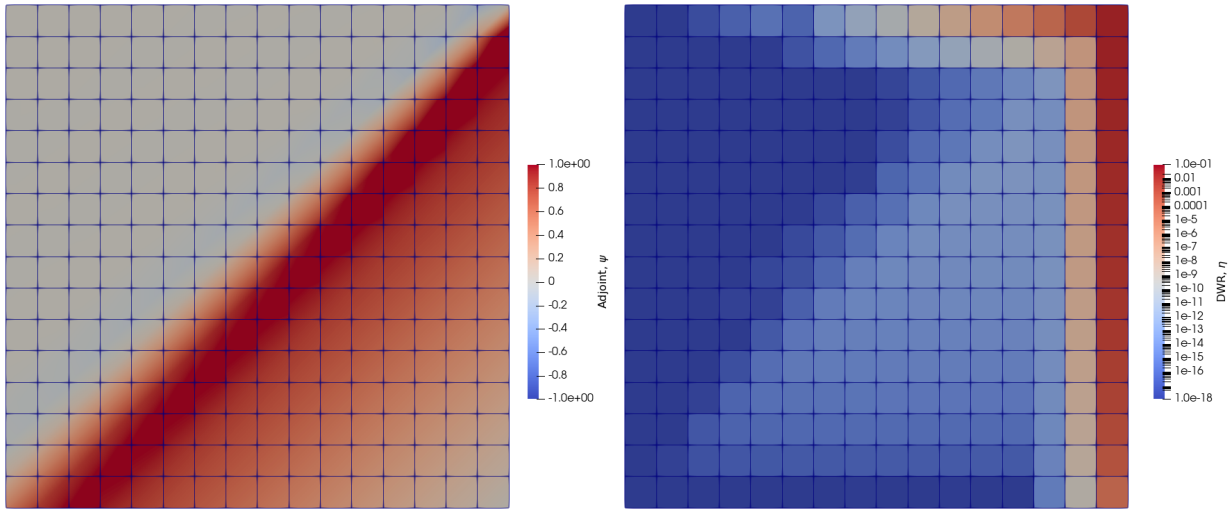
(c) BAMG **(d)** L_p-CVT

Figure 6.27: Final mesh obtained for the goal-oriented S-Shock ($p = 3$).



(a) BAMG **(b)** L_p-CVT

Figure 6.28: Closeup of final mesh obtained for the goal-oriented S-Shock ($p = 3$).



(a) Adjoint, ψ **(b)** DWR, η

Figure 6.29: Initial boundary layer adjoint and logarithmic DWR ($p = 1$).

6.5 Goal-Oriented Boundary Layer Adaptation

Finally, returning to the boundary layer case with a goal-oriented adaptation. Here the same flow constants were used as in Section 6.3 and the previous functional, Eq. 6.6 from Section 6.4. This corresponds to Fig. 6.29, showing the initial adjoint and DWR on a logarithmic scale. Overall, with advection towards the steep boundary layer, the majority of the error is contributed near the edge of the domain in this region. The area leading up to this behaviour is also identified (for $p = 1$), however, at a noticeably lower level. With the 4 methods discussed and varying polynomial orders, the functional convergence plots are shown in Fig. 6.30. Table 6.4 lists the wall-clock time for each step which once again followed similar trends to the previous cases.

Once again, the two continuous methods offered advantages working with this boundary layer case. This shows promise for capturing these features in more detailed and complex problems. Overall, the BAMG method offered the best final mesh in each case. In particular for the $p = 3$ case, it provided nearly a 5 order of magnitude decrease in the error relative to the fixed-fraction. The L_p-CVT started the iterations with strong convergence, especially in the $p = 2$ and $p = 3$ cases where it was initially the best option. How-

		N_{DOF}	$\lvert\mathcal{J}(u) - \mathcal{J}_H(u_H)\rvert$	t (sol.)	t (adj.)
$p = 1$	Fixed-Fraction	1.37×10^4	6.43×10^{-8}	4.32	7.00
	BAMG	1.23×10^4	2.69×10^{-8}	3.92	5.03
	L_p-CVT	1.35×10^4	3.90×10^{-7}	3.35	5.26
$p = 2$	Fixed-Fraction	3.05×10^4	1.96×10^{-10}	22.72	29.35
	BAMG	3.80×10^4	8.41×10^{-12}	30.58	39.18
	L_p-CVT	3.92×10^4	4.81×10^{-10}	29.83	40.00
$p = 3$	Fixed-Fraction	5.54×10^4	5.81×10^{-11}	98.48	96.23
	BAMG	7.46×10^4	7.25×10^{-16}	165.92	90.99
	L_p-CVT	4.74×10^4	2.28×10^{-12}	88.56	86.96

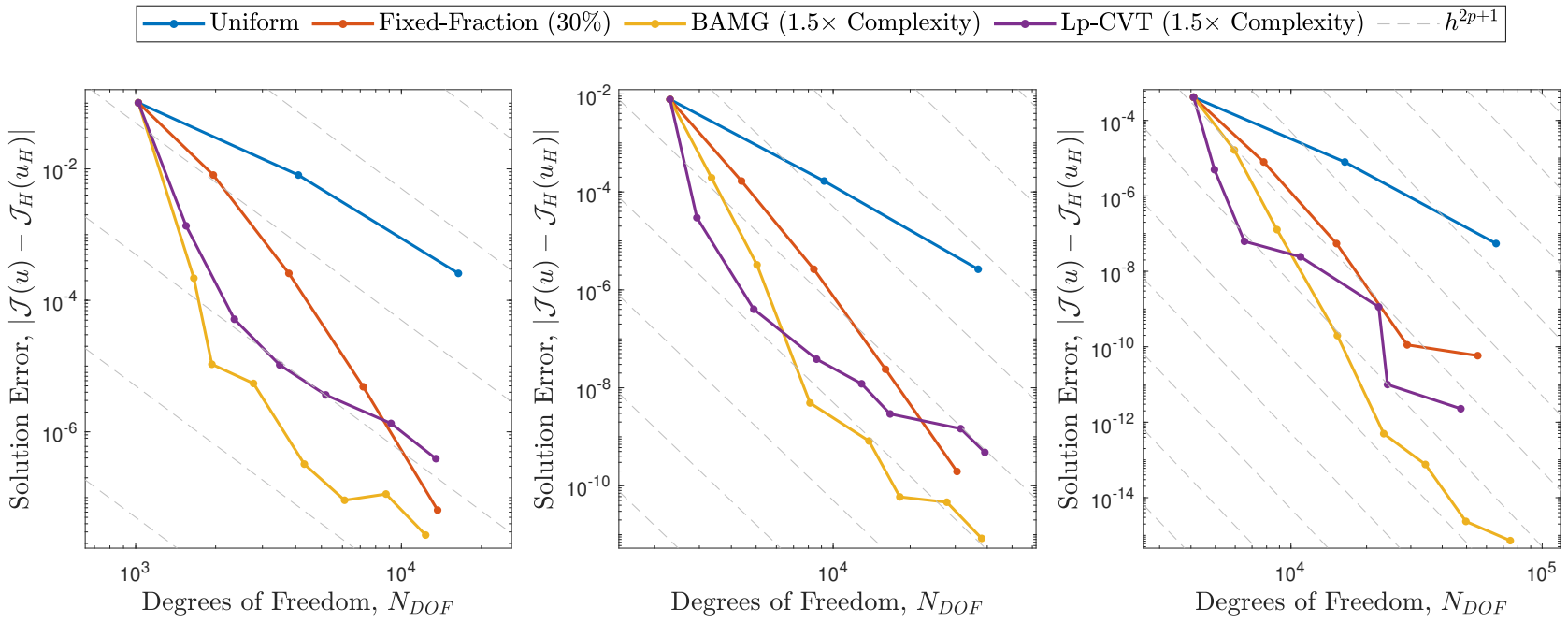
Table 6.4: Comparison of final mesh solution and adjoint problem wall clock times (in seconds) for the goal-oriented boundary layer case.

ever, towards later iterations, it began to show a similar stalling effect observed for the $p = 3$ feature-based case. This is due to a combination of the limitation in the anisotropy and difficulties conforming to rapid variations in the size field. More work will be needed to fully address these problems.

