Application of Moving Mesh Methods for the Solution of Partial Differential Equations

Appella, Simone

Award date:
2023

Awarding institution:
University of Bath

Application of Moving Mesh Methods for the Solution of Partial Differential Equations

submitted by

Simone Appella

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Mathematical Sciences

September 2022

COPYRIGHT NOTICE

DECLARATION OF ANY PREVIOUS SUBMISSION OF THE WORK

The material presented here for examination for the award of a higher degree by research has not been incorporated into a submission for another degree.

Signature of Author . . . . . 

Simone Appella

DECLARATION OF AUTHORSHIP

I am the author of this thesis, and the work described therein was carried out by myself personally in collaboration with my supervisors.

Chapter 3 is reproduced from a submitted and accepted manuscript.

Chapter 4 and 5 are reproduced from a draft manuscript.

Signature of Author . . . . . 

Simone Appella

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisors, Prof. Chris Budd and Dr. Tristan Pryer, for their dedicated support and invaluable advice during my PhD study. Their vast knowledge and experience have continuously provided guidance for my academic research.

I joined alongside my colleagues and friends Tina Yang Zhou, Tosin Babasola and Gianluca Audone many study groups and group meetings organised by Prof. Budd throughout three years. As a result, I have developed a genuine interest in applied mathematical problems and more self-confidence by talking to general audience. I am grateful to them for all those experiences.

This thesis would not have been possible without the inclusive and welcoming environment within the Department of Mathematical Sciences. I would like to thank all my friends and colleagues for the cherished time spent together in the office and in social events. I am also thankful to the SAMBa coordinators, who have offered invaluable moral support and assistance for any concern.

During the COVID-19 pandemic I had to spend most of my time at home. The friendly relationship with my flatmates Marco Murtinu, Saul Gonzalez Resines, Carlo Scali and Umberto De Ambroggio helped me overcome that difficult period and I will never forget our mutual support.

Finally, a special thank to my family and to my friends in Italy. Their belief in me has kept spirit and motivation high during my research journey.

Abstract

This thesis concerns the application of moving mesh methods for the numerical solution of partial differential equations (PDEs) on two-dimensional domains. Recent developments in this field have shown wide applicability for resolving fine scale features such as shocks and singularities in nonlinear PDEs.

As first contribution, we couple a moving mesh PDE and a finite element (FE) method for the solution of the linear advection equation. This scheme also includes a mass-conservative projection operator that avoids any interpolation procedure as used in the standard iterative (rezoning) method. This property is essential for many simulations in the field of computational fluid dynamics (CFD), and prevents the approximate solution from being polluted by the addition of numerical diffusion.

The second contribution of this thesis is related to elliptic PDEs in non-convex domains, discretised with the symmetric interior penalty discontinuous Galerkin (SIP-dG) method. The analytical solution exhibits a singular behaviour at the re-entrant corners, and standard numerical schemes on regular grids are not able to accurately resolve the corner singularities. We will show that both an h-adaptive and r-adaptive approach can be used to increase considerably the accuracy of the solution. For the first method, we derive a new a-posteriori error estimator using the dual formulation of the original problem. Different moving mesh methods are tested for r-adaptivity, and that one based on the Optimal Transport (OT) strategy yields the best accuracy, comparable to the h-refinement method but more computationally efficient. Furthermore, we prove that the local quality measure of the OT mesh is independent on its resolution and cell location. Finally, we provide numerical evidence of the link between the two adaptive methods by showing that the a-posteriori estimator used for h-adaptivity is equidistributed on the OT mesh near the re-entrant corner.

The third contribution proposes an alternative way to solve a model elliptic PDE via deep learning (DL). Under this framework the problem is formulated as the minimisation of a loss function, which encodes desirable properties of the PDE. We show that a neural network with loss defined as the energy functional of the equation yields accurate solution, provided that the training points are chosen according to the equidistribution condition. We link to OT meshes obtained earlier in the thesis. The main result of this work suggests that numerical methods (e.g. used to derive the OT mesh) must be endowed with certain desirable features to ensure compatibility with DL procedures and improve the training process of the neural network.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation for the thesis

Mesh adaptivity is a critical component for the accurate resolution of multi-scale, multi-physics problems modelled by partial differential equations (PDEs). By increasing the mesh resolution or the polynomial degree of the numerical solution, the PDE can be approximated more accurately without significant impact on the computational costs as would be in the case with uniform resolution. There has been a great deal of research in the development and implementation of adaptive mesh methods for resolving solution features, which broadly fit into three categories: *h-adaptive*, *p-adaptive*, and *r-adaptive* [PLW05].

The most widely developed adaptive method is *h-refinement*, which locally coarsens or refines the mesh by removing or adding mesh points, and hence changes the mesh topology and number of points. For this method, the elements to refine/coarsen are usually chosen based on a local error estimate of the solution [Ver94]. This method requires coarsening and refinement for hyperbolic and parabolic PDEs at many iterations. This process can be computationally expensive in high-dimensions and, for finite difference and finite volume methods, must track hanging nodes resulting from the addition or subtraction of mesh points [Gar77]. Nevertheless, *h*-refinement coupled with finite element (FE) methods is a mature technology and offers a rich literature of mathematical results which is lacking in other adaptive strategies [Bab71; Ver94; Bak97; Ape13].

The *p-adaptive* method does not modify the mesh topology and the number of points, but locally increases or decreases the order of the spatial discretisation. For FE method, this involves increasing the order of the basis function, whereas in the Finite Difference method it corresponds to using a higher order differencing formula. There exists a wide literature of research that combines *h*- and *p*- adaptivity to balance the addition of mesh points with the increment of approximation order [AS98; CGH14; RPD06].

In contrast to the previous methods, the *r-adaptive* framework, also known as *moving mesh method*, seeks a map from a computational (uniform) domain to a physical (adapted) domain, so that the desired mesh in the physical domain is the image of a (fixed) mesh in the computational domain. In the discretised form, the adapted mesh has fixed number of points that move according to an *equidistribution* and *alignment* condition, such that the density of mesh points is increased in regions of the domain where the solution exhibits small-scale features. This method requires fewer degrees of freedom than h-refinement and does not increase the order of basis function/differencing formulas, as demanded by p-adaptivity. This method has proved successful in a variety of applications, ranging from Burgers equation [Coo+19], semi-linear blow up equations [BWG04], and numerical weather prediction [MCB18]. The most common scheme used to couple the solution of hyperbolic/parabolic PDEs and the moving mesh method is the *rezoning method*, which alternates the steps of solving the PDE and updating the mesh [HR11]. This method is computationally cheap but does require the interpolation of the approximate solution from the current adapted mesh to the new adapted one at each iteration of the numerical scheme. This introduces artificial diffusion, which ultimately affects the accuracy of the numerical solution. In Chapter 3 we bridge this gap by implementing a conservative data transfer operator that is based on the concept of *supermesh* [FM10].

The theory of Optimal Transport (OT) plays an important role in r-adaptivity. It has been shown in [BHR09] that coupling the equidistribution condition with an OT strategy results in the formulation of a nonlinear Monge-Ampère equation, whose solution leads to a mesh with many desirable properties. Although the numerical solution of the Monge-Ampère equation is time expensive in general, we propose in Chapter 4 an alternative cheaper approach that exploits the topology of the domain and the radial symmetry of the solution of a Poisson problem in non-convex domains. In particular, the OT map that leads to the equidistributed mesh can be rewritten in terms of a non-algebraic equation for a L-shaped and crack-domain. We propose a novel a-posteriori error to drive the h-adaptation and analyse the property of the resulting OT mesh. Finally, we compare the two methods in terms of accuracy and quality of the mesh.

Since *deep learning* is becoming an increasingly popular framework for the solution of PDEs [RPK19; KZK21; SS18], we adopt some of the proposed methods. In Chapter 5 we train a *neural network* for the solution of Poisson's equation on training points, which are located according to an equidistribution condition. The rich literature on this topic has provided important theoretical results, mostly related to the asymptotic behaviour of network parameters, to define the function approximation power of networks with specific configurations [MMN18; SKM21; JHG18]. The training of a neural network consists in the minimisation of a *loss function* that embeds the physical properties of the PDE. This is usually performed using *stochastic gradient descent* method, which discretises a loss (objective) function over a set of training points randomly sampled. Despite the cited papers have proved successful in solving several PDEs, it is still not clear how to train the network to achieve the optimal accuracy and how the location of training points affects the solution of the PDE. Moreover, we will show that this framework is not as robust as the FE method because careful tuning of the network parameters is required to possibly yield a satisfactory convergence rate.

1.2 Achievements of the thesis

This thesis aims to improve the existing applications of moving mesh methods for multi-dimensional hyperbolic and elliptic PDE problems with challenging solution structures. In Chapter 3 the linear advection problem will be solved using a rezoning approach, alternating the mesh adaptation and the approximate solution of the PDE using a FE method. In Chapter 4 Poisson's equation will be solved using a FE method with the help of h- and OT based r-adaptation. The OT mesh will be employed in Chapter 5 to obtain an accurate approximate solution of the same equation by training a neural network. We will compare the accuracy of the network output with the FE solution obtained in Chapter 4.

Here, we list the main achievements for each Chapter of the thesis:

- Chapter 3: we apply r-adaptivity to the linear advection equation using the rezoning approach, which couples the adaptive strategy with the upwinding discontinuous Galerkin (dG) discretisation. This scheme requires an interpolation step and introduces various challenges as it is a common source of diffusion. The novelty of this Chapter is a modified rezoning method that ensures the mass conservation property. We achieve this through a conservative local Galerkin projection, removing one of the inherent disadvantages of interpolation based methods [FM10]. We conduct numerical experiments to assess the robustness and approximability of the proposed scheme. We observe mass conservation and an increase of the solution accuracy due to the adapted mesh.

- Chapter 4: we solve Poisson's equation in non-convex two-dimensional domains using the symmetric interior penalty discontinuous Galerkin (SIP-dG) method. The solution exhibits a singularity at the re-entrant corner and lies in a weighted Sobolev space. Under this framework, we derive a novel L^2 a-posteriori error estimate. We discretise the domain on meshes using either an h-adaptive or an OT based moving mesh strategy. The h-refinement strategy is guided by the proposed L^2 and a L^∞ a-posteriori estimate, and ensures each numerically optimal convergence rate for the SIP-dG method. The OT mesh is derived semi-analytically, by considering the local behaviour of the solution near the re-entrant corner. This second procedure exhibits the same accuracy as the h-refinement strategy but is computationally less expensive. Moreover, we show that the quality measure of the OT mesh elements does not depend on its location and the number of total mesh nodes. Finally, we link the two adaptive strategies by showing that the OT mesh equidistributes the a-posteriori estimate in L^2 and a L^∞ norm.

- Chapter 5: we propose an alternative method to solve Poisson's equation by using the neural network framework. The deep Galerkin (DGM) and the deep Ritz method (DRM) is used to minimise a loss (objective) function related to the problem. The loss functions of these two formulations are related to their FE counterparts. We show that the choice of the quadrature points is crucial for the accuracy of the solution. More precisely, if we use the vertices of the OT mesh derived in the Chapter 4 as training points, the training process is more stable and the accuracy much higher compared to uniformly random sampled points. Moreover, the DRM proves to be

more accurate than the DGM. This may be related to the variational formulation of the DRM, that is supposed to be naturally adaptive during the training of the network.

1.3 Structure of the thesis

In Chapter 2 we propose an overview of the frameworks used in the next three Chapters. We begin with the core mathematical principle at the foundation of moving mesh methods in one and higher dimensions. We consider variational methods and an Optimal Transport based method, which consists of solving a nonlinear Monge-Ampère equation. Next, we introduce the fundamental concepts and theoretical results of the FE method. Finally, we illustrate the main components of the training procedure of a neural network and theoretical results characterising its approximation power.

Chapter 3 deals with the solution of a linear advection problem in two-dimensional space with a moving mesh method that assures mass conservation. The physical PDE is discretised using the lowest order discontinuous Galerkin method, while r-adaptivity is applied through Winslow's method. The projection of the approximate physical solution from the old to the new adapted mesh is achieved through a mass-conservative L^2 projection, which requires the definition of a *supermesh*. Numerical experiments confirm the expected conservation properties and illustrate error analysis. We conclude the Chapter by summarising the implications of the numerical results.

In Chapter 4 we conduct different numerical experiments related to h- and r-adaptivity applied to Poisson's equation in two-dimensional non-convex domains. Under the SIP-dG discretisation, we derive a novel L^2 a-posteriori error estimate and use it to drive h-adaptation. We also expose our method to construct an OT mesh semi-analytically for r-adaptivity. Numerical experiments compare the two methods in terms of accuracy and quality measure.

Chapter 5 concerns the application of neural network as an alternative approach to solve Poisson's equation defined in Chapter 4. We introduce two loss functions, based on the PDE residual and the definition of an energy functional. Both losses are discretised via mean squared error and the trapezoidal quadrature rule. We show that the choice of training points based on the OT strategy provides approximate solutions with much higher accuracy compared to the choice of randomly sampled collocation/quadrature points. Another numerical experiment on a squared domain corroborates the previous result. The Poisson's equation is solved more accurately by using the DRM with non-randomly sampled quadrature points.

We wrap up our results and suggest open areas for future research in Chapter 6.

Chapter 2

Background material

Abstract

This Chapter contains a self-comprehensive review of the theoretical frameworks and the numerical methods used in the next Chapters of the thesis. Firstly, we introduce the concept of mesh adaptivity, which is the main theme all the works are linked to. In particular, we discuss different implementations of one- and multi-dimensional moving mesh methods for the solution of partial differential equations (PDEs). We provide then an overview of Finite Element (FE) methods, with the most important theoretical results. Particular attention is paid on the discretisation of non-convex domains and on mesh quality measures, as these topics will be dealt with to a large extent for the remainder of the thesis. Finally, we illustrate the main components of deep learning (DL) applied for the solution of PDEs. We describe the training procedure of neural networks and illustrate the main theoretical Theorems that express their function approximation power.

Mesh adaptivity has been proved successful in improving the accuracy and efficiency of numerical solutions to PDEs [PLW05; HR11]. There exist different ways the adaptation can be implemented. We will focus mostly on r-adaptive (*moving mesh*) methods, which relocate a fixed number of mesh nodes and must satisfy an *equidistribution* and *alignment* condition. The connectivity of the mesh is unaltered and the accuracy that is obtained by these methods is much higher compared to a uniform mesh with the same number of points. Moving mesh methods have received a huge interest in the past decades in both theoretical research and practical applications for the solutions to PDEs in different fields, ranging from computational fluid dynamics (CFD) [Tan05; Zha+93; QS07] to pattern formation [Gav+11; GL13]. The general idea behind the application of r-adaptivity in general dimensions is related to the concept of *equidistribution* and *alignment* condition. There exist several formulations of moving

mesh strategies to accomplish those two conditions, and they are generally independent on the specific type of the underlying physical PDE that must be solved. The implementation of r-adaptivity may be computationally expensive, as involves the discretisation of both the physical PDE and of a moving mesh PDE to drive the mesh adaptation. However, this problem can be mitigated for specific choices of FE spaces by parallelising the code. In the following section, we will examine mesh equidistribution in one spatial dimension and detail different methods to derive numerically the equidistributed mesh. We will then discuss mesh adaptivity in the multidimensional context, with a particular focus on the Optimal Transport (OT) strategy.