Examining the resulting mesh for the $p = 2$ case in Fig. 6.33. There is a clear targeting of the right hand side as expected. The continuous methods benefit from the introduction of anisotropy in this area and their ability to iteratively remesh the domain as more information becomes available. However, as the goal-oriented error estimates only provide iterative size field updates and do not embed underlying information about the exact behaviour. These changes must be reflected over several steps and they are more susceptible to oscillation. This also lead to a noticeable coarsening of the upper edge which is irrelevant to the functional despite showing strong solution behaviour. From the closeup of the upper-right corner in Fig. 6.34, we can see the boundary resolution achieved by the continuous methods. Here the L_p-CVT method maintained good alignment, but with less anisotropy compared to the corresponding feature-based mesh. On the other hand, while lacking this alignment the BAMG mesh generator exhibited smooth transitions from the anisotropic region with better refinement along the diagonal line leading to the upper right corner. Here, the smooth transition is also seen in the full mesh plot and resulted in

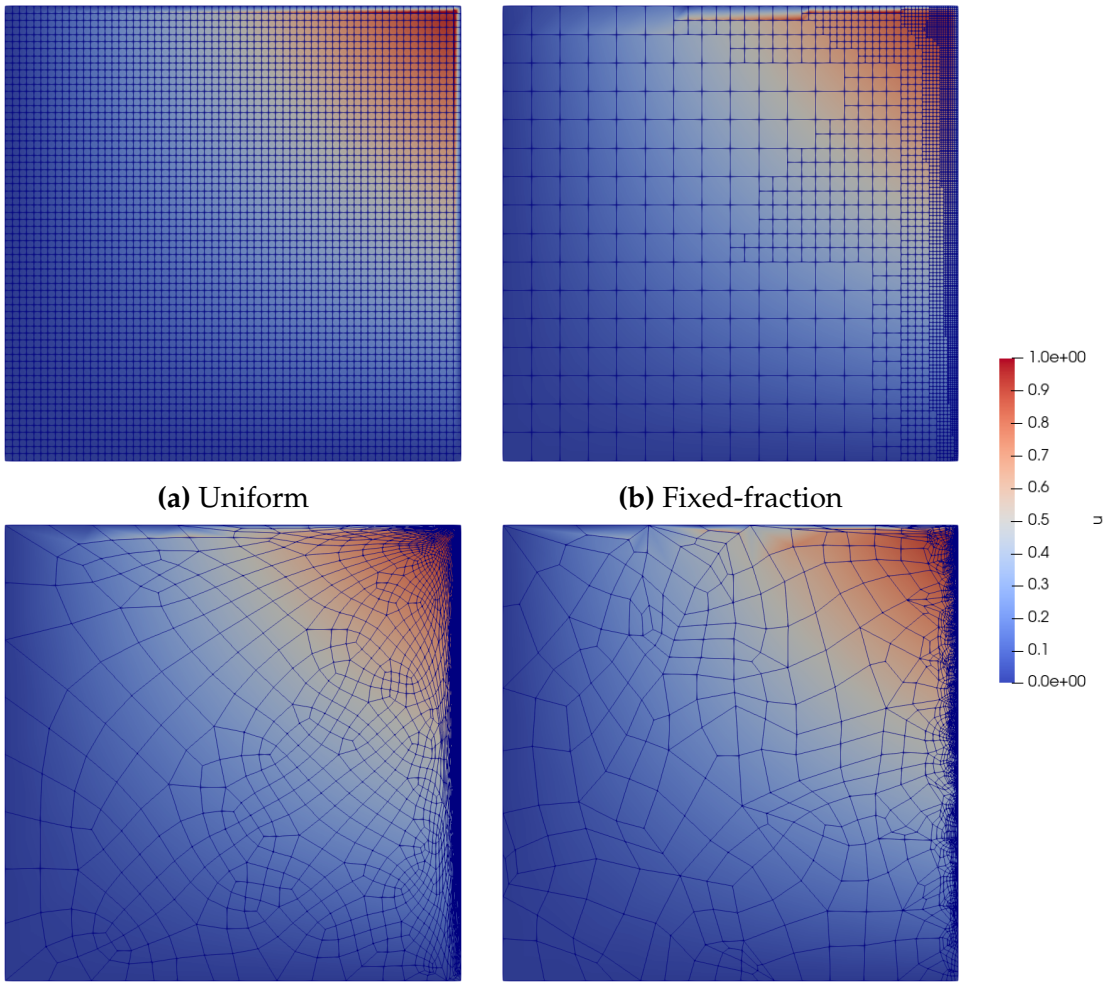
a well-behaved mesh in the intermediate area. This difference highlights the importance of metric smoothing when working with this choice of goal-oriented error estimate.

For the $p = 1$ case (Figs. 6.31 and 6.32), similar trends were seen along the boundary. However, only the BAMG method was able to produce isotropic refinement in the relevant half of the quadratic region. Finally, for the $p = 3$ case (6.35 and 6.36), trends largely match the $p = 2$ case. Mainly the goal-oriented adaptation resulted in more isotropic cells along the boundary when compared with the previously examined feature-based case. As the choice of anisotropy is still governed by the primal solution, these effects result from the size variations in DWR based updates. Resolving this limitation would help to capture the full benefits of the unstructured mesh adaptation procedure.



(a) $p = 1$ **(b)** $p = 2$ **(c)** $p = 3$

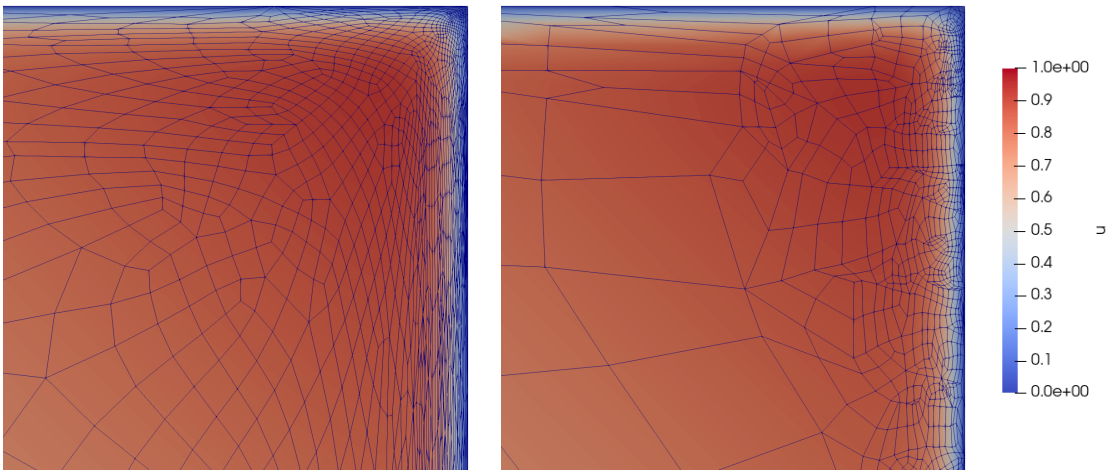
Figure 6.30: Convergence of functional error for the goal-oriented boundary layer case.



(a) Uniform **(b)** Fixed-fraction

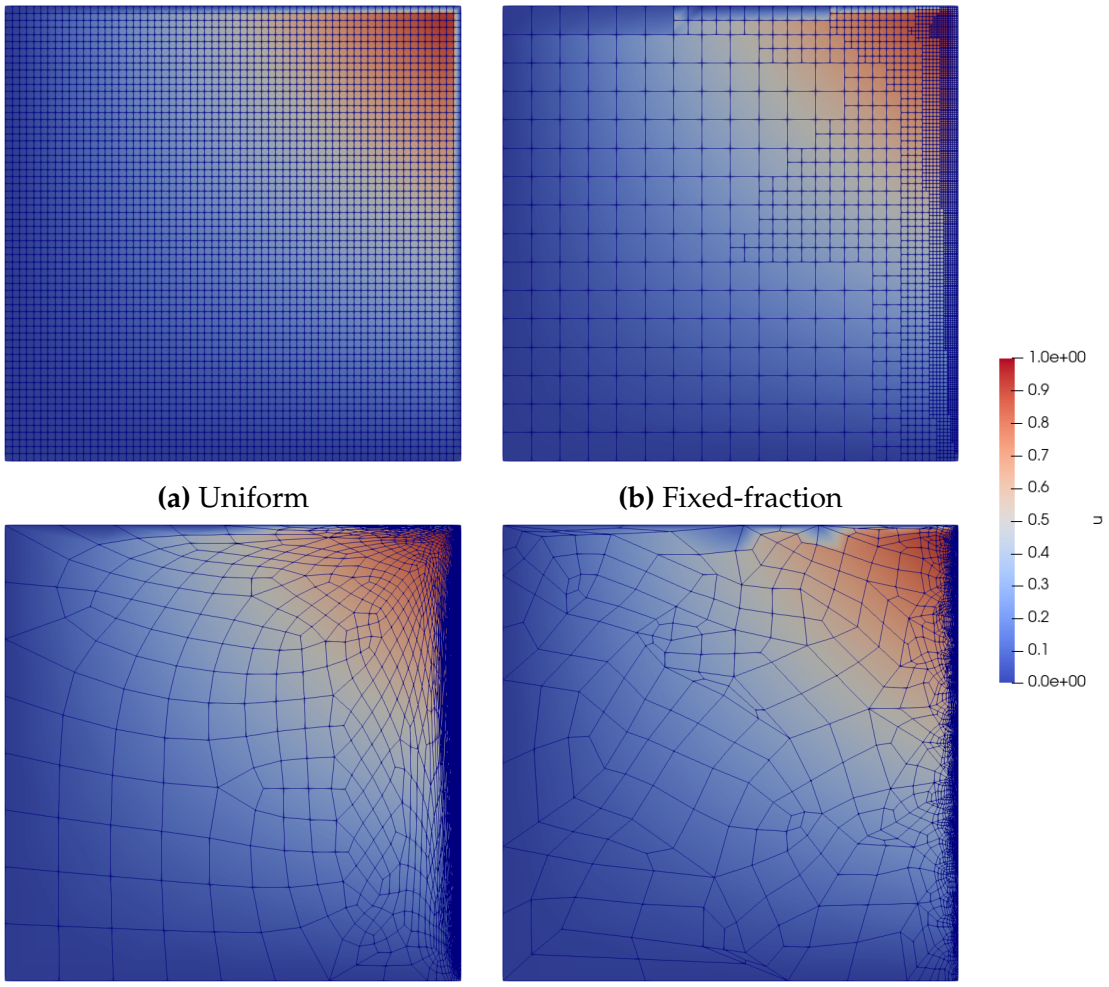
(c) BAMG **(d)** L_p-CVT

Figure 6.31: Final mesh obtained for the goal-oriented boundary layer ($p = 1$).



(a) BAMG **(b)** L_p-CVT

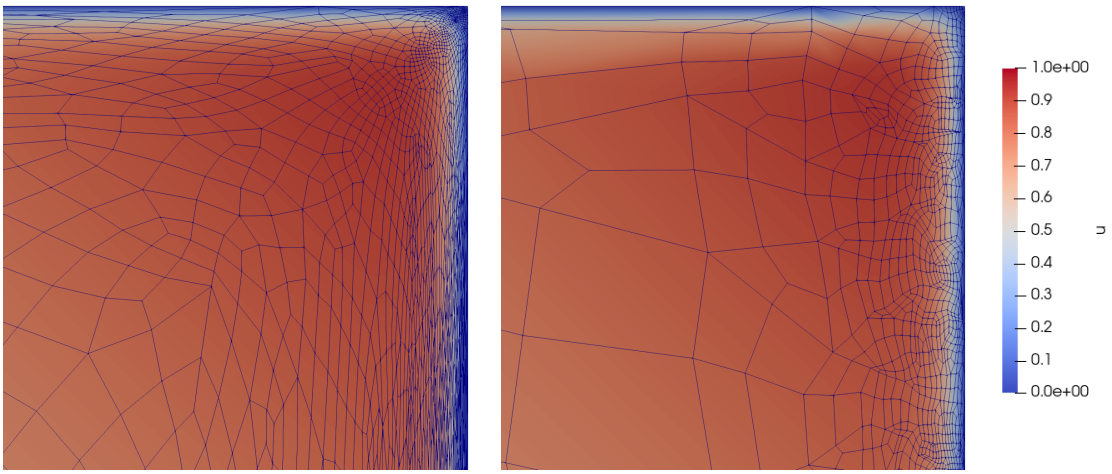
Figure 6.32: Closeup of final mesh obtained for the goal-oriented boundary layer ($p = 1$).