The h- and p- versions of the Finite Element method (FEM) are different ways of adding degrees of freedom (*dofs*) to the model. The h-adaptive method improves the accuracy of the result by refining the mesh in areas containing a high error. A criterion must be specified to select the elements that are going to be refined. The chosen elements decrease their length and are split into two or more elements of the same type. The p-adaptive strategy improves the accuracy of the numerical solution by using the same mesh, with the same number of elements, but increasing the *dofs* in some elements based on some criteria. The main difference between the h-adaptive and p-adaptive method lies in how these elements are treated. The h-method uses many simple elements, whereas the p-method uses few complex elements. In general, we see different convergence behaviors for h- and p- refinements. While h-refinement leads to algebraic convergence error in the energy norm, p-refinement leads to an exponential convergence under specific global regularity of the solution [BG92; Bab71]. If singularities are present in the model, both p- and h-refinement lead to algebraic convergence while the slope of curve for p-refinement is twice that of the h-refinement [FDF11]. In this case, the higher costs of p-elements and the more dense system matrices might be a problem. Therefore, an hp-refinement can be used to recover the exponential rate of convergence and decrease the influence of the singularities. We will show that h-refinement is enough to ensure exponential convergence rate using a prescribed a-posteriori estimator to address the solution of Poisson's equation in non-convex domains [BBO99; HSS02].

2.1 Moving meshes in 1D

We start our discussion on mesh adaptivity by looking at the problem of constructing an r-adapted mesh in 1D. We aim to find the optimal location of a fixed number of points to best approximate a known function $u(x)$ by means of piecewise constant/linear interpolation. The equidistribution condition is the key to find a solution to this problem. Given an integer $N > 1$ and a continuous integrable monitor function $\rho(x) > 0$, the equidistribution condition involves the derivation of a mesh $\mathcal{T} = \{x_i\}_{i=1}^{N}$ so that $x_1 = a < x_2 < \cdots < x_N = b$, which evenly distributes ρ over all the sub-intervals:

$$\int_{x_1}^{x_2} \rho(x)\,\mathrm{d}x = \cdots = \int_{x_{N-1}}^{x_N} \rho(x)\,\mathrm{d}x. \tag{2.1.1}$$

This entails that the area under ρ is the same for each interval. The function ρ is defined as the *monitor*

function and encodes geometric information about u, such as slope or curvature. To express these quantities accurately, we first define the space of continuously differentiable functions of order k by:

Definition 2.1.1 (Spaces of continuously differentiable functions) *Let k be a nonnegative integer, $\Omega \subset \mathbb{R}^d$ an open bounded domain, and α a multi-index defined as $\alpha = (\alpha_1, \ldots, \alpha_d)$ and with $|\alpha| = \alpha_1 + \cdots + \alpha_d$. Then*

$$C^k(\overline{\Omega}) = \{u(x) : D^\alpha u(x) \in C(\overline{\Omega}),\ |\alpha| \leq k\}$$

is a Banach space equipped with the norm $\|u\|_{C^k(\overline{\Omega})} = \sum_{|\alpha|\leq k} \sup |D^\alpha u(\boldsymbol{x})|$. Here $D^\alpha u := \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}}$.

In 1D, there exists a unique equidistributed mesh of N points satisfying (2.1.1) for any strictly positive monitor function.

For practical purposes, it is useful to rewrite (2.1.1) as

$$\int_a^{x_j} \rho(x)\,\mathrm{d}x = \frac{(j-1)}{(N-1)}\sigma, \quad j = 1, \ldots, N, \tag{2.1.2}$$

where $\sigma = \int_a^b \rho(x)dx$ and $\xi_j = \frac{(j-1)}{(N-1)},\ j = 1, \ldots, N$ compose a uniform mesh on $[0, 1]$.

For a better mathematical understanding of the equidistribution process, we can think of $\mathcal{T}$ as being generated by the differentiable coordinate transformation $x = x(\xi) : [0, 1] \to [a, b]$ such that $x_j = x(\xi_j), \quad j = 1, \ldots, N$. The continuous version of (2.1.2) can be then rewritten as

$$\int_a^{x(\xi)} \rho(x)\,\mathrm{d}x = \sigma\xi \quad \forall \xi \in (0, 1). \tag{2.1.3}$$

Differentiating (2.1.3) with respect to ξ, the output of the map $x(\xi)$ satisfies

$$\rho(x)\frac{\mathrm{d}x}{\mathrm{d}\xi} = \sigma. \tag{2.1.4}$$

This formula says that if the density function $\rho(x)$ takes on high values, the derivative $\mathrm{d}x/\,\mathrm{d}\xi$, and hence the distance between two consecutive mesh points, has to be small. If $u \in C^2\big([a, b]\big)$ and we consider the question of approximating it with a piecewise linear interpolating function, then the density ρ must be a normalised scaling of its curvature in order to minimise the interpolation error, such that

$$\rho(x) = \sqrt{1 + \beta u_{xx}^2(x)}, \tag{2.1.5}$$

where the constant β is tuned to control the amount of mesh points that cluster in proximity of highest curvature regions. Another popular choice is the arc-length monitor function $\rho(x) = \sqrt{1+\beta u_x(x)^2}$, which does require lower regularity for u and is used if we are trying to interpolate it using a piecewise constant function [HR11].

2.1.1 De Boor's algorithm

The de Boor's algorithm is a practical method of finding an equidistributed mesh. For $\Omega = [a, b]$, the algorithm defines initially a uniform mesh $\mathscr{T}^0 = \{x_i^0\}_{i=1}^N$ [Boo73]. We then approximate ρ as a piecewise constant function and define $P(x) = \int_a^x \rho(x)\,\mathrm{d}x$. Then we discretise the integral as

$$P(x_j) = \sum_{i=2}^{j} (x_i - x_{i-1}) \frac{\rho(x_i) + \rho(x_{i-1})}{2}, \quad j = 2, \dots, N. \tag{2.1.6}$$

The equidistribution condition (2.1.2) reads as: $P(x_j) = \xi_j P(b), \quad j = 2, \dots, N-1.$

To find the mesh $\mathscr{T}^1 = \{x_j^1\}_{j=1}^N$ at the next iteration, one first determines the index k such that

$$P(x_{k-1}^0) < \xi_j P(b) < P(x_k^0). \tag{2.1.7}$$

We then obtain from the linear approximation of $P(x)$ with respect to the mesh points

$$\xi_j P(b) - P(x_{k-1}^0) = (x_j^1 - x_{k-1}^0) \frac{\rho(x_{k-1}) + \rho(x_k)}{2}, \tag{2.1.8}$$

from which we can derive the node position at the next iteration

$$x_j^{n+1} = x_{k-1}^n + \frac{2\Big(\xi_j P(b) - P(x_{k-1}^n)\Big)}{\rho(x_{k-1}^n) + \rho(x_k^n)}.$$

Even though de Boor's algorithm equidistributes meshes for a piecewise constant monitor function ρ, the sequence of meshes $\{\mathscr{T}^n\}$ converges to $\hat{\mathscr{T}} = \{\hat{x}\}_{i=1}^N$, satisfying

$$(\hat{x}_j - \hat{x}_{j-1}) \frac{\rho(\hat{x}_{j-1}) + \rho(\hat{x}_j)}{2} = (\xi_j - \xi_{j-1}) \hat{\sigma}_h, \quad j = 2, \dots, N, \tag{2.1.9}$$

where $\hat{\sigma}_h = \sum_{j=2}^N (\hat{x}_j - \hat{x}_{j-1}) \frac{\rho(\hat{x}_{j-1}) + \rho(\hat{x}_j)}{2}$.

The mesh sequence produced by the corresponding iterative method with a piecewise *linear* approximation to the mesh density function converges to a limit mesh satisfying (2.1.9) with $\max_i\{x_i^{n+1}-x_i^n\} \xrightarrow{n} 0$. The mesh difference decreases at roughly the rate $O(\frac{1}{N^2})$, provided that the mesh density function is sufficiently smooth and N sufficiently large [Pry89]. Subsequent theoretical results in [Xu+10] show convergence for the above case of piecewise constant interpolation.

2.1.2 The Boundary Value Problem method

We illustrate another practical method to compute the equidistributing mesh $\mathscr{T}$ based on a boundary value problem (BVP) formulation of the equidistribution principle.

Differentiating equation (2.1.4) with respect to ξ we obtain the second order differential equation

$$\frac{\mathrm{d}}{\mathrm{d}\xi}\left(\rho(x)\frac{\mathrm{d}x}{\mathrm{d}\xi}\right) = 0, \tag{2.1.10}$$

subject to the boundary conditions $x(0) = a$ and $x(1) = b$.

Suppose that an approximation $(\mathscr{T},\rho)^n$ to $(\mathscr{T},\rho)$ and a mesh density function ρ^n are given at iteration n. Discretising the BVP (2.1.10) on a fixed computational mesh $\mathscr{T}_c : \{\xi_j\}_{j=1}^N$ and using central finite difference we obtain

$$\frac{2}{\xi_{j+1}-\xi_{j-1}}\left(\frac{\rho(x_{j+1}^n)+\rho(x_j^n)}{2}\cdot\frac{x_{j+1}^{n+1}-x_j^{n+1}}{\xi_{j+1}-\xi_j} - \frac{\rho(x_j^n)+\rho(x_{j-1}^n)}{2}\cdot\frac{x_j^{n+1}-x_{j-1}^{n+1}}{\xi_j-\xi_{j-1}}\right) = 0, \quad j = 2,\dots,N-1, \tag{2.1.11}$$

with $x_1 = a$ and $x_N = b$. Keeping ρ fixed on the current iteration, the system is linear for the next iteration and can be solved for the mesh $\mathscr{T}^{n+1}$ until reaching convergence to $\mathscr{T}$.

2.1.3 Moving Mesh PDEs

For the numerical solution of time-dependent problems, the monitor function ρ will depend on the time variable, too. The coordinate transformation is chosen such that $x = x(\xi,t)$ satisfies

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\rho(x)\frac{\mathrm{d}x}{\mathrm{d}\xi}\right) &= 0, \\ x(0,t) = a, \quad x(1,t) &= b. \end{aligned} \tag{2.1.12}$$

A semi-discretisation of eq.(2.1.12) using finite differences or finite elements gives a system of ODEs. Moreover, the introduction of the mesh speed into the equation provides a degree of temporal smoothing for mesh movement, which is necessary for an accurate solution of many PDEs.

A mesh equation involving mesh speed is referred to as *moving mesh PDE* (MMPDE). There exists numerous ways to define the MMPDEs [HRR94a; HRR94b]. The most popular ones derives from the gradient flow equation of an adaptation functional [HR98a; HR98b]. A popular choice of MMPDE, named MMPDE5, reads as

$$\frac{\partial x}{\partial t} = \frac{1}{\tau}\frac{\partial}{\partial \xi}\left(\rho \frac{\partial x}{\partial \xi}\right), \tag{2.1.13}$$

where the parameter τ controls the speed of the mesh points which move over time. We note that the right hand side contains the equidistribution relation (2.1.10). This term drives the mesh movement towards the equidistribution of the monitor function ρ. The term vanishes when the equidistribution is satisfied, giving no mesh movement. Moreover, if ρ is time-independent we obtain

$$x(\xi) = \lim_{t\to\infty} x(\xi, t) \;\Rightarrow\; \frac{1}{\tau}\frac{\partial}{\partial \xi}\left(\rho \frac{\partial x}{\partial \xi}\right) \to 0. \tag{2.1.14}$$

The MMPDE5 can be discretised in space using central finite differences or a finite element method in space and a forward/backward Euler method in time.

2.2 Higher dimensional Moving Mesh methods

In one dimension, the equidistribution condition with suitable boundary conditions is sufficient to determine uniquely the map $\xi \to x(\xi)$ from the computational domain Ω_c to the physical domain Ω. In higher dimensions, that condition only fixes the size, but not the orientation and the skewness of the mesh elements. Additional constraints are required to define a unique and well-posed mesh mapping. We will give a brief introduction to meshes constructed from variational principles (e.g. MMPDE) and optimally transported mesh (e..g. Monge-Ampère equation) that will be used in Chapter 4 and 5. Such a moving mesh PDE in higher dimensions can be found by minimising a functional, which characterises specific mesh properties, such as alignment or orthogonality. A matrix monitor function is also described to control all those properties.

The functional is formulated in terms of either the coordinate transformation $\boldsymbol{x} = \boldsymbol{F}(\boldsymbol{\xi}, t)$ or the inverse transformation $\boldsymbol{\xi} = \boldsymbol{F}^{-1}(\boldsymbol{x}, t)$. The latter is used more frequently as avoids mesh tangling [Dvi91]. Hence, the general idea is to first solve the MMPDE for $\boldsymbol{\xi}(\boldsymbol{x})$ and then determine the transformed coordinate $\boldsymbol{x}(\boldsymbol{\xi})$ by interpolation [Hag94].

The MMPDE for the inverse map is derived from the minimisation of the adapted functional

$$I[\boldsymbol{\xi}] = \int_{\Omega} G(\boldsymbol{M}, \boldsymbol{\xi}, \nabla_{\boldsymbol{x}}\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{x}, \tag{2.2.1}$$

where G is a continuous function and $\boldsymbol{M}$ is the matrix monitor function. Supposing that there exists a steady mesh generation strategy, which consists of minimising $I[\boldsymbol{\xi}]$ along the first-order functional derivative, the gradient flow equation is defined by

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = -\frac{\delta I}{\delta \boldsymbol{\xi}}. \tag{2.2.2}$$

In practice, it is more suitable to define different descent directions than the fastest one and to adjust the time scale of the mesh equation. Hence, equation 2.2.2 is redefined as

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = -\frac{P}{\tau}\frac{\delta I}{\delta \boldsymbol{\xi}}, \tag{2.2.3}$$

where P is a *positive-definite differential operator*, defined as a bounded symmetric operator such that the inner product $\langle P\boldsymbol{x}, \boldsymbol{x}\rangle > 0$ for all $\boldsymbol{x} \neq 0$. The positive constant ε controls the timescale of the mesh movement [HK15a; HR98b].

For sake of treatment, we consider the one-dimensional analogue of (2.2.3). In one dimension, De Boor's equidistribution principle is used to form the steady mesh generator. The idea behind this principle is to choose the coordinate transformation by equidistributing a monitor function $G(x) > 0$. The equidistribution equation can be obtained by minimising

$$I[\xi] = \frac{1}{2}\int_0^1 \frac{1}{G}\Big(\frac{\partial \xi}{\partial x}\Big)^2\,\mathrm{d}x, \tag{2.2.4}$$

where for convenience both the computational and physical domain are the unit interval. The Euler-Lagrange equation for (2.2.4) is

$$\frac{\delta I}{\delta \xi} = -\frac{\partial}{\partial x}\Big(\frac{1}{G}\frac{\partial \xi}{\partial x}\Big) = 0. \tag{2.2.5}$$

Taking $P = \big(\frac{G}{\xi_x}\big)^2 I$, where I is the identity matrix and $\xi_x = \frac{\partial \xi}{\partial x}$, we obtain the one-dimensional MMPDE

$$\frac{\partial \xi}{\partial t} = \frac{G^2}{\tau \xi_x^2}\frac{\partial}{\partial x}\Big(\frac{1}{G}\frac{\partial \xi}{\partial x}\Big). \tag{2.2.6}$$

After exchanging the roles of dependent and independent variables we obtain

$$\frac{\partial x}{\partial t} = \frac{1}{\tau}\frac{\partial}{\partial \xi}\Big(G\frac{\partial x}{\partial \xi}\Big), \tag{2.2.7}$$

which is exactly the MMPDE5 defined in equation (2.1.13). Obviously, the above general procedure works in any dimension, provided that a functional for steady mesh adaptation is given. However, the extension of equidistribution to two dimensions is neither straightforward nor unique. The derivation of the Euler-Lagrange equation for (2.2.1) is given in [HR11, Chapter 6].

Several moving mesh methods have been developed based on this method, depending on the choice of the adaptation functional. We refer to [BHR09; HR11] for the characterisation of different functionals and differential operators. We now describe Winslow's method, which will be used to solve a MMPDE coupled with a linear advection equation in Chapter 3 and a Poisson problem in Chapter 4.

2.2.1 Winslow's variational-based diffusion method

In this section, we review Winslow's adaptive diffusion method for generating iteratively an adapted mesh $\mathcal{T}$, tesselating an open bounded domain $\Omega \subset \mathbb{R}^d$, with respect to a given target function [Win66]. The computational mesh $\mathcal{T}_c$ remains fixed, whereas the physical mesh points of $\mathcal{T}_i$ move at each iteration. In a continuous setting, the problem can be formulated as seeking the image from a computational domain Ω_c, discretised by $\mathcal{T}_c$, to the physical domain Ω, tessellated by $\mathcal{T}_i$, $i \in \mathbb{N}$. With a variational method, this mapping is determined as the minimiser of an adaptation functional [Hua15; HS03]. The MMPDE associated with the minimiser will be solved by an Euler-Lagrange equation, that will be discretised in space and time in the following subsections. An example of adapted mesh with Winslow's method is provided in figure 2-1.

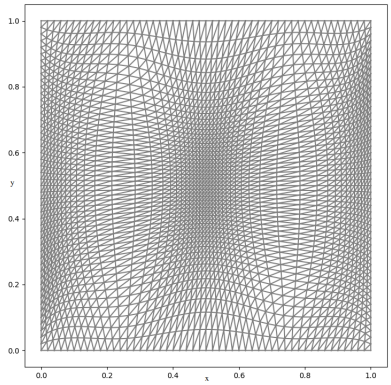


Figure 2-1: Adapted mesh to $u(x, y) = 100 + \sin(2\pi x)\sin(\pi y)$ using Winslow's diffusion method.

1 Derivation of the MMPDE As mentioned in the previous section, a common way to solve the MMPDE associated with $\boldsymbol{x}(\boldsymbol{\xi})$ is to solve first the MMPDE associated with the inverse coordinate transformation $\boldsymbol{\xi}(\boldsymbol{x})$ and then apply a change of variable.

The adaptation functional $I[\boldsymbol{\xi}]$ for the inverse map $\boldsymbol{\xi}(\boldsymbol{x})$ takes the form

$$I[\boldsymbol{\xi}] = \int_{\Omega} F(\nabla_{\boldsymbol{x}}\boldsymbol{\xi}, \boldsymbol{\xi}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \tag{2.2.8}$$

where the adaptation functional F accounts for the concept of *equidistribution* and *alignment*.

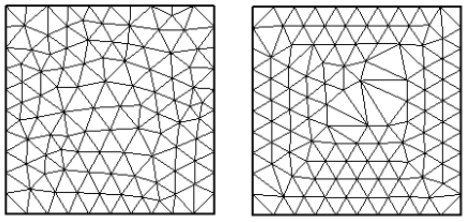
The former controls the size of each element of the mesh through a continuous integrable monitor function $\rho(\boldsymbol{x}) > 0$ such that

$$\int_{K} \rho(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \frac{\sigma}{N} \; \forall K \in \mathscr{T}, \tag{2.2.9}$$

where N is the number of elements of $\mathscr{T}$ and $\sigma = \int_{\Omega} \rho(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$. The equidistribution condition uniquely defines the transformation in one-dimension. A further *alignment* condition must be imposed for higher dimensions. This specifies a preferred direction for the edges of $\mathscr{T}$ to follow. We assume that the physical domain $\Omega \subset \mathbb{R}^d$ is polyhedral and that each reference element $K \in \mathscr{T}$ is an equilateral d-simplex with unit volume. Let $\gamma_1, \ldots, \gamma_{d(d+1)/2}$ be the edges of an element K, the condition requires that

$$|\gamma_1|_{\boldsymbol{M}} = \cdots = |\gamma_{d(d+1)/2}|_{\boldsymbol{M}}, \tag{2.2.10}$$

where $|\gamma_i|_{\boldsymbol{M}}$ is the length of γ_i in the metric $\boldsymbol{M}$. In the case of Winslow's adaptation method, the metric $\boldsymbol{M}$ does not enforce any preferred direction and can be written as $\boldsymbol{M} = \rho(\boldsymbol{x})\boldsymbol{I}_d$, where $\boldsymbol{I}_d$ is the identity matrix in $\mathbb{R}^{d\times d}$. Thus, the resulting adapted mesh is *isotropic*. Conversely, the mesh is said to be *anisotropic* when stretched elements are used, as visible in Figure 2-2. More precisely, the anisotropic mesh matrix monitor function defines the mapping from an ellipsoid into a unit sphere [Rem+00].



a Isotropic mesh. b Anisotropic mesh.

Figure 2-2: Visualisation of an isotropic and anisotropic mesh on a squared domain.

Incorporating equidistribution and alignment condition into an adaptation functional $I[\boldsymbol{\xi}]$ leads to a

MMPDE. In this thesis, we work with the simple and commonly used Winslow's adaptation functional [Win66]. The functional is given by

$$I_{win}[\boldsymbol{\xi}] = \frac{1}{2}\int_{\Omega}\frac{1}{\rho}\sum_{i}(\nabla\xi_i)^T(\nabla\xi_i)\,\mathrm{d}\boldsymbol{x}, \tag{2.2.11}$$

where ρ is a given monitor function. This function can be chosen to depend on the solution u of the physical PDE. A natural choice for $u \in C^1(\Omega)$ is given by the arc-length monitor function

$$\rho = \sqrt{1 + \frac{1}{\delta}|\nabla u|^2}, \tag{2.2.12}$$

where δ is a user-specified intensity parameter for $\mathscr{T}$ [HR11; BHR09], as this places more points in regions of high gradient. Different monitor functions are proposed §4.4.1 of Chapter 4.

Setting the first variation of (2.2.11) to zero leads to the following MMPDE [CRR15, equation 6]:

$$\frac{\partial\boldsymbol{x}}{\partial t} = \frac{1}{\tau b(\boldsymbol{x},t)}\left[\sum_{i,j}A_{i,j}\frac{\partial^2\boldsymbol{x}}{\partial\xi_i\partial\xi_j} + \sum_{i}B_i\frac{\partial\boldsymbol{x}}{\partial\xi_i}\right], \tag{2.2.13}$$

where $\boldsymbol{A}$ is a matrix and B is a vector involving the Jacobian matrix

$$\boldsymbol{J} = \frac{\partial\boldsymbol{x}}{\partial\boldsymbol{\xi}} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix},\ |J| = \det(\boldsymbol{J}). \tag{2.2.14}$$

The balancing function $b(\boldsymbol{x},t)$ should be chosen so that all mesh points move with a uniform time scale, while the user-controlled parameter $\tau > 0$ is chosen to adjust the time-scale of the mesh movement.

In the case $d = 2$, the computational domain is referenced by the coordinates $\boldsymbol{\xi} = (\xi,\eta)$, while the physical domain is expressed in terms of the continuous coordinates $\boldsymbol{x} = (x,y)$.

The matrix $\boldsymbol{A}$ and vector B are defined as

$$\begin{aligned}
A_{1,1} &= \frac{1}{|J|^2\rho}(x_\eta^2 + y_\eta^2),\ A_{1,2} = -\frac{1}{|J|^2\rho}(x_\xi x_\eta + y_\xi y_\eta) = A_{2,1},\ A_{2,2} = \frac{1}{|J|^2\rho}(x_\xi^2 + y_\xi^2),\\
B_1 &= \frac{1}{|J|^2\rho^2}\left[(x_\eta^2 + y_\eta^2)\rho_\xi - (x_\xi x_\eta + y_\xi y_\eta)\rho_\eta\right],\\
B_2 &= \frac{1}{|J|^2\rho^2}\left[-(x_\xi x_\eta + y_\xi y_\eta)\rho_\xi + (x_\xi^2 + y_\xi^2)\rho_\eta\right],
\end{aligned}$$

By substituting these coefficients into equation (2.2.13) and re-arranging some terms we obtain the parabolic MMPDE

$$\begin{aligned}\frac{\partial \boldsymbol{x}}{\partial t} = \frac{1}{|J|^2\rho^2}\Bigg[&(x_\eta^2 + y_\eta^2)\frac{\partial}{\partial \xi}\left(\rho\frac{\partial \boldsymbol{x}}{\partial \xi}\right) - (x_\xi x_\eta + y_\xi y_\eta)\frac{\partial}{\partial \eta}\left(\rho\frac{\partial \boldsymbol{x}}{\partial \xi}\right)\\ &- (x_\xi x_\eta + y_\xi y_\eta)\frac{\partial}{\partial \xi}\left(\rho\frac{\partial \boldsymbol{x}}{\partial \eta}\right) + (x_\xi^2 + y_\xi^2)\frac{\partial}{\partial \eta}\left(\rho\frac{\partial \boldsymbol{x}}{\partial \eta}\right)\Bigg].\end{aligned} \tag{2.2.15}$$

The discretisation in time of (2.2.15) allows us to compute $\boldsymbol{x}(\boldsymbol{\xi}, t_{i+1})$ and hence $\mathscr{T}_{i+1}$ given the knowledge of $\rho(\boldsymbol{x}(\boldsymbol{\xi}, t_i))$ and $\mathscr{T}_i$. The full spatio-temporal discretisation will be discussed in detail in Chapter 3 to drive the mesh adaptation for the linear advection equation. The same approach will be employed in Chapter 4 for Poisson's equation in L-shaped and crack domains.

2.2.2 Optimal Transport based mesh adaptation

A different way to find a unique map in higher dimensions is to impose an Optimal Transport (OT) constraint on the mesh. This method is a natural extension from the MMPDE methods in one dimension because it retains much of their simplicity, such as solving a scalar equation and having an automatic boundary mapping. It has been shown through several applications to be general enough to produce well resolved, high quality meshes [BW06; BW09; BCW13]. We first redefine the equidistribution principle in higher dimensions as

$$\boldsymbol{M}(\boldsymbol{x})|J(\boldsymbol{x})| = \theta := \frac{\int_\Omega M(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int_{\Omega_c}\mathrm{d}\boldsymbol{\xi}}, \tag{2.2.16}$$

where the Jacobian of the coordinate transformation $\boldsymbol{J}(\boldsymbol{x})$ and its determinant $|J(\boldsymbol{x})|$ in the 2D Euclidean space are given by

$$J(\boldsymbol{x}) = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix},\ |J(\boldsymbol{x})| = x_\xi y_\eta - x_\eta y_\xi. \tag{2.2.17}$$

To ensure that the mesh does not tangle, the map (2.2.16) must be locally invertible, so that $|J(\boldsymbol{x})| \neq 0$ for all $\boldsymbol{\xi}$. In 2D, we can define the concept of tangling by first referring to the orientation of the mesh elements. In particular, a triangle element defined by three ordered points abc is *positively/negatively oriented* if the interior of the element lies to the left/right of the oriented segment ab, or equivalently bc or ca. Then, a mesh is *tangled* if it contains elements of opposite orientation, or equivalently if it displays overlapping elements [DS13], as visible in Figure 2-3.

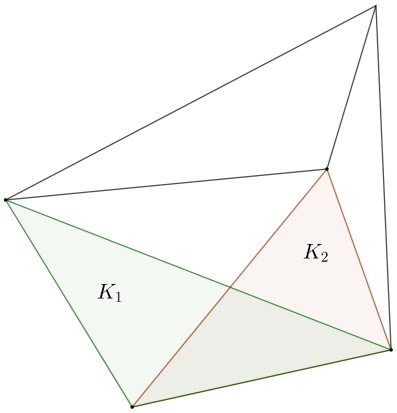


Figure 2-3: Tangled mesh with overlapping elements K_1 and K_2.