(a) Uniform **(b)** Fixed-fraction

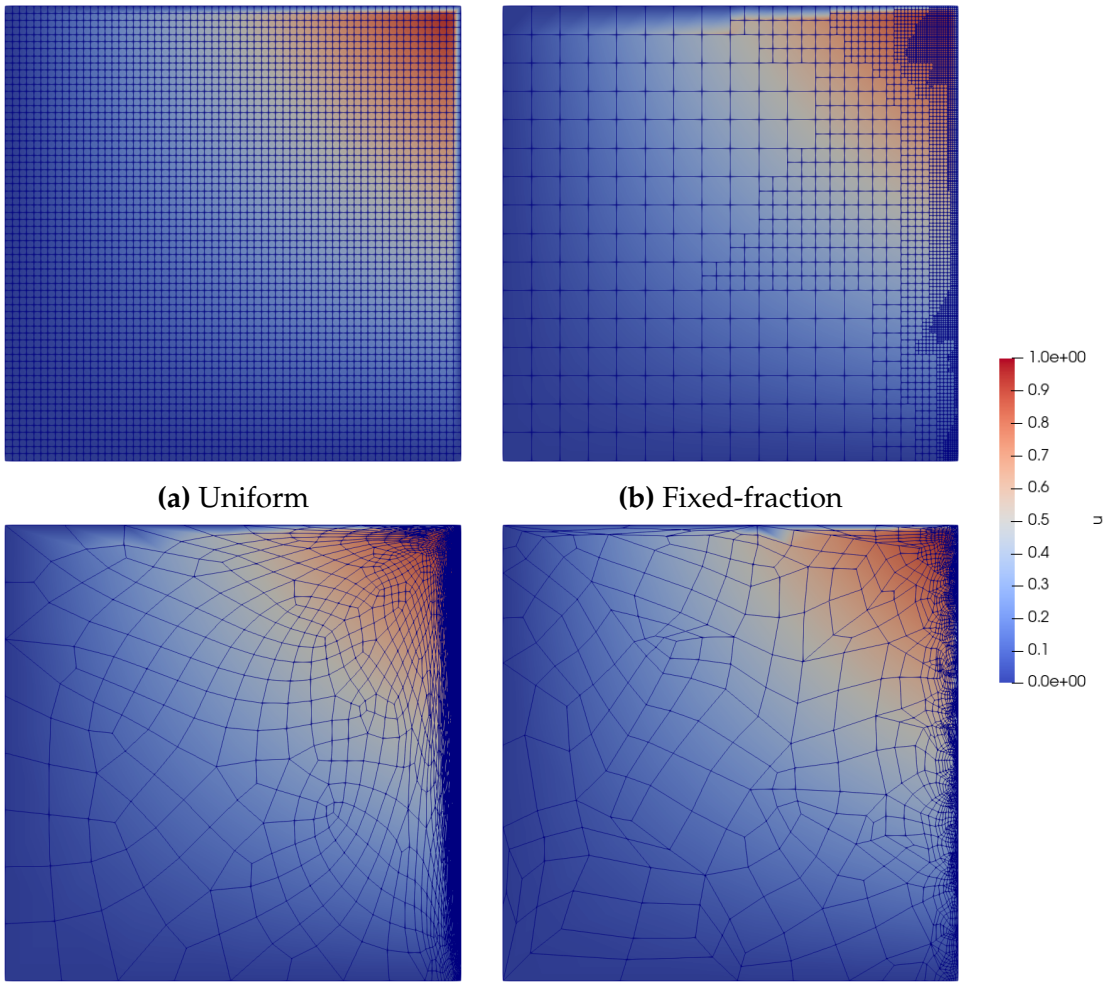
(c) BAMG **(d)** L_p-CVT

Figure 6.33: Final mesh obtained for the goal-oriented boundary layer ($p = 2$).



(a) BAMG **(b)** L_p-CVT

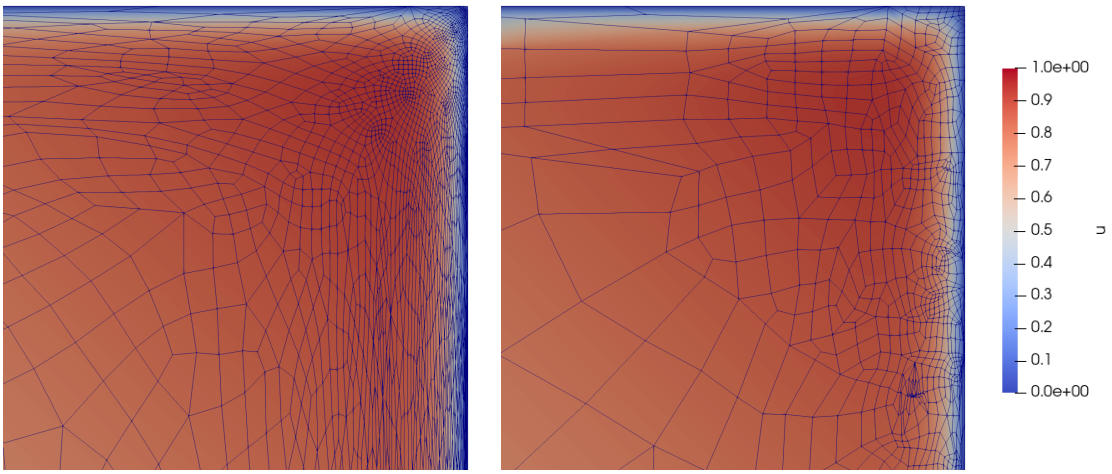
Figure 6.34: Closeup of final mesh obtained for the goal-oriented boundary layer ($p = 2$).



(a) Uniform **(b)** Fixed-fraction

(c) BAMG **(d)** L_p-CVT

Figure 6.35: Final mesh obtained for the goal-oriented boundary layer ($p = 3$).



(a) BAMG **(b)** L_p-CVT

Figure 6.36: Closeup of final mesh obtained for the goal-oriented boundary layer ($p = 3$).

Chapter 7

Conclusions

In conclusion, this thesis has introduced an all-quad unstructured mesh adaptation framework suitable for use with high-order methods such as the Discontinuous Galerkin method. At its core, this involved the introduction of the continuous mesh and the continuous error models. The approach relied on the use of a frame field to define the target element orientation, anisotropy and size for the discrete mesh as continuous quantities. As a result, ideas originating from anisotropic triangular mesh adaptation under the influence of a Riemannian metric space can be directly extended to the quad meshing case with additional control over the cell orientations. This allowed us to restate the discrete error minimization problem in a way that could be directly solved by calculus of variations. The minimization targeted a globally optimal distribution of degrees of freedom tailored to the discrete solution obtained from the flow solver. The evaluation of these estimates was integrated in our in-house solver, PHiLiP. This also provides the added flexibility of using these targets with other adaptive techniques in the future.

Additionally, by incorporating the dual-weighted residual (DWR), we were able to consider extensions to goal-oriented adaptation problems. These are particularly useful in CFD applications as they directly and effectively target improvements in the chosen functional. In the present work, this was achieved through iterative updates to the size

field based on the DWR distribution in the logarithmic space. Therefore, allowing the effects to be integrated with our anisotropic target cell shapes from the flow solution.

Next, the work also considered the design and implementation of an L_p-CVT mesh generator capable of conforming to the target frame field derived from the aforementioned error estimates. Here, the mesh nodes were distributed based on an energy minimization procedure in the L_p anisotropic space. For this work, the anisotropic metric resulted from the inverse linear transformation associated with the discrete frame field. Insertion and removal of nodes was controlled through a valid edge length criteria. Finally, a series of post-processing steps were undergone to triangulate, merge and output an all-quad mesh suitable for use with the flow solver on subsequent iterations. This was developed as a standalone code, allowing the individual components of this project to be used in conjunction or independently.

Finally, through the series of test cases examined, this work showed great promise for the future of the methods. In the feature-based cases, the continuous estimates offered significant reduction in error compared to a more traditional fixed-fraction approach. This was partially due to the ability of these methods to completely remesh the domain as new information becomes available. Additionally, in the case of the L_p-CVT mesh generator, anisotropy and good alignment were observed with the flow features. For the goal-oriented adaptation, the behaviour was more case dependent. From a meshing perspective, these estimates introduced additional challenges as they incorporate localized effects and resulted in sharper variations in the mesh gradation than previously seen. For the boundary layer case, the use of the continuous estimates offered notable improvements when coupled with the BAMG mesh generator. However, based on the strong convergence observed during the first iterations, the results in other cases may also further benefit from these methods once the current limitations are addressed.

Overall, the current work provides a foundation for further in-depth studies of all-quad mesh generation and adaptation techniques. These methods have shown great potential to be extended to more challenging problems where they may aid in the under-

standing of complex flow phenomenon. Therefore, the contributions of this work have served as a stepping stone to facilitate future progress in hopes of one day incorporating highly accurate and automated simulations in the CFD workflow.

7.1 Future Work

In addition to the results and developments achieved, this work helped to highlight future directions for progression of these methods. With many of the current limitations, there are clear paths that may address them and improve the overall robustness. Here we consider logical next steps for the mesh generator and continuous error estimation procedures.