The OT condition aims at finding the map $\boldsymbol{x}(\boldsymbol{\xi})$ which minimises the distance

$$I[\boldsymbol{\xi}] = \int_{\Omega_c} |\boldsymbol{x}(\boldsymbol{\xi}) - \boldsymbol{\xi}|^2 \, \mathrm{d}\boldsymbol{\xi}. \tag{2.2.18}$$

This constraint seeks a mesh that equidistributes the monitor function $M(\boldsymbol{x})$ while staying as close to the uniform mesh as possible. It has been proved in [Caf90] and [Bre91] that from the OT (2.2.18) and equidistribution (2.2.16) constraint there exists a unique and regular map:

Theorem 2.1 ([Bre91] and [Caf90]) *There exists a unique, optimal map $\boldsymbol{x}(\boldsymbol{\xi})$ that satisfies the equidistribution principle* (2.2.16)*. The mapping has the same regularity as the monitor function $M(\boldsymbol{x})$. The mapping can be written as the gradient of a unique convex potential $\phi(\boldsymbol{\xi})$ up to a constant as*

$$\boldsymbol{x}(\boldsymbol{\xi}) = \nabla_{\boldsymbol{\xi}} \phi(\boldsymbol{\xi}), \ \Delta_{\boldsymbol{\xi}} \phi(\boldsymbol{\xi}) > 0.$$

The Jacobian of the mapping is then

$$\boldsymbol{J}(\nabla_{\boldsymbol{\xi}} \phi) = \begin{bmatrix} \phi_{\xi\xi} & \phi_{\xi\eta} \\ \phi_{\eta\xi} & \phi_{\eta\eta} \end{bmatrix} := D^2 \phi. \tag{2.2.19}$$

Replacing the value for the Jacobian of the mapping (2.2.19) for the convex mesh potential $\phi(\boldsymbol{\xi})$ into the equidistribution principle (2.2.16), it follows that $\phi(\boldsymbol{\xi})$ satisfies the Monge-Ampère equation

$$M\left(\boldsymbol{x}(\boldsymbol{\xi})\right) |D^2 \phi| = \theta. \tag{2.2.20}$$

The Monge-Ampère equation is a highly studied equation in differential geometry and physics. It has

numerous applications outside of grid generation, including meteorology and oceanography [BCW13], movement of materials around obstacles [Fel99], and optimal reflector design [CKO00]. The existence and regularity of its solutions has been a highly studied problem for a quadratic cost function [DF11; Fig07].

The equation (2.2.20) must be supplemented with suitable boundary conditions. Under the assumption of a convex two-dimensional region, it has been shown that it admits a unique convex solution [Del+08; Caf96].

2.3 The Finite Element method

Having constructed a mesh using one of the previous methods, we now consider the problem of solving a PDE on this mesh. The Finite Element (FE) method provides high flexibility for doing this, handling both structured and unstructured meshes, where in the first case each mesh node has the same connectivity (not for the latter). In the next Chapters, this will be applied to discretise both MMPDEs and physical PDEs. We will illustrate now finite elements for linear elements on triangular meshes, even though the approach generalises for any polygonal or polyhedral domain.

2.3.1 Model Problem

Let Ω denote a bounded domain in $\mathbb{R}^d,\ d \geq 1$. We define the Hölder spaces as

Definition 2.3.1 (Hölder Spaces) *Let $\alpha \in (0, 1]$. Then $C^{0,\alpha}(\Omega)$ is the subset of $C^0(\Omega)$ comprising all $f \in C^0(\Omega)$ such that:*

$$|f|_{C^{0,\alpha}(\Omega)} = \sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega,\boldsymbol{x}\neq\boldsymbol{y}} \frac{|f(\boldsymbol{x}) - f(\boldsymbol{y})|}{\|\boldsymbol{x} - \boldsymbol{y}\|^{\alpha}} < \infty, \tag{2.3.1}$$

where $\|\cdot\|$ is the Euclidean norm in Ω. For $f \in C^{0,\alpha}(\Omega)$, we define

$$\|f\|_{C^{0,\alpha}(\Omega)} = \|f\|_{C^0(\Omega)} + |f|_{C^{0,\alpha}(\Omega)}.$$

The Laplacian operator is $\Delta := \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$. For illustrative purposes, we seek the solution $u \in C^2(\Omega)$ of the strong form of the Poisson problem

$$-\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), \tag{2.3.2}$$

given the source term $f \in C^{0,\alpha}(\Omega)$. Throughout the thesis, we denote the boundary of Ω by $\partial\Omega$ or Γ.

Various boundary conditions can be applied to the PDE (2.3.2) such that the problem has a unique solution. The most commonly used boundary conditions are

- Dirichlet boundary conditions, or *essential boundary conditions*, are in the form:

$$u = u_D \quad \text{on } \partial\Omega. \tag{2.3.3}$$

- Neumann boundary conditions, or *natural boundary conditions*, appear when integration by parts is applied to the PDE. They specify the flux of the solution on the boundary $\partial\Omega$, where $\frac{\partial u}{\partial \boldsymbol{n}} = \nabla u \cdot \boldsymbol{n}$ and denotes the normal derivative at the boundary with normal vector n (see Figure 2-4):

$$k\frac{\partial u}{\partial \boldsymbol{n}} = u_N \quad \text{on } \partial\Omega. \tag{2.3.4}$$

- Robin boundary conditions are a weighted combination of both Dirichlet and Neumann boundary conditions:

$$\alpha u + \beta\frac{\partial u}{\partial \boldsymbol{n}} = u_R \quad \text{on } \partial\Omega, \tag{2.3.5}$$

  where α and β are two constant parameters.

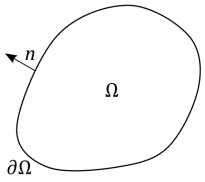


Figure 2-4: Representation of a domain Ω with boundary $\partial\Omega$ and normal vector $\boldsymbol{n}$.

For simplicity, we will only consider homogeneous Dirichlet boundary condition $u_D = 0$ on $\partial\Omega$ for the remainder of this Chapter.

2.3.2 Functional setting

In order to solve the weak formulation of Poisson's equation, we need to define the functional spaces that contains integrals of function derivatives. We start by introducing the Lebesgue space of square integrable functions on Ω as

$$\mathrm{L}^2(\Omega) = \left\{u : \int_\Omega |u(\boldsymbol{x})|^2 \,\mathrm{d}\boldsymbol{x} < +\infty\right\}, \tag{2.3.6}$$

equipped with the norm $\|u\|^2_{\mathrm{L}^2(\Omega)} := \int_\Omega |u(\boldsymbol{x})|^2 \,\mathrm{d}\boldsymbol{x}$.

The Hilbert space $\mathrm{H}^k(\Omega)$ of order $k \in \mathbb{N}$ includes solution of PDEs in the weak form. It is defined by

$$\mathrm{H}^k(\Omega) := \Big\{ u \in \mathrm{L}^2(\Omega) : \sum_{\alpha:|\alpha|\le k} \|D^\alpha u\|_{\mathrm{L}^2(\Omega)} < \infty \Big\}, \tag{2.3.7}$$

with norm $\|u\|^2_{\mathrm{H}^k(\Omega)} := \sum_{\alpha:|\alpha|\le k} \|D^\alpha u\|^2_{\mathrm{L}^2(\Omega)}$. The term α is a multi-index and the derivatives D^α are understood in weak sense [Eva10, §5.2.1].

Sobolev spaces of order (k, p), with $p \ge 1$, generalise Hilbert spaces of order k and are defined by

$$\begin{aligned} W^{k,p}(\Omega) &= \Big\{ u \in \mathrm{L}^p(\Omega) : D^\alpha u \in \mathrm{L}^p(\Omega) \text{ for } |\alpha| \le k \Big\}, \\ \|u\|^2_{W^{k,p}(\Omega)} &:= \sum_{\alpha:|\alpha|\le k} \|D^\alpha u\|^2_{\mathrm{L}^p(\Omega)} \,. \end{aligned} \tag{2.3.8}$$

For $p = \infty$ we have

$$\begin{aligned} W^{k,\infty}(\Omega) &= \Big\{ u \in \mathrm{L}^\infty(\Omega) : \max_{\alpha:|\alpha|\le k} \operatorname{ess\,sup} |D^\alpha u(\boldsymbol{x})| < +\infty \Big\}, \\ \|u\|_{W^{n,\infty}(\Omega)} &:= \max_{\alpha:|\alpha|\le k} \operatorname{ess\,sup} |D^\alpha u(\boldsymbol{x})|. \end{aligned} \tag{2.3.9}$$

For the remainder of this Chapter, we denote the Hilbert space $V = \mathrm{H}^1(\Omega)$ as a linear function space with inner product $\langle \cdot, \cdot \rangle_V : V \times V \to \mathbb{R}$, which takes the expression:

$$\langle u, v \rangle_V = \int_\Omega uv \,\mathrm{d}\boldsymbol{x} + \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}\boldsymbol{x} = \langle u, v \rangle_{\mathrm{L}^2(\Omega)} + \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}\boldsymbol{x}, \tag{2.3.10}$$

and the corresponding norm $\|\cdot\|_V = \langle \cdot, \cdot \rangle_V^{1/2}$.

Before applying the finite element discretisation, we need to formulate the problem (2.3.2) in a variational or weak form, where we combine the PDE and the boundary conditions into one expression. The homogeneous Dirichlet boundary condition for the solution u is embedded in the function space $V_0 = \{u \in \mathrm{H}^1(\Omega) : u|_{\partial\Omega} = 0\}$.

We first multiply both sides of equation (2.3.2) by a smooth test function $v \in V_0$ such that

$$-\int_\Omega \Delta u v \,\mathrm{d}\boldsymbol{x} = \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}\boldsymbol{x} - \int_{\partial\Omega} \frac{\partial u}{\partial n} v \,\mathrm{d}s \quad \forall v \in V_0. \tag{2.3.11}$$

Applying integration by parts in equation (2.3.11) we obtain

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x} - \int_{\partial\Omega} \frac{\partial u}{\partial n} v \, \mathrm{d}s = \int_\Omega f v \, \mathrm{d}\boldsymbol{x} \quad \forall v \in V_0, \tag{2.3.12}$$

and with $v = 0$ on $\partial\Omega$ the weak formulation of the Poisson problem reads as

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x} = \int_\Omega f v \, \mathrm{d}\boldsymbol{x} \quad \forall v \in V_0. \tag{2.3.13}$$

We note that in the strong formulation the solution u is required to be in $C^2(\Omega)$. The weak form reduces the smoothness of the solution as it needs only the first weak derivatives of u.

In the abstract form, the weak formulation is defined as

$$\begin{aligned} &\text{Find } u \in V \text{ such that:} \\ &\mathscr{A}(u, v) = l(v) \quad \forall v \in V, \end{aligned} \tag{2.3.14}$$

where $\mathscr{A}(\cdot, \cdot)$ is a continuous bilinear form on $V \times V$ and $l(\cdot)$ is a continuous linear form on V. When solving a PDE of physical phenomena is crucial to have a well-posed problem. This means that the problem has a solution, the solution is unique and changes continuously with the initial conditions. There are several properties that ensure the well-posedness of the problem:

Definition 2.3.2 (Continuity [BS08, Definition 2.5.2]) *A bilinear form $\mathscr{A}(\cdot, \cdot)$ is continuous on $V \times V$ if there exists a positive constant M such that*

$$\mathscr{A}(v, w) \leq M \|v\|_V \|w\|_V \quad \forall v, w \in V. \tag{2.3.15}$$

Definition 2.3.3 (Coercivity [BS08, Definition 2.5.2]) *A bilinear form is coercive in V if there exists a positive constant α such that for any $v \in V$*

$$\mathscr{A}(v, v) \geq \alpha \|v\|_V^2 \quad \forall v \in V. \tag{2.3.16}$$

Given $u \in V$, we recall that a continuous linear functional on V can be defined by $l_u(v) = \langle u, v \rangle$. We show that the converse is also true.

Theorem 2.2 (Riesz Representation Theorem [BS08, Theorem 2.4.2]) *Any continuous linear functional l on a Hilbert space V can be represented uniquely as*

$$l(v) = \langle u, v \rangle,$$

for some $u \in V$. *Furthermore, we have*

$$\|u\|_V = \|l\|_{V'},$$

where V' *is the dual space of* V.

These conditions ensure the existence and uniqueness of the solution of problem (2.3.13), as stated in the Lax-Milgram Theorem:

Theorem 2.3 (Lax-Milgram [BS08, Theorem 2.7.7]) *Let* V *be a Hilbert space, let* $\mathscr{A}(\cdot,\cdot)$ *be a symmetric continuous, coercive bilinear form on* $V \times V$ *and let* $l \in V'$ *be a continuous linear functional. Then, there exists a unique* $u \in V$ *such that* (2.3.14) *satisfies:*

$$\mathscr{A}(u, v) = l(v) \quad \forall v \in V. \tag{2.3.17}$$

2.3.3 The Ritz and Galerkin methods for elliptic problems

We showed that original Dirichlet problem has been restated in the weak formulation, which admits a unique solution under certain conditions. The approximation of the weak problem defines the Ritz and the Galerkin method. Both of them are presented for a conforming FE discretisation, so that the solution space V is replaced by a finite dimensional subspace $\mathbb{V} \subset V$, with $\dim(\mathbb{V}) = N$. We denote $\mathscr{A}_h(\cdot,\cdot) : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$ an approximation of $\mathscr{A}(\cdot,\cdot)$ and $l_h(\cdot) : \mathbb{V} \to \mathbb{R}$ an approximation of $l(\cdot)$.

Provided that $\mathscr{A}_h(\cdot,\cdot)$ is symmetric, the Ritz method is stated as the minimisation problem:

$$\begin{aligned} &\text{Find } u_h \in \mathbb{V}, \mathbb{V} \subset V, \text{ such that :} \\ &J(u_h) \leq J(v_h) \quad \forall v_h \in \mathbb{V}, \\ &\text{with } J(v_h) = \frac{1}{2}\mathscr{A}_h(v_h, v_h) - l(v_h). \end{aligned} \tag{2.3.18}$$

The Galerkin method uses a similar approach as for Ritz method, except that the abstract problem does not require the symmetry of the bilinear form. Therefore we might not endow $\mathbb{V}$ with a norm defined from the scalar product based on $\mathscr{A}_h(\cdot,\cdot)$. The approximate weak Galerkin method is stated as

$$\begin{aligned} &\text{Find } u_h \in \mathbb{V}, \mathbb{V} \subset V, \text{ such that :} \\ &\mathscr{A}_h(u_h, v_h) = l_h(v_h) \quad \forall v_h \in \mathbb{V}, \end{aligned} \tag{2.3.19}$$

with $\mathscr{A}_h(\cdot,\cdot)$ being a coercive continuous bilinear form and $l_h(\cdot)$ being a continuous linear form.

In particular, the approximation of the Poisson problem in (2.3.13) reads as

$$\begin{aligned} &\text{Find } u_h \in \mathbb{V}, \ f \in \mathrm{L}^2(\Omega), \text{ such that :} \\ &\int_\Omega \nabla u_h \cdot \nabla v_h \, \mathrm{d}\boldsymbol{x} = \int_\Omega f v_h \, \mathrm{d}\boldsymbol{x} \quad \forall v_h \in \mathbb{V}. \end{aligned} \tag{2.3.20}$$

The approximation space is defined in terms of piecewise polynomials of total degree $p \geq 0$. We denote the polynomial space for each element K of the mesh $\mathscr{T}$ as $\mathbb{P}^p(K)$. Finally, the approximation space is written as $\mathbb{V} = \left\{ v \in C^0(\Omega) \cap V_0 : v|_K \in \mathbb{P}^p(K), \ \forall K \in \mathscr{T} \right\}$.

The two proposed methods will be employed to derive a loss function for a neural network to solve Poisson's equation in Chapter 5.

2.3.4 Finite Element spaces

In the previous section we defined the Ritz and Galerkin method for the approximation of the solution of the PDE. The following results will refer to the Galerkin discretisation in (2.3.19). Provided this abstract framework, which allows us to seek approximate solutions to PDEs, we need to chose the approximate space $\mathbb{V}$ and construct a basis $\mathcal{B} = (\phi_1, \ldots, \phi_N)$ of $\mathbb{V}$ on which the discrete solution is decomposed:

$$u_h = \sum_{j=1}^{N} u_j \phi_j, \tag{2.3.21}$$

where $N = \dim(\mathbb{V})$, $\{u_j\}$ are called *global degrees of freedom* and $\{\phi_j\}$ are named *global shape functions*. The approximate space $\mathbb{V}$ is constructed with an admissible mesh $\mathscr{T}$ from a tessellation of the domain Ω. We can now state the following Lemma that relates the approximate solution u_h and the solution u of problem (2.3.19):

Lemma 2.4 (Galerkin orthogonality [BS08, Lemma 2.17]) *Let V be a Hilbert space and $\mathbb{V}$ a finite dimensional subspace of V. We denote by $u \in V$, $u_h \in \mathbb{V}$ the solution to problem* (2.3.14) *and to approximate problem* (2.3.19)*, respectively. We have then*

$$\mathscr{A}_h(u - u_h, v_h) = 0 \quad \forall v_h \in \mathbb{V}. \tag{2.3.22}$$

The relation (2.3.22) indicates that the error $u - u_h$ is orthogonal to every element of the approximating space $\mathbb{V}$ in the inner product induced by $\mathscr{A}_h(\cdot, \cdot)$.

Let u be the solution to the variational problem (2.3.14) and u_h be the solution to the approximation problem (2.3.19). We now want to estimate the error $\|u - u_h\|_V$. We do so by the following Lemma:

Lemma 2.5 (Céa's Lemma [BS08, Theorem 2.8.1]) *Given the assumption in Lemma 2.4 we have*

$$\|u - u_h\|_V \leq \frac{M}{\alpha} \|u - v_h\|_V \quad \forall v_h \in \mathbb{V}, \tag{2.3.23}$$

with $M > 0$ the continuity constant and $\alpha > 0$ the coercivity constant.

Moreover, the bound (2.3.23) *implies that*

$$\|u - u_h\|_V \leq \frac{M}{\alpha} \inf_{v_h \in \mathbb{V}_h} \|u - v_h\|_V \quad \forall v_h \in \mathbb{V}. \tag{2.3.24}$$

If we work in the energy norm $\|\cdot\|_E = \mathscr{A}_h(\cdot, \cdot)^{1/2}$, we obtain the following result:

Corollary 2.6 (Estimate in the energy norm) *Given the assumptions in Theorem 2.5, the following inequality holds:*

$$\|u - u_h\|_E \leq \|u - v_h\|_E \quad \forall v_h \in \mathbb{V}. \tag{2.3.25}$$

Thus the FEM computes the best approximation of u in the energy norm. We can use (2.3.23) to obtain a rate of convergence and estimate the error $u - u_h$. We do this by choosing $v_h \in \mathbb{V}$ so that the right hand side of (2.3.23) can be easily computed and is a good estimate of the error.

The discontinuous Galerkin Finite Element method

Discontinuous Galerkin (dG) methods are locally conservative, stable, and high-order accurate methods that can easily handle complex geometries, unstructured meshes, and approximations that have polynomials of different degrees in different elements. These properties, which render them ideal to be used with hp-adaptive strategies, have brought these methods into the field of computational fluid dynamics, second-order elliptic problems, elasticity, and Korteweg-deVries equations [Arn+02; Arn+00].

The definition of the dG method needs quantities that link the values of the approximate solution u_h between different elements of the mesh $\mathscr{T}$. All degrees of freedom of a discontinuous finite element

are internal to the element, which means that no global continuity is imposed by these elements. This is illustrated in Figure 2-5.

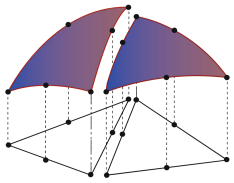


Figure 2-5: Visualisation of discontinuous quadratic Lagrange elements in 2D [LW10].

We will define more precisely the quantities used for the dG discretisation in Chapter 3 and Chapter 4.

2.3.5 Mesh and Quality measure

We defined in §2.1 and §2.2 different methods to compute a mesh in 1D and 2D. In this section, we provide other definitions in order to characterise the regularity and the level of adaptivity of a mesh.

Mesh regularity is a geometric property indicating how close mesh elements are to being equilateral. An example of such metric is the *shape regularity*, that will be defined based on the *mesh size* and *in-diameter* of the mesh elements.

Mesh adaptivity characterises how well a mesh adapts to the solution of a PDE and is thus a property related to the physical problem being solved. The level of adaptivity can be quantified via equidistribution and alignment, that is via a given matrix monitor function $M(\boldsymbol{x})$ in the context of r-adaptivity, as showed in §2.2.

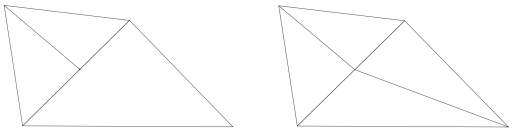
Definition 2.3.4 (Conforming mesh in 2D) *Let $\Omega \subset \mathbb{R}^2$ be a 2D polygonal domain, we define $\mathcal{T}$ a conforming triangulation as a finite family $\{K_i\}_{i\in\mathbb{N}}$ of cells satisfying:*

1. *$K \in \mathcal{T}$ implies K is an open triangle.*
2. *For any $K, J \in \mathcal{T}$ we have that $\overline{K} \cap \overline{J}$ is either empty, a vertex, or a common edge of both $\overline{K}$ and $\overline{J}$.*
3. *$\cup_{K\in\mathcal{T}} \overline{K} = \overline{\Omega}$.*

For each $K \in \mathcal{T}$ we introduce the *cell diameter*

$$h_K = diam(K) := sup_{\boldsymbol{x},\boldsymbol{y}\in K} d(\boldsymbol{x},\boldsymbol{y}), \tag{2.3.26}$$

where $d(\boldsymbol{x},\boldsymbol{y})$ is the (Euclidean) distance between $\boldsymbol{x}$ and $\boldsymbol{y}$.



a Triangulation with hanging node.

b Conforming triangulation.

Figure 2-6: Visualisation of a non-conforming triangulation with hanging node and a conforming triangulation.

The *in-diameter* of K is

$$\rho_K := \sup\{\text{diam}(B) : B \text{ is an open ball contained in } K\}. \tag{2.3.27}$$

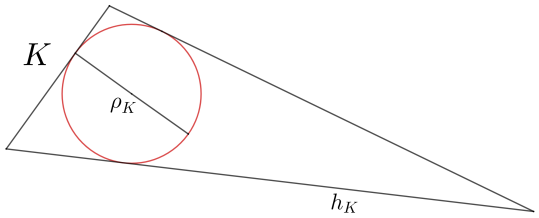


Figure 2-7: Visualisation of the diameter and in-diameter for a triangular element K.

We will also assume in Chapter 4 that all the meshes generated by h-refinement are *shape regular*:

Definition 2.3.5 (Shape regularity) *Let $\mathcal{G} = \{\mathcal{T}_i\}_{i \in \mathbb{N}}$ be a family of meshes, then $\mathcal{G}$ is characterised by the shape regularity $\mu(\mathcal{T})$ if*

$$\mu(\mathcal{T}) := \min_{K \in \mathcal{T}_i} \frac{h_K}{\rho_K} > 0. \tag{2.3.28}$$

Additionally, we define the mesh size by

Definition 2.3.6 (Mesh size)

$$h_{\mathcal{T}} := \max_{K \in \mathcal{T}} (diam(K)). \tag{2.3.29}$$

We denote the maximum and minimum *mesh width* of $\mathcal{T}$ by

$$\overline{h} = \max_{K \in \mathcal{T}} h_K, \quad \underline{h} = \min_{K \in \mathcal{T}} h_K.$$

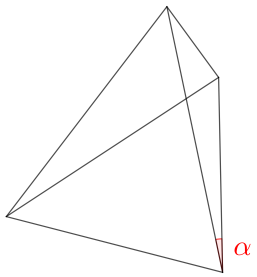


Figure 2-8: The mesh is shape regular if the minimum angle α among all elements is uniformly bounded from below by $\mu(\mathscr{T})$.

Finally, we define the level of mesh adaptivity in 2D in terms of *skewness*. This quantity indicates how far mesh elements are from being equilateral. For triangular elements, the skewness can be computed as the ratio between the minimum and maximum vertex angle. A more general indicator employs the local map $Q_K : \widehat{K} \to K$, with $\widehat{K} \in \mathscr{T}_c$ and $K \in \mathscr{T}$.

In order to visually interpret this quantity, we follow the steps in [BRW15]. Let $\boldsymbol{J}_K$ be the Jacobian of the map $\boldsymbol{x} : \Omega_c \to \Omega$ as defined in equation and $\widehat{K}$ be a circular set in Ω_c centered at $\boldsymbol{\xi}_0$ and

$$\widehat{K} = \{\boldsymbol{\xi} : (\boldsymbol{\xi} - \boldsymbol{\xi}_0)^T (\boldsymbol{\xi} - \boldsymbol{\xi}_0) = \hat{r}^2\},$$

where the radius $\hat{r} \propto (|\Omega_c|/N)^{1/2}$. The linearisation about $\boldsymbol{\xi}_0$ is given by

$$\boldsymbol{x}(\boldsymbol{\xi}) = \boldsymbol{x}(\boldsymbol{\xi}_0) + \boldsymbol{J}(\boldsymbol{\xi}_0)(\boldsymbol{\xi} - \boldsymbol{\xi}_0) + O(|\boldsymbol{\xi} - \boldsymbol{\xi}_0|^2).$$

The corresponding image set $K = \boldsymbol{x}(\widehat{K})$ in Ω is approximately given by

$$K = \left[\boldsymbol{x} : (\boldsymbol{x} - \boldsymbol{x}(\boldsymbol{\xi}_0))^T \boldsymbol{J}^{-T} \boldsymbol{J}^{-1} (\boldsymbol{x} - \boldsymbol{x}(\boldsymbol{\xi}_0)) = \hat{r}^2\right].$$

Let the singular value decomposition (SVD) of $\boldsymbol{J}$ be

$$\boldsymbol{J} = U\Sigma Z^T,$$

where $U = [\boldsymbol{u}_1, \boldsymbol{u}_2]$ (the left singular vectors), $V = [\boldsymbol{v}_1, \boldsymbol{v}_2]$ (the right singular vectors) are orthogonal matrices, and λ_1, λ_2 are the positive singular values of the matrix $\Sigma = \mathrm{diag}(\lambda_1, \lambda_2)$. It follows that

$$K = \left[\boldsymbol{x} : (\boldsymbol{x} - \boldsymbol{x}(\boldsymbol{\xi}_0))^T U\Sigma^{-2}U^T (\boldsymbol{x} - \boldsymbol{x}(\boldsymbol{\xi}_0)) = \hat{r}^2\right],$$

so that the orientation of K is determined by the left singular vectors of U, and size and shape by the singular values σ_1 and σ_2. We quantify the size, shape and orientation of an element K using the singular values and left singular vectors of $\boldsymbol{J}$ and the eigenvalues and eigenvectors of the *metric tensor* $\mathcal{M} = \boldsymbol{J^{-T}J^{-1}} = U\Sigma^{-2}U^T$. The eigenvectors of $\mathcal{M}$ are u_1 and u_2 and the eigenvalues σ_1, σ_2.

Finally, the skewness of an element K is defined as

Definition 2.3.7 (Skewness)

$$Q_K := \frac{tr(\boldsymbol{J^T J})}{2\det(\boldsymbol{J^T J})^{1/2}} = \frac{1}{2}\left(\frac{\lambda_1}{\lambda_2} + \frac{\lambda_2}{\lambda_1}\right). \tag{2.3.30}$$

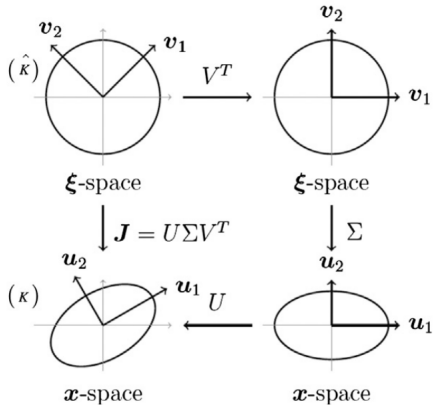


Figure 2-9: The two-dimensional mapping of an element $\hat{K}$ (a circle) in Ω_c, to a physical mesh element K (an ellipse) in Ω, under $\boldsymbol{x}(\boldsymbol{\xi})$. The skewness of the transformed element is evident from the degree of compression and stretching of the ellipse [BRW15].

We define the *global quality measure* (global skewness) as

$$Q = \max_{K \in \mathcal{T}} Q_K. \tag{2.3.31}$$

We will compute the skewness of the Optimal Transport mesh elements in Chapter 4, and show that it is independent on their location and the dimension of the mesh.

2.3.6 Error analysis for FEM

The goal of this section is to bound the error $e_h := u - u_h$ arising from the Finite Element discretisation in a Sobolev norm and relate it to the mesh properties defined in the previous section. We will define two categories of error estimators based on the knowledge of the true solution u of a prescribed problem. These are named *a-priori* and *a-posteriori* error estimators.

1 A-priori error estimator A-priori estimators provide useful information on the asymptotic behaviour of the approximate solution. The most important property for any conforming finite element formulation, that is with $\mathbb{V} \subset V$, is the inequality (2.3.24) in Céa's Lemma:

$$\|u - u_h\|_V \leq \frac{M}{\alpha} \|u - v_h\|_V \quad \forall v_h \in \mathbb{V},$$

which gives a control on the discretisation error using the norm in the solution space V. Since v_h can be chosen as an interpolant, Céa's Lemma asserts that the error of the finite element solution measured in the V norm is of the same order as the interpolation error. Choosing $V \subset \mathrm{H}^1(\Omega)$ and employing interpolation estimates, it turns out that the error measured in the H^1 norm is

$$\|u - u_h\|_{\mathrm{H}^1(\Omega)} \leq c h_{\mathscr{T}}^p \|u\|_{\mathrm{H}^{p+1}(\Omega)}, \tag{2.3.32}$$

where $p \geq 1$, c is a stability and interpolation constant which does not depend on the interpolation space and $h_{\mathscr{T}}$ is the mesh size in Def.2.3.6. We also assume that the solution $u \in \mathrm{H}^{p+1}(\Omega)$.