7.1.1 L_p-CVT Mesh Generator

Improve output anisotropy - One of the main limitations was the difficulty in conforming to highly anisotropic meshing targets. As mentioned in [74], the energy functional becomes ill-conditioned in these regions. This issue is further exacerbated through the discontinuities in the background metric function rendering the analytic gradients inaccurate. The Voronoi cell shapes in these regions are highly sensitive to perturbations in the node positions. Some frame field methods have achieved this behaviour by performing isotropic meshing on a deformed surface [99]. Other possible solutions specific to CVT methods involve modifying the local polygon for the energy integration (replacing the Voronoi cells). To name a few possibilities, this has been studied through the Anisotropic Centroidal Voronoi Tessellation (ACVT) [37] or by introduction of the Bregman diagram for the Optimal Voronoi Tessellation (OVT) [19].

Extend clipping procedure to non-convex geometries - Next, one of the major restrictions is that the current code works only on convex domains. This is due to the simplified form of the clipping procedure that was used. However, extensions to the standard CVT have

already been proposed which resolve this issue and could also be applied to our case [132]. As the DG method and solver are already capable of handling these configurations, this change would allow us to study more geometrically complex cases.

Incorporate metric field smoothing - The behaviour of the metric field could be improved both globally and cell-wise. First, globally a metric smoothing or edge length smoothing could be used to prevent areas of high mesh gradation that are problematic for the output mesh. Second, locally an interpolation scheme could be used to reconstruct the metric behaviour and reinstate the terms lost in the analytic gradients. For our case, we have the additional challenge that the matrix representation is non-symmetric. Some possible approaches have been proposed in [41] such as linear, exponential (from Pennec et al. [102]) and spline based interpolations for the symmetric case, however, inspiration could also be drawn from frame field interpolation methods [123].

Various speed improvements - Finally, in comparison to more traditional mesh generators, the computational cost and speed of this approach are major bottlenecks. Some ideas to reduce this effect include modifying the point location data-structure to use a traversal search [132] or starting from an anisotropic triangulation of the domain to circumvent the costly additional splitting cycles (as used in the isotropic case [10]).

7.1.2 Continuous Error Estimation

Study the mesh adaptation of more complex cases - Assuming suitable adjustments to the mesh generator have been made (or extending usage with BAMG), more complex geometries should be studied (Gaussian Bump, airfoils, etc.). Additionally, we should also consider flow problems involving several variables, such as with the Euler equations.

Incorporate *hp* adaptivity - The method proposed for hp adaptivity decisions by Dolejsi et al. [35] can serve as a direct extension to our quad-meshing case. However, first,

additional implementation and validation of the solver under these conditions will be required. This allows the benefits of large cells with p refinement in smooth areas and small low-order cells with h refinement near discontinuities.

Consider alternative anisotropic quad generation techniques - The current developments are not bound by the particular choice of mesh generator. Thus far, the L_p-CVT method has provided a convenient means for studying anisotropic quadrangulations. However, several promising methods are currently under development and should also be considered going forward, particularly for speed and practicality [15, 21, 88, 99].

Alternative goal-oriented adaptation - The current goal-oriented adaptation procedure is heavily dependent on previous mesh iterations. Additionally, it introduced challenging variations in the size field. However, some triangular mesh generation methods avoid this by directly incorporating the adjoint in the goal-oriented metric definition (e.g. [82] in the $p = 1$ case). As the adjoint will also converge to a physical value during the iterations, these continuous targets approach a consistent distribution (up to the choice of scaling parameter needed to match the complexity target).

References

[1] ALAUZET, F., AND LOSEILLE, A. A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. *Computer-Aided Design 72* (March 2016), 13–39.

[2] ANDERSON, B. D., BENZLEY, S. E., AND OWEN, S. J. Automatic All Quadrilateral Mesh Adaption through Refinement and Coarsening. In *Proceedings of the 18th International Meshing Roundtable*. Springer Berlin Heidelberg, Berlin, Heidelberg, January 2009, pp. 557–574.

[3] ANDERSON, J. D., AND WENDT, J. *Computational fluid dynamics*. Springer, 1995.

[4] ARNDT, D., BANGERTH, W., BLAIS, B., CLEVENGER, T. C., FEHLING, M., GRAYVER, A. V., HEISTER, T., HELTAI, L., KRONBICHLER, M., MAIER, M., MUNCH, P., PELTERET, J.-P., RASTAK, R., THOMAS, I., TURCKSIN, B., WANG, Z., AND WELLS, D. The `deal.II` library, version 9.2. *Journal of Numerical Mathematics 28*, 3 (2020), 131–146.

[5] BAKER, T. J. Automatic mesh generation for complex three-dimensional regions using a constrained Delaunay triangulation. *Engineering with Computers 5*, 3-4 (June 1989), 161–175.

[6] BALAN, A., WOOPEN, M., AND MAY, G. Adjoint-based *hp*-adaptivity on anisotropic meshes for high-order compressible flow simulations. *Computers and Fluids 139* (2016).

[7] BARTOŠ, O., DOLEJŠÍ, V., MAY, G., RANGARAJAN, A., AND ROSKOVEC, F. A goal-oriented anisotropic *hp*-mesh adaptation method for linear convection–diffusion–reaction problems. *Computers & Mathematics with Applications 78*, 9 (2019), 2973 – 2993. Applications of Partial Differential Equations in Science and Engineering.

[8] BASSI, F., AND REBAY, S. A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier–Stokes Equations. *Journal of Computational Physics 131*, 2 (March 1997), 267–279.

[9] BAUDOUIN, T., REMACLE, J.-F., MARCHANDISE, E., HENROTTE, F., AND GEUZAINE, C. A frontal approach to hex-dominant mesh generation. *Advanced Modeling and Simulation in Engineering Sciences 1*, 1 (2014), 8.

[10] BAUDOUIN, T. C., REMACLE, J.-F., MARCHANDISE, E., LAMBRECHTS, J., AND HENROTTE, F. Lloyd's energy minimization in the L_p norm for quadrilateral surface mesh generation. *Engineering with Computers 30*, 1 (2014), 97–110.

[11] BELME, A., DERVIEUX, A., AND ALAUZET, F. Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows. *Journal of Computational Physics 231*, 19 (August 2012), 6323–6348.

[12] BERGER, M. *A Panoramic View of Riemannian Geometry*, vol. 7. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[13] BLACKER, T. D., AND STEPHENSON, M. B. Paving: A new approach to automated quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering 32*, 4 (September 1991), 811–847.

[14] BLAZEK, J. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[15] BOMMES, D., LÉVY, B., PIETRONI, N., PUPPO, E., SILVA, C., TARINI, M., AND ZORIN, D. Quad-Mesh Generation and Processing: A Survey. *Computer Graphics Forum 32*, 6 (September 2013), 51–76.

[16] BOMMES, D., ZIMMER, H., AND KOBBELT, L. Mixed-integer quadrangulation. In *ACM SIGGRAPH 2009 papers on - SIGGRAPH '09* (New York, New York, USA, 2009), ACM Press, p. 1.

[17] BOROUCHAKI, H., AND FREY, P. J. Adaptive triangular-quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering 41*, 5 (March 1998), 915–934.

[18] BOWYER, A. Computing Dirichlet tessellations. *The Computer Journal 24*, 2 (February 1981), 162–166.

[19] BUDNINSKIY, M., LIU, B., DE GOES, F., TONG, Y., ALLIEZ, P., AND DESBRUN, M. Optimal Voronoi Tessellations with hessian-based anisotropy. *ACM Transactions on Graphics 35*, 6 (November 2016), 1–12.

[20] BURSTEDDE, C., WILCOX, L. C., AND GHATTAS, O. `p4est`: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing 33*, 3 (2011), 1103–1133.

[21] CAMPEN, M., BOMMES, D., AND KOBBELT, L. Quantized global parametrization. In *ACM Transactions on Graphics* (November 2015), vol. 34, Association for Computing Machinery.

[22] CAO, W. An Interpolation Error Estimate on Anisotropic Meshes in $\mathcal{R}^n$ and Optimal Metrics for Mesh Refinement. *SIAM Journal on Numerical Analysis 45*, 6 (January 2007), 2368–2391.

[23] CAO, W. An interpolation error estimate in $\mathcal{R}^2$ based on the anisotropic measures of higher order derivatives. *Mathematics of Computation 77*, 261 (January 2008), 265–286.

[24] CASTILLO, P. E. Stencil reduction algorithms for the local discontinuous Galerkin method. *International Journal for Numerical Methods in Engineering* (February 2009).

[25] CASTRO-DÍAZ, M. J., HECHT, F., MOHAMMADI, B., AND PIRONNEAU, O. Anisotropic unstructured mesh adaption for flow simulations. *International Journal for Numerical Methods in Fluids 25*, 4 (August 1997), 475–491.

[26] CEZE, M., AND FIDKOWSKI, K. J. Anisotropic *hp*-Adaptation Framework for Functional Prediction. *AIAA Journal 51*, 2 (February 2013), 492–509.

[27] CHEN, L., AND XU, J.-C. OPTIMAL DELAUNAY TRIANGULATIONS. *Journal of Computational Mathematics 22*, 2 (2004), 299–308.

[28] CHEW, L. P. Constrained Delaunay triangulations. In *Proceedings of the third annual symposium on Computational geometry - SCG '87* (New York, New York, USA, 1987), ACM Press, pp. 215–222.

[29] COULAUD, O., AND LOSEILLE, A. Very High Order Anisotropic Metric-Based Mesh Adaptation in 3D. *Procedia Engineering 163* (2016), 353–365.

[30] DELAUNAY, B. Sur la sphere vide. *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques 7*, 793-800 (1934), 1–2.

[31] DHAUBHADEL, M. N. Review: CFD Applications in the Automotive Industry. *Journal of Fluids Engineering 118*, 4 (December 1996), 647–653.

[32] DIAMANTI, O., VAXMAN, A., PANOZZO, D., AND SORKINE-HORNUNG, O. Designing N-polyvector fields with complex polynomials. *Eurographics Symposium on Geometry Processing 33*, 5 (2014), 1–11.

[33] DOBRZYNSKI, C., AND FREY, P. Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations. In *Proceedings of the 17th International Meshing Roundtable*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 177–194.

[34] DOLEJŠÍ, V. Anisotropic *hp*-adaptive method based on interpolation error estimates in the Lq-norm. *Applied Numerical Mathematics 82* (August 2014), 80–114.

[35] DOLEJŠÍ, V., MAY, G., AND RANGARAJAN, A. A continuous *hp*-mesh model for adaptive discontinuous Galerkin schemes. *Applied Numerical Mathematics 124* (February 2018), 1–21.

[36] DU, Q., FABER, V., AND GUNZBURGER, M. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM review 41*, 4 (1999), 637–676.

[37] DU, Q., AND WANG, D. Anisotropic Centroidal Voronoi Tessellations and Their Applications. *SIAM Journal on Scientific Computing 26*, 3 (January 2005), 737–761.

[38] EDMONDS, J. Paths, Trees, and Flowers. *Canadian Journal of Mathematics 17* (November 1965), 449–467.

[39] EKATERINARIS, J. A. High-order accurate, low numerical diffusion methods for aerodynamics. *Progress in Aerospace Sciences 41*, 3-4 (April 2005), 192–300.

[40] EKELSCHOT, D., CEZE, M., GARAI, A., AND MURMAN, S. M. Robust metric-aligned quad-dominant meshing using Lp centroidal Voronoi Tessellation. *AIAA Aerospace Sciences Meeting, 2018*, 210059 (2018).

[41] EKELSCHOT, D., CEZE, M., GARAI, A., AND MURMAN, S. M. Parallel high-order anisotropic meshing using discrete metric tensors. *AIAA Scitech 2019 Forum* (January 2019).

[42] FENG, L., ALLIEZ, P., BUSÉ, L., DELINGETTE, H., AND DESBRUN, M. Curved optimal delaunay triangulation. *ACM Transactions on Graphics 37*, 4 (August 2018), 1–16.

[43] FIDKOWSKI, K. J., AND DARMOFAL, D. L. Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics. *AIAA Journal 49*, 4 (2011), 673–694.

[44] FORTUNE, S. A sweepline algorithm for Voronoi diagrams. *Algorithmica 2*, 1-4 (November 1987), 153–174.

[45] FREY, P. Yams a fully automatic adaptive isotropic surface remeshing procedure.

[46] FREY, P. J., AND GEORGE, P.-L. *Mesh Generation: Application to Finite Elements*. 2008.

[47] FUJII, K. Progress and future prospects of CFD in aerospace—Wind tunnel and beyond. *Progress in Aerospace Sciences 41*, 6 (August 2005), 455–470.

[48] GEORGE, P. Gamanic3d, adaptive anisotropic tetrahedral mesh generator. Tech. rep., Technical Report, INRIA, 2002.

[49] GEORGE, P., HECHT, F., AND VALLET, M. Creation of internal points in Voronoi's type method. Control adaptation. *Advances in Engineering Software and Workstations 13*, 5-6 (September 1991), 303–312.

[50] GEUZAINE, C., AND REMACLE, J.-F. Gmsh 4.7.1 manual.

[51] GEUZAINE, C., AND REMACLE, J. F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering 79*, 11 (2009), 1309–1331.

[52] GILES, M., AND PIERCE, N. Improved lift and drag estimates using adjoint Euler equations. In *14th Computational Fluid Dynamics Conference* (Reston, Virigina, November 1999), American Institute of Aeronautics and Astronautics, pp. 369–380.

[53] GILES, M. B. On Adjoint Equations for Error Analysis and Optimal Grid Adaptation in CFD. In *Frontiers of Computational Fluid Dynamics 1998*, no. 97. WORLD SCIENTIFIC, November 1998, pp. 155–169.

[54] GREEN, P. J., AND SIBSON, R. Computing Dirichlet Tessellations in the Plane. *The Computer Journal 21*, 2 (May 1978), 168–173.

[55] HARRIS, N. J., BENZLEY, S., AND OWEN, S. Conformal refinement of all-hexahedral element meshes based on multiple twist plane insertion. In *IMR* (2004).

[56] HARTMANN, R., AND HOUSTON, P. Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations. *Journal of Computational Physics 183*, 2 (December 2002), 508–532.

[57] HATELEY, J. C., WEI, H., AND CHEN, L. Fast Methods for Computing Centroidal Voronoi Tessellations. *Journal of Scientific Computing 63*, 1 (April 2015), 185–212.

[58] HECHT, F. BAMG: Bidimensional Anisotropic Mesh Generator. *User Guide. INRIA, Rocquencourt 17* (1998).

[59] HECHT, F., AND KUATE, R. An approximation of anisotropic metrics from higher order interpolation error for triangular mesh adaptation. *Journal of Computational and Applied Mathematics 258* (March 2014), 99–115.