Furthermore, we have for the error in the L^2 norm with $p \geq 1$

$$\|u - u_h\|_{\mathrm{L}^2(\Omega)} \leq c h_{\mathscr{T}}^{p+1} \|u\|_{\mathrm{H}^{p+1}(\Omega)}, \tag{2.3.33}$$

which means that the convergence rate of the solution is $O(h_{\mathscr{T}}^p)$ and $O(h_{\mathscr{T}}^{p+1})$ in the H^1 and L^2 norm, respectively [BS08]. The dG method has the same order of convergence in the L^2 norm [Arn+00].

2 A-posteriori error estimator The purpose of an a-posteriori error estimator is to provide bounds for the solution error in a specified norm. Ideally, an effective error estimator should be close to the actual (unknown) error and be asymptotically correct, in the sense that with increasing mesh resolution the convergence rate should be the same as the actual error [GB05].

Explicit error estimators involve direct computation of the interior element residuals and the jumps at the element boundaries to find an estimate for the error in the energy norm.

Let u and u_h be the solution of (2.3.13) and (2.3.20), we start with the error representation

$$\mathscr{A}_h(e_h, v) = l_h(v) - \mathscr{A}_h(u_h, v), \tag{2.3.34}$$

which holds true for arbitrary test functions $v \in V$. If the domain integral is split into the contribution for each element $K \in \mathscr{T}$, then equation (2.3.34) can be rewritten as

$$\mathscr{A}_h(e_h, v) = \sum_{K \in \mathscr{T}} \Big(\int_K f v \, \mathrm{d}\boldsymbol{x} - \int_K \nabla u_h \cdot \nabla v \, \mathrm{d}\boldsymbol{x} \Big) \quad \forall v \in V. \tag{2.3.35}$$

Applying integration by parts to the last term of equation (2.3.35) leads to

$$\mathscr{A}_h(e_h, v) = \sum_{K \in \mathscr{T}} \Big(\int_K R v \, \mathrm{d}\boldsymbol{x} + \sum_{e \in \Gamma} \int_e J v \, \mathrm{d}\boldsymbol{x} \Big) \quad \forall v \in V, \tag{2.3.36}$$

where R is the interior element residual and defined as $R := (f + \Delta u_h)|_K$. The jump term J of the gradient across element edge e is

$$J := \begin{cases} \boldsymbol{n}_e \cdot \nabla u_h + \boldsymbol{n}'_e \cdot \nabla u'_h & \text{if } e \subset \Gamma_I \\ 0 & \text{if } e \subset \Gamma_D \end{cases}, \tag{2.3.37}$$

where Γ_I is the set of internal edges. In (2.3.37) the edge e is shared between K and K', such that $e = K \cap K'$.

Using the Galerkin orthogonality condition we can introduce a piecewise linear interpolant $\Pi_h v$ in equation (2.3.36). Using interpolation inequalities with constant c_I [AO97] and the Cauchy-Schwarz inequality we finally obtain the error bound

$$\|e_h\|_E^2 \leq c_I \Big(\sum_{K \in \mathscr{T}} h_K^2 \, \|\mathcal{R}\|^2_{\mathrm{L}^2(K)} + \sum_{e \in \Gamma} h_K \, \|J\|^2_{\mathrm{L}^2(e)} \Big) \tag{2.3.38}$$

In practice, the term 2.3.38 is regrouped so as to define the *local a-posteriori estimate*:

$$\|e_h\|_E^2 \leq \sum_{K \in \mathscr{T}} \Big(c_1 h_K^2 \, \|\mathcal{R}\|^2_{\mathrm{L}^2(K)} + c_2 h_K \, \|J\|^2_{\mathrm{L}^2(\partial K)} \Big) := \sum_{K \in \mathscr{T}} \eta_K^2. \tag{2.3.39}$$

The local a-posteriori estimate η_K^2 can prove very useful for h-adaptive schemes, as it can detect cells that need refining. This process ultimately leads to the equidistribution of the local a-posteriori estimator. We will apply such estimate in L^2 and L^∞ norm in Chapter 4 for the Poisson problem. We will also link η_K to the equidistribuion condition in the context of r-adaptivity.

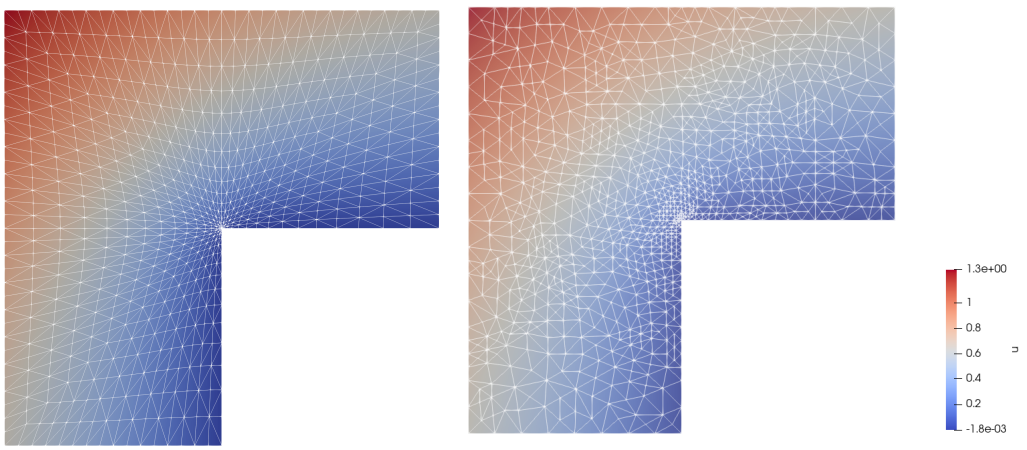
2.3.7 Non-convex domains

Solutions of elliptic boundary value problems defined in domains Ω with re-entrant corners have singular behaviour at these corners. This occurs even when data of the underlying problem are very smooth. Such

singular behaviour affects the accuracy of the Finite Element method throughout the whole domain. For example, it is well known that the solution u of Poisson's equation defined on a polygonal domain with re-entrant corners has a singular function representation.

Given a finite element approximation of u, it has been proved that $u \in \mathrm{H}^k(\Omega)$ for $k < 1 + \pi/\omega$, where ω is the maximum of the re-entrant angles [Gri11]. This lack of regularity affects the accuracy of the finite element approximation. In particular, standard dG linear finite element methods on a quasi-uniform triangulation yield $\mathcal{O}(h_{\mathcal{T}}^{(\pi/\omega)-\varepsilon})$ and $\mathcal{O}(h_{\mathcal{T}}^{(2\pi/\omega)-\varepsilon})$ accuracy for any $\varepsilon > 0$ in the H^1 and L^2 norm, respectively. Adapted methods have been developed in order to retain the optimal convergence orders. Local mesh grading was introduced by [OR68; BKP79]. These papers treat only the case of piecewise linear finite elements. Other methods include the singular function method (augmenting technique) described in [ZSG02; Str08].

In Chapter 4, we will address Poisson's equation in non-convex domains using the Symmetric Interior Penalty dG (SIP-dG) finite element method. We will define weighted Sobolev spaces in order to ensure the existence and uniqueness of the solution. We will show numerically that the optimal convergence rate is recovered when the SIP-dG method is combined with h- and r- adaptive strategies. The resulting meshes are shown in figure 2-10.



a Optimal Transport strategy. b h-refinement method.

Figure 2-10: Solution of Poisson's equation on a L-shaped domain with homogeneous Dirichlet boundary conditions using OT based r-adaptivity and h-refinement.

Chapter 5 will deal with the same problem but by using the deep learning framework. In particular, we will compare the accuracy of the solution of the neural network output using randomly sampled training points and the OT-based mesh vertices derived in §4.4.5 of Chapter 4.

2.4 Background on neural networks

Here we give a simple introduction of the main concepts behind deep learning and PINNs. In general context, the training procedure of a deep neural network (DNN) consists of three components:

- Neural network structure.
- Loss function.
- Optimisation method for training the DNN.

At the end of the training process, the DNN will construct an approximated solution whose regularity depends on the network structure. In the next section, we will describe all the network components in more detail.

2.4.1 Network structure

The architecture of an artificial neural network is based on the concept of the biological neuron. An important role is played by the flow of information that is passed through interconnected neurons arranged in layers. A DNN can then be regarded as a concatenated structure of layers as visualized in Figure 2-11. This can be also referred to as a feedforward neural network:

Definition 2.4.1 (feedforward neural network (FNN)) *Let L, d, $n_1, \dots, n_L \in \mathbb{N}$ and $n_0 := d$. A L-layer feedforward neural network $u_\Theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ with affine linear maps $A_l : \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$, $\boldsymbol{x} \to A_l(\boldsymbol{x}) = \boldsymbol{W}_l \boldsymbol{x} + \boldsymbol{b}_l$ with $\boldsymbol{W}_l \in \mathbb{R}^{n_l \times n_{l-1}}, \boldsymbol{b}_l \in \mathbb{R}^{n_l}$ and a bounded continuous activation functions $\sigma_l : \mathbb{R} \to \mathbb{R}, l = 1, \dots, L$ is defined as*

- *Input Layer:* $u_\Theta^0(\boldsymbol{x}) = \boldsymbol{x} \in \mathbb{R}^d$.
- *Hidden Layers:* $u_\Theta^l(\boldsymbol{x}) = \sigma_l\big(\boldsymbol{W}_l u_\Theta^{l-1}(\boldsymbol{x}) + \boldsymbol{b}_l\big) \in \mathbb{R}^{n_l}$ *for* $1 \le l \le L-1$.
- *Output Layer:* $u_\Theta(\boldsymbol{x}) = \boldsymbol{W}_L u_\Theta^{L-1}(\boldsymbol{x}) + \boldsymbol{b}_L \in \mathbb{R}^{n_L}$,

where the activation functions are used component-wise. Here, d is the dimension of the input layer, L denotes the number of layers, also called depth of u_Θ, $n_1, \dots, n_{L-1}$ denote the number of neurons for each of the $L-1$ hidden layers, also called width of the respective layer. If $n_1 = \cdots = n_{L-1}$, then n_i is called width of u_Θ for $i \in \{1, \dots, L-1\}$. The dimension of the output layer is n_L. The matrices $\boldsymbol{W}_l$ contain the network weights and the vectors $\boldsymbol{b}_l$ contain the network biases. We encode them into one parameter $\Theta_l \in \mathbb{R}^{n_l \times n_{l-1}+n_l}$ for $l \in \{1, \dots, L\}$. Finally, all the parameters are encoded into the vector $\Theta := (\Theta_1, \dots, \Theta_L) \in \mathbb{R}^c$, with $c = \sum_{l=1}^{L} n_l(n_{l-1}+1)$, by which we specify the dependence of the network with the notation $u_\Theta(\boldsymbol{x})$.

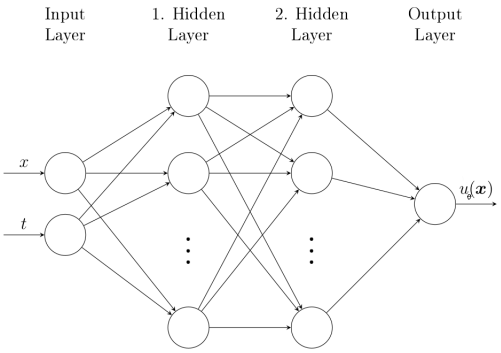


Figure 2-11: FNN with two hidden layers and $\boldsymbol{x} = (x, t)^T \in \mathbb{R}^2$ as input.

The graph of a FNN is acyclic and has only edges which follow the direction from input to output. The activation function determines whether a neuron is activated or inhibited. Moreover, it is responsible for the nonlinear transformation of the input which prevents the model from being a linear regression model. Commonly used activation functions are displayed in Figure 2-12.

Each activation function has its advantages and disadvantages [Nwa+18]. We will use the *hyperbolic tangent function*: $tanh(x) = \frac{sinh(x)}{coshx}$ in Chapter 5 because it is infinitely continuously differentiable, so that the approximate solution is also sufficiently smooth.

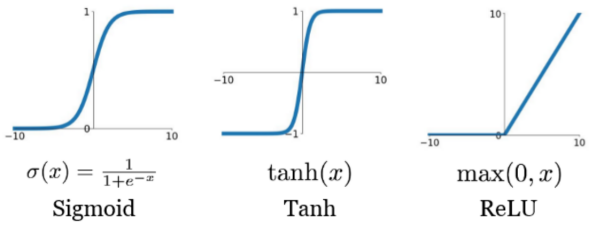


Figure 2-12: Examples of activation functions used in deep learning.

2.4.2 Training and Loss

The training of the network parameters corresponds to minimising a specified loss function. For a given input $\boldsymbol{x}$ the loss function describes a measurement of the error between the network output $u_\Theta(\boldsymbol{x})$ and the target function $u(\boldsymbol{x})$. There exist several loss functions that are used for machine learning applications according to which task they have to solve. The mean squared error (MSE) loss is one of the most common loss functions:

Definition 2.4.2 (Mean squared error (MSE) Loss [GBC16]) *Let $u_\Theta : \mathbb{R}^d \to \mathbb{R}^{n_L}$, with $d, n_L \in \mathbb{N}$, be a FNN as in Def.2.4.1. Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and $u : \Omega \to \mathbb{R}^{n_L}$ be the target function that the FNN aims to approximate. Then, for a given set $\{\boldsymbol{x}_i\}_{i=1}^{N_r}$ of N_r training points the mean squared*

error loss function $\mathcal{L}_{MSE} : \mathbb{R}^c \to \mathbb{R}$ *is defined as*

$$\mathcal{L}_{MSE}(\Theta) = \frac{1}{N_r} \sum_{i=1}^{N_r} \Big(u_\Theta(\boldsymbol{x}_i) - u(\boldsymbol{x}_i) \Big)^2.$$

Due to the squaring, larger errors are penalised more heavily than smaller ones. In addition to that, quadratic functions are preferred over other mappings, (e.g. $|\cdot|$*), since they are differentiable in each point.*

The differentiability of the loss function follows from the differentiability of the activation function and the fact that a quadratic function is differentiable as well. The training process of the network under the MSE leads to the network u_{Θ^*}, which can be viewed as the result of the optimisation problem

$$u_{\Theta^*} = \arg\min_{u_\Theta} \mathcal{L}_{MSE}(\Theta). \tag{2.4.1}$$

The minimisation can be done using a gradient descent approach. The method of steepest descent described in Algorithm 1 is the simplest gradient method for optimisation [Yua06]. However, for non-convex functions it is possible that the algorithm fails to converge towards the global minimum. The main risk is that the algorithm converges more slowly and possibly towards a local minimum, independently on the choice of the learning rate. Therefore, this method is rarely used in practice.

Algorithm 1 Steepest Descent

Require: Objective function $\mathcal{L}(\Theta)$ and initialisation of Θ_0
Ensure: Θ_k
1: Initialise iteration step $k := 0$
2: Compute stepsize $\alpha_k > 0$ via line search
3: **while** Θ_k not converged **do**:
4: $\quad k \leftarrow k + 1$
5: $\quad$ Compute gradient $g_k = \nabla_\Theta \mathcal{L}(\Theta_{k-1})$
6: $\quad$ Compute new parameters $\Theta_k = \Theta_{k-1} - \alpha_{k-1} g_k$
7: $\quad$ Update stepsize $\alpha_k \leftarrow \alpha_{k-1}$
8: **end while**

In this thesis, we consider a *stochastic gradient descent* (SGD) approach. Unlike gradient descent, a SGD method only takes small subsets of the training dataset, so-called mini-batches, in each optimisation step. This yields only approximations of the actual gradients leading to a noisy gradient in total. Due to this noise, the method is capable of escaping possible local minima. If the algorithm reaches a local minimum, it is likely to leave it again due to the momentum and the algorithm continues optimizing.

The SGD method we are dealing with is called *Adam* (Adaptive Moment estimation) [KB14] and is described in Algorithm 2. This only requires first-order gradients and the magnitudes of parameter updates are invariant to rescaling of the gradient. Adam tackles the SGD problems by using estimations

of first and second moments of the gradient to adapt the learning rate for the network parameters in each update.

Algorithm 2 Adam

Require: Stepsize α, decay rates $\beta_1, \beta_2 \in [0, 1)$
Require: Initialisation Θ_0 and objective function $\mathcal{L}(\Theta)$
Ensure: Θ_k
1: Initialise iteration step $k := 0$
2: Initialise 1st moment vector $m_0 := 0$
3: Initialise 2nd moment vector $v_0 := 0$
4: **while** Θ_k not converged **do**:
5: $k \leftarrow k + 1$
6: Compute gradient $g_k = \nabla_\Theta \mathcal{L}(\Theta_{k-1})$
7: Update 1st moment estimate $m_k \leftarrow \beta_1 m_{k-1} + (1 - \beta_1) g_k$
8: Update 2nd moment estimate $v_k \leftarrow \beta_2 v_{k-1} + (1 - \beta_2) g_k^2$
9: Compute corrected 1st moment estimate $\hat{m}_k = \frac{m_k}{1-\beta_1}$
10: Compute corrected 2nd moment estimate $\hat{v}_k = \frac{v_k}{1-\beta_2}$
11: Compute new parameters $\Theta_k = \Theta_{k-1} - \alpha \frac{\hat{m}_k}{\sqrt{\hat{v}_k} + \varepsilon}$
12: **end while**

In Algorithm (2) g_k^2 denotes the element-wise square. The authors recommend $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$ and $\varepsilon = 10^{-8}$ as default parameters [KB14].

In deep learning, the gradient $\nabla \mathcal{L}(\Theta)$ is obtained through *back-propagation*, which computes the gradient by using the chain rule while propagating backwards through the network [Bis95, pp.140-148]. This technique is implemented efficiently in several machine learning frameworks, such as TensorFlow and PyTorch [Pas+19] via automatic differentiation (AD).

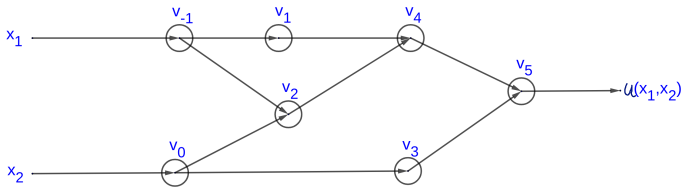
2.4.3 Automatic-Differentiation

Automatic Differentiation (AD) is a set of techniques that allow to compute derivatives efficiently. There exists two types of automatic-differentiation, depending on the ratio between input nodes and output nodes. The *forward mode* is more suitable when the number of input nodes is smaller than the output nodes. On the contrary, the *reverse mode* is more convenient to use for a large number of input variables [GW08; Bay+18].

Forward mode

AD in forward accumulation mode requires only operations on a computational graph, which relates the primal variable x to the final output $u(x)$. The derivative $u'(x)$ is computed by associating each intermediate variable v_i with the derivative $\dot{v}_i = \frac{\partial v_i}{\partial x_j}$, for $j = 1, \ldots, d$. Applying the chain rule to each elementary operation, we generate the corresponding *primal trace*, given on the left hand side in Figure 2-13. Evaluating the primals v_i with their corresponding derivatives $\dot{v}_i$ gives the required derivative in

the final variable $u(x)$. The method can be naturally extended to multivariable functions to compute the Jacobian [Bay+18].



Forward Primal Trace			Forward Tangent (Derivative) Trace		
v_{-1}	$= x_1$	$= 2$	$\dot{v}_{-1}$	$= \dot{x}_1$	$= 1$
v_0	$= x_2$	$= 5$	$\dot{v}_0$	$= \dot{x}_2$	$= 0$
v_1	$= \ln v_{-1}$	$= \ln 2$	$\dot{v}_1$	$= \dot{v}_{-1}/v_{-1}$	$= 1/2$
v_2	$= v_{-1} \times v_0$	$= 2 \times 5$	$\dot{v}_2$	$= \dot{v}_{-1} \times v_0 + \dot{v}_0 \times v_{-1}$	$= 1 \times 5 + 0 \times 2$
v_3	$= \sin v_0$	$= \sin 5$	$\dot{v}_3$	$= \dot{v}_0 \times \cos v_0$	$= 0 \times \cos 5$
v_4	$= v_1 + v_2$	$= 0.693 + 10$	$\dot{v}_4$	$= \dot{v}_1 + \dot{v}_2$	$= 0.5 + 5$
v_5	$= v_4 - v_3$	$= 10.693 + 0.959$	$\dot{v}_5$	$= \dot{v}_4 - \dot{v}_3$	$= 5.5 - 0$
y	$= v_5$	$= 11.652$	$\dot{y}$	$= \dot{v}_5$	$= 5.5$

Figure 2-13: Forward mode with $u(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$ evaluated at $(x_1, x_2) = (2, 5)$. The original forward evaluation of the primal variables on the left is augmented by the derivatives $\dot{v}_i$ with respect to x_1 on the right [Bay+18].

Reverse mode

AD in reverse mode corresponds to a generalized back-propagation algorithm, as it propagates derivatives from a given output. This is achieved by complementing each intermediate variable v_i in the computational graph with the adjoint $\overline{v}_i = \frac{\partial y}{\partial v_i}$.

The derivatives are computed in the second phase of a two-phase process. In the first phase, the forward mode is run once, populating intermediate variables v_i and recording the dependencies in the computational graph. In the second phase, derivatives are calculated by propagating adjoints $\overline{v}_i$ in reverse, from the outputs to the inputs. The computations are illustrated in Figure 2-14

2.4.4 Approximation theorems

In the mathematical theory of artificial neural networks, universal approximation Theorems are results that establish the accuracy of generated class of functions within a given function space of interest. These results are important to quantify the accuracy of the solution of a PDE from a PINN under specific configurations. The approximation capabilities of the feedforward architecture are usually confined to space of continuous functions between two Euclidean spaces.

In 1989, Hornik, Stinchombe, and White published a proof showing that for any continuous function u on a compact set $K \subset \mathbb{R}^d$, there exists a feedforward neural network, having only a single hidden layer, which uniformly approximates u to within an arbitrary $\varepsilon > 0$ on K [HSW89]. Before stating the

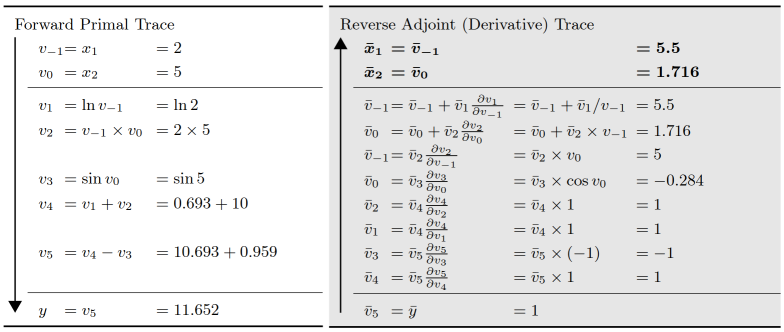


Figure 2-14: Reverse mode with $u(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$ evaluated at $(x_1, x_2) = (2, 5)$. After the forward evaluation of the primal variables, the adjoint operations are evaluated backwards starting from $\overline{v}_5 = \overline{y} = 1$ [Bay+18]. Compared to the Forward mode, here we obtain the derivatives with respect to both x_1 and x_2.

Theorem, we recall that the uniform norm of a function $u : K \longrightarrow \mathbb{R}$ is defined as

$$\|u\|_{\sup} = \sup\{|u(\boldsymbol{x})| : \boldsymbol{x} \in K\}. \tag{2.4.2}$$

In addition, given $I_d := [0, 1]^d$, we define $M(I_d)$ the space of finite, signed regular Borel measures on I_d. A measure μ is *regular* if and only if the following conditions are satisfied:

1. $\mu(K) < \infty$ for all compact sets K.
2. $\mu(E) = \inf\{\mu(U) : E \subseteq U,\ U \text{ open}\}$.
3. $\mu(E) = \sup\{\mu(K) : K \subseteq E,\ K \text{ compact}\}$.

Regular measures exhibit properties that one would expect for compact sets. In fact, the measure of a compact set has finite measure when the measure being applied is regular. This makes sense as compact sets are closed and bounded in a metric space. Similarly, if we are measuring a set E and we take the lower bound on the measure of open sets containing E, we obtain the measure of E. The same holds when we approximate E by compact sets from below. Another reason for restricting the Theorem to regular measures is that this is a prerequisite for invoking the Riesz representation Theorem in the original proof. Finally, we define a *discriminatory* activation function by

Definition 2.4.3 (Discriminatory activation function [ZHS09]) *An activation function $\sigma(\cdot)$ is called discriminatory for $\mu \in M(I_d)$ if*

$$\int_{I_d} \sigma\left(\boldsymbol{w}_i^T \boldsymbol{x} + b_i\right) \mathrm{d}\mu(\boldsymbol{x}) = 0 \tag{2.4.3}$$

for all weights $\boldsymbol{w}_i \in \mathbb{R}^d$, biases $b_i \in \mathbb{R}$ implies that $\mu \equiv 0$.

In some sense, a discriminatory $\sigma(\cdot)$ entails a finite measure when it acts on linear transformations of input. That is, the definition tells us that for nonzero μ, there exist $\boldsymbol{w}_i$, b_i such that equation (2.4.3) does not hold. In other words, the discriminative property of $\sigma(\cdot)$ prevents it from losing the information conveyed from one layer to the next one. The universal approximation Theorem is stated as

Theorem 2.7 (Universal approximation Theorem [HSW89]) *Let u be a continuous function with domain in a compact set $K \subset \mathbb{R}^d$. Let σ be a continuous, discriminatory activation function. Then for any $\varepsilon > 0$ there exists a neural network u_Θ such that*

$$\|u(\boldsymbol{x}) - u_\Theta(\boldsymbol{x})\|_{\sup} < \varepsilon, \tag{2.4.4}$$

where

$$u_\Theta(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i \sigma(\boldsymbol{w}_i \boldsymbol{x} + b_i),\ \alpha_i \in \mathbb{R}.$$

The universal approximation Theorem for arbitrary width and bounded depth has been proved in [Yar17]. We consider the Sobolev space $W^{n,\infty}([0,1]^d)$ as in eq.(2.3.9) and denote the unit ball in $W^{n,\infty}([0,1]^d)$ by

$$F_{n,d} = \{u \in W^{n,\infty}([0,1]^d) : \|u\|_{W^{n,\infty}([0,1]^d)} \leq 1\}.$$

The following Theorem holds for a network architecture with unspecified weights:

Theorem 2.8 (Approximation Theorem[Yar17, Theorem 1]) *For any d, n and $\varepsilon \in (0,1)$, there is a ReLU network architecture that*

1. *is capable of expressing any function from $F_{d,n}$ with error ε.*
2. *has the depth at most $c(ln(1/\varepsilon) + 1)$ and at most $c\varepsilon^{-d/n}(ln(1/\varepsilon) + 1)$ weights and computation units, with some constant $c := c(d, n)$.*

It follows from the Bramble-Hilbert lemma that localised Taylor polynomials can approximate a function $f \in W^{n,\infty}([0,1]^d)$. For functions $f \in C^n([0,1]^d)$, this approximation follows from Taylor's theorem.

An extension of the previous result for approximating function u in $\mathrm{L}^p,\ p \in (0,\infty)$, that does not require logarithmic growth of network layers, has been derived in [PV18].

Given the class of smooth functions

$$\mathcal{F}_{\beta,d,B} := \left\{u \in C^n([-1/2,1/2]) : \|u\|_{C^{0,\beta}([-1/2,1/2])} \leq B\right\},$$

with $\|u\|_{C^{0,\beta}([a,b])} = \sup_{x,y\in[a,b],x\neq y}\left\{\frac{|u(x)-u(y)|}{|x-y|^\beta} < \infty\right\}$, we can state the Theorem:

Theorem 2.9 (Generalised approximation Theorem [PV18, Theorem 3.1]) *For any $d \in \mathbb{N}, \beta, B, p > 0$ there exists constants $s = s(d,\beta,B,p)$ and $c = c(d,\beta,B) > 0$ so that for any function $f \in \mathcal{F}_{\beta,d,B}$ and any $\varepsilon \in (0,1/2)$, there is a ReLU neural network u_Θ with at most $(2+\lceil\log_2\beta\rceil)(11+\beta/d)$ layers, and at most $c\varepsilon^{-d/\beta}$ nonzero weights satisfying*

$$\|u_\Theta - u\|_{\mathrm{L}^p([-1/2,1/2]^d)} < \varepsilon, \quad \|u_\Theta\|_{\sup} < \lceil B\rceil.$$

Theorems 2.7–2.9 refer only to the error of the approximation function with respect to the network parameters, but do not provide any convergence rate with respect to the number of training points or other scaling quantities. This is a key difference from the FEM, as we have showed that an order of convergence can be defined based on the mesh size in §5.3.3. We will evidence that the neural network is unable to scale the accuracy of the approximate solution as the FEM does in Chapter 5.