[60] HESTHAVEN, J. S., AND WARBURTON, T. *Nodal Discontinuous Galerkin Methods*, vol. 54 of *Texts in Applied Mathematics*. Springer New York, New York, NY, 2008.

[61] HÖHNE, T., KREPPER, E., AND ROHDE, U. Application of CFD codes in nuclear reactor safety analysis. *Science and Technology of Nuclear Installations 2010* (2010).

[62] HOUSTON, P., SCHWAB, C., AND SÜLI, E. Discontinuous *hp*- Finite Element Methods for Advection-Diffusion-Reaction Problems. *SIAM Journal on Numerical Analysis 39*, 6 (January 2002), 2133–2163.

[63] HUANG, J., JIANG, T., WANG, Y., TONG, Y., AND BAO, H. Automatic Frame Field Guided Hexahedral Mesh Generation Technique Report. Tech. rep.

[64] HUANG, J., TONG, Y., WEI, H., AND BAO, H. Boundary aligned smooth 3D cross-frame field. *ACM Transactions on Graphics 30*, 6 (2011).

[65] JAMESON, A. Aerodynamic design via control theory. *Journal of Scientific Computing 3*, 3 (September 1988), 233–260.

[66] JAMESON, A. Optimum aerodynamic design using control theory. *Computational Fluid Dynamics Review 3* (1995), 495–528.

[67] KOWALSKI, N., LEDOUX, F., AND FREY, P. A PDE Based Approach to Multidomain Partitioning and Quadrilateral Meshing. In *Proceedings of the 21st International Meshing Roundtable*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 137–154.

[68] LAUG, P., AND BOROUCHAKI, H. Bl2d-v2 : mailleur bidimensionnel adaptatif.

[69] LAWSON, C. Software for C1 Surface Interpolation. In *Mathematical Software*. Elsevier, 1977, pp. 161–194.

[70] LEE, C., AND LO, S. A new scheme for the generation of a graded quadrilateral mesh. *Computers & Structures 52*, 5 (September 1994), 847–857.

[71] LEE, D. T., AND SCHACHTER, B. J. Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer & Information Sciences 9*, 3 (June 1980), 219–242.

[72] LEICHT, T., AND HARTMANN, R. Error estimation and anisotropic mesh refinement for 3d laminar aerodynamic flow simulations. *Journal of Computational Physics 229*, 19 (September 2010), 7344–7360.

[73] LESAINT, P., AND RAVIART, P. On a Finite Element Method for Solving the Neutron Transport Equation. In *Mathematical Aspects of Finite Elements in Partial Differential Equations*. 1974, pp. 89–123.

[74] LÉVY, B., AND LIU, Y. Lp Centroidal Voronoi Tessellation and its Applications. *ACM Trans. Graph 29* (2010).

[75] LIONS, J. L. *Optimal control of systems governed by partial differential equations*. Springer-Verlag, Berlin,New York, 1971.

[76] LIU, Y., WANG, W., LÉVY, B., SUN, F., YAN, D.-M., LU, L., AND YANG, C. On Centroidal Voronoi Tessellation—energy smoothness and fast computation. *ACM Transactions on Graphics 28*, 4 (August 2009), 1–17.

[77] LLOYD, S. P. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory 28*, 2 (1982), 129–137.

[78] LOSEILLE, A. Metric-orthogonal Anisotropic Mesh Generation. *Procedia Engineering 82* (January 2014), 403–415.

[79] LOSEILLE, A., AND ALAUZET, F. Continuous Mesh Framework Part I: Well-Posed Continuous Interpolation Error. *SIAM Journal on Numerical Analysis 49*, 1 (January 2011), 38–60.

[80] LOSEILLE, A., AND ALAUZET, F. Continuous Mesh Framework Part II: Validations and Applications. *SIAM Journal on Numerical Analysis 49*, 1 (January 2011), 61–86.

[81] LOSEILLE, A., ALAUZET, F., AND MENIER, V. Unique cavity-based operator and hierarchical domain partitioning for fast parallel generation of anisotropic meshes. *Computer-Aided Design 85* (April 2017), 53–67.

[82] LOSEILLE, A., DERVIEUX, A., AND ALAUZET, F. Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations. *Journal of Computational Physics 229*, 8 (2010), 2866–2897.

[83] LOSEILLE, A., DERVIEUX, A., AND ALAUZET, F. Anisotropic Norm-Oriented Mesh Adaptation for Compressible Flows. *53rd AIAA Aerospace Sciences Meeting* (January 2015).

[84] LOSEILLE, A., DERVIEUX, A., FREY, P. J., AND ALAUZET, F. Achievement of Global Second Order Mesh Convergence for Discontinuous Flows with Adapted Unstructured Meshes. Tech. rep., 2007.

[85] LOSEILLE, A., AND LOHNER, R. Anisotropic Adaptive Simulations in Aerodynamics. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition* (Reston, Virigina, January 2010), American Institute of Aeronautics and Astronautics.

[86] LOSEILLE, A., AND LÖHNER, R. Cavity-based operators for mesh adaptation. *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013* (January 2013), 1–8.

[87] LOSEILLE, A., AND MENIER, V. Serial and Parallel Mesh Modification Through a Unique Cavity-Based Primitive. In *Proceedings of the 22nd International Meshing Roundtable*, J. Sarrate and M. Staten, Eds., no. 1. Springer International Publishing, Cham, 2014, pp. 541–558.

[88] LYON, M., CAMPEN, M., BOMMES, D., AND KOBBELT, L. Parametrization quantization with free boundaries for trimmed quad meshing. *ACM Transactions on Graphics 38*, 4 (July 2019), 1–14.

[89] MAUR, P. Delaunay triangulation in 3D. Tech. rep., 2002.

[90] MAVRIPLIS, D. J. Adaptive mesh generation for viscous flows using triangulation. *Journal of Computational Physics 90*, 2 (October 1990), 271–291.

[91] MCLAIN, D. H. Two Dimensional Interpolation from Random Data. *The Computer Journal 19*, 2 (May 1976), 178–181.

[92] MÜLLER, J.-D., ROE, P. L., AND DECONINCK, H. A frontal approach for internal node generation in Delaunay triangulations. *International Journal for Numerical Methods in Fluids 17*, 3 (August 1993), 241–255.

[93] NGUYEN, H., BURKARDT, J., GUNZBURGER, M., JU, L., AND SAKA, Y. Constrained CVT meshes and a comparison of triangular mesh generators. *Computational Geometry 42*, 1 (January 2009), 1–19.

[94] NOCEDAL, J. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation 35*, 151 (July 1980), 773.

[95] NOCEDAL, J., AND WRIGHT, S. *Numerical optimization*. Springer Science & Business Media, 2006.

[96] OPENMP ARCHITECTURE REVIEW BOARD. OpenMP application program interface version 3.0, May 2008.

[97] OWEN, S. J., STATEN, M. L., CANANN, S. A., AND SAIGAL, S. Q-Morph: an indirect approach to advancing front quad meshing. *International Journal for Numerical Methods in Engineering 44*, 9 (March 1999), 1317–1340.

[98] PALACIOS, J., AND ZHANG, E. Rotational symmetry field design on surfaces. *ACM Trans. Graph. 26*, 3 (July 2007), 55–64.

[99] PANOZZO, D., PUPPO, E., TARINI, M., AND SORKINE-HORNUNG, O. Frame fields: anisotropic and non-orthogonal cross fields. *ACM Transactions on Graphics 33*, 4 (July 2014), 1–11.

[100] PARK, C.-H., AND YANG, D.-Y. Adaptive refinement of all-hexahedral elements for three-dimensional metal forming analysis. *Finite Elements in Analysis and Design 43*, 1 (November 2006), 22–35.

[101] PARK, M. A. Thesis: Anisotropic Output-Based Adaptation with Tetrahedral Cut Cells for Compressible Flows. 164.

[102] PENNEC, X., FILLARD, P., AND AYACHE, N. A Riemannian Framework for Tensor Computing. *International Journal of Computer Vision 66*, 1 (January 2006), 41–66.

[103] PIERCE, N. A., AND GILES, M. B. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review 42*, 2 (2000), 247–264.

[104] RANGARAJAN, A., CHAKRABORTHY, A., AND MAY, G. A goal oriented hp-optimization technique for tetrahedral grids using a continuous-mesh model. In *AIAA Scitech 2019 Forum* (Reston, Virginia, January 2019), American Institute of Aeronautics and Astronautics.

[105] RANGARAJAN, A. M., CHAKRABORTY, A., MAY, G., AND DOLEJSI, V. A continuous-mesh optimization technique for piecewise polynomial approximation on tetrahedral grids. In *2018 Fluid Dynamics Conference* (Reston, Virginia, June 2018), American Institute of Aeronautics and Astronautics.

[106] RAY, N., AND BRUNO, L. Practical 3D Frame Field Generation. 1–9.

[107] REBAY, S. Efficient Unstructured Mesh Generation by Means of Delaunay Triangulation and Bowyer-Watson Algorithm. *Journal of Computational Physics 106*, 1 (May 1993), 125–138.

[108] REED, W. H., AND HILL, T. R. Triangular mesh methods for the neutron transport equation.

[109] REMACLE, J.-F., HENROTTE, F., CARRIER-BAUDOUIN, T., BÉCHET, E., MARCHANDISE, E., GEUZAINE, C., AND MOUTON, T. A frontal Delaunay quad mesh generator using the L∞ norm. *International Journal for Numerical Methods in Engineering 94*, 5 (May 2013), 494–512.

[110] REMACLE, J.-F., LAMBRECHTS, J., SENY, B., MARCHANDISE, E., JOHNEN, A., AND GEUZAINET, C. Blossom-quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm. *International Journal for Numerical Methods in Engineering 89*, 9 (2012), 1102–1119.

[111] RICHTER, G. R. An optimal-order error estimate for the discontinuous Galerkin method. *Mathematics of Computation 50*, 181 (1988), 75–75.

[112] RINGUE, N. *Thesis: An Optimization-based Approach to Mesh-Polynomial Adaptation of High-Order Discretizations*. PhD thesis, McGill University, 2019.

[113] RINGUE, N., AND NADARAJAH, S. K. An optimization-based framework for anisotropic *hp*-adaptation of high-order discretizations. *Journal of Computational Physics 375* (2018), 589–618.

[114] SCHNEIDERS, R. Refining quadrilateral and hexahedral element meshes. In *5th International Conference on Grid Generation in Computational Field Simulations* (1996), CRC Press, pp. 679–688.

[115] SHAMOS, M. I., AND HOEY, D. Closest-point problems. In *16th Annual Symposium on Foundations of Computer Science (sfcs 1975)* (October 1975), IEEE, pp. 151–162.

[116] SLOTNICK, J., KHODADOUST, A., ALONSO, J., GROPP, W., AND MAVRIPLIS, D. CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. Tech. rep., 2014.

[117] STERN, F., WANG, Z., YANG, J., SADAT-HOSSEINI, H., MOUSAVIRAAD, M., BHUSHAN, S., DIEZ, M., SUNG-HWAN, Y., WU, P.-C., YEON, S. M., DOGAN, T., KIM, D.-H., VOLPI, S., CONGER, M., MICHAEL, T., XING, T., THODAL, R. S., AND GRENESTEDT, J. L. Recent progress in CFD for naval architecture and ocean engineering. *Journal of Hydrodynamics 27*, 1 (February 2015), 1–23.

[118] SUTHERLAND, I. E., AND HODGMAN, G. W. Reentrant polygon clipping. *Communications of the ACM 17*, 1 (January 1974), 32–42.

[119] THE CGAL PROJECT. *CGAL User and Reference Manual*, 5.1.1 ed. CGAL Editorial Board, 2020.

[120] TOTH, C. D., O'ROURKE, J., AND GOODMAN, J. E. *Handbook of discrete and computational geometry*. CRC press, 2017.

[121] TRILINOS PROJECT TEAM, T. *The Trilinos Project Website*, 2020.

[122] VALLET, M.-G. Generation de maillages anisotropes adaptes, application a la capture de couches limites.

[123] VAXMAN, A., CAMPEN, M., DIAMANTI, O., PANOZZO, D., BOMMES, D., HILDEBRANDT, K., AND BEN-CHEN, M. Directional field synthesis, design, and processing. *Computer Graphics Forum 35*, 2 (2016), 545–572.

[124] VENDITTI, D. A., AND DARMOFAL, D. L. Adjoint Error Estimation and Grid Adaptation for Functional Outputs: Application to Quasi-One-Dimensional Flow. *Journal of Computational Physics 164*, 1 (2000), 204–227.

[125] VENDITTI, D. A., AND DARMOFAL, D. L. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal of Computational Physics 176*, 1 (2001), 40–69.

[126] VENDITTI, D. A., AND DARMOFAL, D. L. Anisotropic grid adaptation for functional outputs: Application to two-dimensional viscous flows. *Journal of Computational Physics 187*, 1 (2003), 22–46.

[127] VIERTEL, R., AND OSTING, B. An approach to quad meshing based on harmonic cross-valued maps and the Ginzburg-Landau theory. *SIAM Journal on Scientific Computing 41*, 1 (2019), A452–A479.

[128] VISWANATH, N., SHIMADA, K., AND ITOH, T. Quadrilateral Meshing with Anisotropy and Directionality Control via Close Packing of Rectangular Cells.pdf. *Imr9*, 412 (2000).

[129] WANG, Z., FIDKOWSKI, K., ABGRALL, R., BASSI, F., CARAENI, D., CARY, A., DECONINCK, H., HARTMANN, R., HILLEWAERT, K., HUYNH, H., KROLL, N., MAY,

G., PERSSON, P.-O., VAN LEER, B., AND VISBAL, M. High-order CFD methods: current status and perspective. *International Journal for Numerical Methods in Fluids 72*, 8 (July 2013), 811–845.

[130] WANG, Z. J. High-order methods for the Euler and Navier-Stokes equations on unstructured grids. *Progress in Aerospace Sciences 43*, 1-3 (2007), 1–41.

[131] WATSON, D. F. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *The Computer Journal 24*, 2 (February 1981), 167–172.

[132] YAN, D. M., WANG, W., LÉVY, B., AND LIU, Y. Efficient computation of clipped Voronoi diagram for mesh generation. *CAD Computer Aided Design 45*, 4 (2013), 843–852.

[133] YANO, M., AND DARMOFAL, D. L. An optimization-based framework for anisotropic simplex mesh adaptation. *Journal of Computational Physics 231*, 22 (September 2012), 7626–7649.

[134] ZHANG, L., CUI, T., AND LIU, H. A Set of Symmetric Quadrature Rules on Triangles and Tetrahedra. *Journal of Computational Mathematics 27*, 1 (2009), 89–96.

[135] ZHOU, T., AND SHIMADA, K. An angle-based approach to two-dimensional mesh smoothing. In *In Proceedings, 9th International Meshing Roundtable* (2000), pp. 373–384.