The Authors in [OPS20] address ReLU DNNs approximation rates for analytic functions in $I = (0,1)$ with possibly a point singularity. Based on their results, DNNs can emulate high-order h-FEM on general partitions of a bounded interval, as well as p- and hp-FEM. From an approximation theoretical point of view, RELU DNNs perform as well as the best FE approximation for a number of function classes, which are solutions of elliptic PDEs. Therefore, the huge interest arisen by DL methodologies is not really justified and should be contextualized in terms of other well established mathematical frameworks.

Chapter 5 will treat the training of a neural network to solve Poisson's equation on the L-shaped domain, as done in Chapter 4. We will show that the location of the training points impacts dramatically on the accuracy of the approximate solution. If they are chosen as vertices of the OT mesh derived in Chapter 4, the result will be much more accurate that a random choice of quadrature points. We will then compare the convergence rate obtained by the network output with that one retrieved by the SIP-dG method defined in Chapter 4. We will conclude the Chapter by repeating the same experiment on a simpler Poisson problem posed over a squared domain.

Chapter 3

An adaptive conservative moving mesh method

Abstract

We present an adaptive moving mesh strategy endowed with a mass conservation property for the numerical solution of the two-dimensional linear advection equation in which the solution exhibits localised near singular behaviour. We achieve this through the solution of an auxiliary moving mesh problem driven by a gradient based monitor function. The method we propose is of staggered type, hence requires an appropriate data transfer operator over different meshes. We make use of an L^2 projection operator that requires the construction of a *supermesh* for evaluation. We show by extensive numerical experiments that the method is robust and is able to approximate challenging numerical features.

3.1 Introduction

Partial differential equations (PDEs) arise from a variety of areas [MG90], including meteorological, such as the semi-geostrophic equations [RN94], oceanographical, such as the Korteweg-de Vries (KdV) and optical, such as the nonlinear Schrödinger equations [MGO05]. These equations are examples of *conservation laws*, which often develop very localised transient features, such as shocks and moving fronts. These features are difficult to approximate using standard numerical schemes posed over uniform meshes. One potential solution to this, well used in finite element techniques, is to employ different refinement strategies to increase the spatial accuracy in the region of the propagating shock. These include

local mesh refinement [PLW05], local polynomial enrichment [AS98] and moving the underlying mesh nodes [Eis87], *h*-, *p*- and *r*-refinement, respectively. One of the common features of these refinement strategies is that they all alter the number or the location of the degrees of freedom in the mesh.

Moving mesh methods are, by now, well studied in the numerical solution of PDEs [BHR09; BHR96]. As discussed in the introduction of Chapter 2, the main advantage of these methods is that they do not alter the topology of the initial mesh, preserving the number of nodes, connectivity and data structures. This allows for an efficient implementation and coupling to existing computational fluid dynamics (CFD) solvers [Her+13]. The main ingredients, in addition to the underlying PDE discretisation, are the mesh equations and the monitor function, which controls the relative density of the mesh points in the physical domain and is commonly based on the gradient or curvature of the field [BHR09; HR11].

Fundamentally, moving mesh strategies can be split into two categories [Tan05]:

- Interpolation free (*quasi-Lagrange*) methods - The mesh equation and PDE are approximated simultaneously. This is computationally more complex as the position of the nodes of the mesh are additional degrees of freedom in the system. There is, however, no requirement to transfer data from an old mesh to the new one.
- Interpolation based (*rezoning*) methods - The mesh equation and PDE are staggered, the PDE is solved first and mesh equation subsequently based on that solution. Computationally, this is cheaper than interpolation free methods. However a *data transfer operator* is required to interpret the discrete solution over the new mesh.

In this Chapter, we are concerned with interpolation based methods and pay particular attention to properties of the *data transfer operator*, a necessity allowing the evolution of the discrete solution as the underlying mesh changes. Its definition does introduce various challenges as it is a common source of diffusion. The long term dynamics of solutions can be destroyed by the addition of *artificial numerical diffusion*. The reason for inclusion of the proposed transfer operator is the desirable stability properties this endows on the discrete scheme.

Our focus in this Chapter is illustrating how the data transfer step can be done in a conservative fashion through a *local Galerkin projection*, removing one of the inherent disadvantages of interpolation based methods over those that are interpolation free [FM10]. More specifically, the conservative property is the total mass of the scalar field on which this procedure is applied. This requires the construction of a *supermesh*, a conforming triangulation obtained by the intersection between the old and the new mesh.

The novelty of this approach consists of coupling the conservative Galerkin projection with *Winslow's variable diffusion method* for mesh movement, described in §2.2.1 of Chapter 2. To showcase the properties of our approach, we employ the adaptive strategy with a well used upwinding *discontinuous Galerkin* (dG) discretisation of the linear advection equation. We use this equation, with initial conditions from smooth to singular, as our test problem as it is arguably the most challenging conservation law to

approximate despite its linearity. The reason for this is that any numerical errors are propagated in time, there is no chance to rectify this through entropic loss as is common in other (systems of) conservation laws that develop shocks, shallow water equations or Euler's equations, for example.

We conduct numerical experiments to assess the robustness and approximability of the r-adaptive procedure and the adaptive-conservative dG scheme. In this case, we are able to show that:

1. The moving mesh method is able to generate meshes locally refined around areas of interest. Families of these meshes are able to optimally approximate even discontinuous functions.

2. The conservative r-adaptive approximation significantly outperforms the non-conservative variant as well as the equivalent simulation over a uniform grid.

3. The moving mesh problem is sensitive to parametric choices, especially for irregular functions.

4. With good choices of these parameters, the r-adaptive approximation of the advection equation outperforms the uniform counterpart.

The remainder of the Chapter is structured as follows: In §3.2, we introduce the problem, relevant notation and the discretisation scheme. §3.3 contains an overview of moving mesh methods used for solving time-dependent PDEs as well as our proposed moving mesh scheme using Winslow's method. In §3.4, we describe the local conservative Galerkin projection and the algorithm that is used to construct and process the supermesh. §3.5 illustrate numerical experiments designed to test the discretisation of the moving mesh PDE. Finally, we summarise our results in §3.6 and provide an overview of the topic addressed in Chapter 4.

3.2 Problem setup and discretisation

Throughout this Chapter we denote the standard Lebesgue spaces by $\mathrm{L}^p(\Omega)$ for $\Omega \subseteq \mathbb{R}$, $p \in [1, \infty]$, equipped with corresponding norms $\|u\|_{\mathrm{L}^p(\Omega)}$. In equation (2.3.7) of Chapter 2 the Hilbert space of order $k \in \mathbb{N}$ was defined by:

$$\mathrm{H}^k(\Omega) := \{u \in \mathrm{L}^2(\Omega) : \|u\|_{\mathrm{H}^k(\Omega)} < \infty\}, \tag{3.2.1}$$

where $\|u\|^2_{\mathrm{H}^k(\Omega)} := \sum_{\alpha:|\alpha|\leq k} \|D^\alpha u\|^2_{\mathrm{L}^2(\Omega)}$ and D^α are understood in weak sense.

Furthermore, for a portion of the boundary $\Gamma_D \subset \partial\Omega$, we define

$$\mathrm{H}^1_{0,\Gamma_D}(\Omega) := \left\{u \in \mathrm{H}^1(\Omega) : \ u|_{\Gamma_D} = 0\right\}. \tag{3.2.2}$$

Let $\Omega \subset \mathbb{R}^2$ be a convex, simply connected domain of interest with outward pointing normal $\boldsymbol{n}$ and consider the problem

$$\begin{aligned} \frac{\partial u}{\partial t} + \boldsymbol{v} \cdot \nabla u = 0 \text{ in } \Omega \times [0,T], \\ u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}) \text{ on } \Omega \times \{0\}, \\ u = g \text{ on } \partial\Omega_- \times [0,T], \end{aligned} \tag{3.2.3}$$

where

$$\partial\Omega_- = \{x \in \partial\Omega : \ \boldsymbol{v} \cdot \boldsymbol{n} < 0\}, \tag{3.2.4}$$

the constant prescribed velocity $\boldsymbol{v} \in L^\infty$, $u_0 \in \mathrm{H}^0(\Omega)$, and the inflow boundary condition $g \in \mathrm{L}^2(\partial\Omega_-)$ for $t \in [0,T]$. We will be particularly interested in problems for which $u_0(\boldsymbol{x})$ has a (near) singular behaviour.

Given those conditions, we can define a map $u : t \to u(\boldsymbol{x}, t)$ from the temporal domain to the Hilbert space $\mathrm{H}^1(\Omega)$. We make use of the following notation for time-dependent Bochner spaces:

$$\mathrm{L}^p(0,T;\mathrm{H}^1(\Omega)) := \left\{ u : [0,T] \to \mathrm{H}^1(\Omega) : \ \int_0^T \|u(t)\|^p_{\mathrm{H}^1(\Omega)} \, \mathrm{d}t < \infty \right\}. \tag{3.2.5}$$

The solution of the advection equation $u(\boldsymbol{x}, t)$, $(\boldsymbol{x}, t) \in \Omega \times [0,T]$ lies in this space [Eva10, Section 5.9.2].

Let $\mathscr{T}$ be a conforming mesh of Ω comprised of simplicial and/or box-type elements as definition 2.3.4 of Chapter 2.

We consider the *finite element space*

$$\mathbb{V} := \{\Phi \in \mathrm{L}^2(\Omega) : \Phi|_K \in \mathbb{P}^p(K)\}, \tag{3.2.6}$$

where $\mathbb{P}^p(K)$ is the space of polynomials of total degree p for $p \geq 0$.

For $w \in \mathbb{V}$ and any $K \in \mathscr{T}$, we define the *upwind jump* across the inflow boundary ∂K_- by

$$\lfloor w \rfloor (\boldsymbol{x}) := \lim_{\epsilon \to 0^+} (w(\boldsymbol{x} + \epsilon \boldsymbol{v}) - w(\boldsymbol{x} - \epsilon \boldsymbol{v})). \tag{3.2.7}$$

Let us denote $\mathscr{G} := \{w \in \mathrm{L}^2(\Omega) : \boldsymbol{v} \cdot \nabla w \in \mathrm{L}^2(\Omega)\}$ as the graph space of the PDE. For $w \in \mathscr{G}$, we let w_K^+ and w_K^- denote the interior and exterior trace of w on K, respectively. The exterior trace on K can

be referred to as the interior trace on K', where K' shares an edge contained in $\partial K_- / \partial\Omega_-$. Then

$$\lfloor w \rfloor |_K := w|_K^+ - w|_K^-. \tag{3.2.8}$$

Let

$$\begin{aligned} \mathscr{A}(u, \Phi) &:= \sum_{K \in \mathscr{T}} \left[\int_K \frac{\partial u}{\partial t} \Phi + \boldsymbol{v} \cdot \nabla u \Phi \, \mathrm{d}x - \int_{\partial K_- / \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} \lfloor u \rfloor \Phi^+ \, \mathrm{d}s - \int_{\partial K_- \cap \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} u^+ \Phi^+ \, \mathrm{d}s \right], \\ l(\Phi) &:= \sum_{K \in \mathscr{T}} \left[\int_{\partial K_- \cap \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} g \Phi^+ \, \mathrm{d}s \right]. \end{aligned} \tag{3.2.9}$$

The semidiscrete dG method we consider to approximate (3.2.3) is to seek $U \in \mathrm{C}^1([0,T]; \mathbb{V})$ such that

$$\mathscr{A}_h(U, \Phi) = l_h(\Phi) \quad \forall \, \Phi \in \mathbb{V}. \tag{3.2.10}$$

Fully discrete scheme. Subdivide the time domain $[0,T]$ into a partition of N_T consecutive subintervals with endpoints denoted $0 = t_0 < t_1 < \cdots < t_{N_T} = T$. We denote the n-th timestep as $\tau_n = t_n - t_{n-1}$ and consistently use the shorthand $F^n(\cdot) = F(\cdot, t_n)$ for a time-dependent function or function space.

We discretise (3.2.10) using a three-stage explicit Strong-Stability-Preserving Runge-Kutta (SSPRK) scheme [GGI]. This high order time discretisation method preserves the strong stability properties of first order explicit Euler time stepping. Let $\boldsymbol{\Phi}^n \in \mathbb{R}^M$ denote the vector of basis functions such that $\mathbb{V}^n = \mathrm{span}(\boldsymbol{\Phi}^n)$. The SSPRK method can be realised by defining mass and advection matrices $\boldsymbol{M}^n$ and $\boldsymbol{A}^n$ respectively componentwise as

$$\begin{aligned} \boldsymbol{M}^n_{i,j} &= \int_\Omega \boldsymbol{\Phi}^n_j \boldsymbol{\Phi}^n_i \, \mathrm{d}\boldsymbol{x}, \\ \boldsymbol{A}^n_{i,j} &= -\sum_{K \in \mathscr{T}} \left[\int_K \left(\boldsymbol{v} \cdot \nabla \boldsymbol{\Phi}^n_j \right) \boldsymbol{\Phi}^n_i \, \mathrm{d}x - \int_{\partial K_- / \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} \left\lfloor \boldsymbol{\Phi}^n_j \right\rfloor \boldsymbol{\Phi}^{n,+}_i \, \mathrm{d}s - \int_{\partial K_- \cap \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} \boldsymbol{\Phi}^{n,+}_j \boldsymbol{\Phi}^{n,+}_i \, \mathrm{d}s \right], \\ \boldsymbol{b}^n_i &= \sum_{K \in \mathscr{T}} \int_{\partial K_- \cap \partial \Omega_-} \boldsymbol{v} \cdot \boldsymbol{n} \boldsymbol{\Phi}^{n,+}_i g \, \mathrm{d}s. \end{aligned} \tag{3.2.11}$$

Note that each of the matrices are time-dependent. This is since $\mathbb{V}^n$ changes in time through movement of degrees of freedom.

The SSPRK3 solution $U^{n+1} \in \mathbb{V}^n$ is computed by the following multi-stage iterations for $n = 0, \ldots, N-1$:

$$\begin{aligned} \boldsymbol{M}^n W^1 &= U^n + \tau_n(\boldsymbol{A}^n U^n - \boldsymbol{b}^n)\,, \\ \boldsymbol{M}^n W^2 &= \frac{3}{4}U^n + \frac{1}{4}\Big(W^1 + \tau_n\Big(\boldsymbol{A}^n W^1 - \boldsymbol{b}^n\Big)\Big)\,, \\ \boldsymbol{M}^n U^{n+1} &= \frac{1}{3}U^n + \frac{2}{3}\Big(W^2 + \tau_n\Big(\boldsymbol{A}^n W^2 - \boldsymbol{b}^n\Big)\Big)\,, \end{aligned} \tag{3.2.12}$$

where $W^{1,2} \in \mathbb{R}^M$ and, by an abuse of notation, we denote U^i as the array of finite element coefficients at time t_i. The $\boldsymbol{M}^n$ is a block diagonal matrix, thus can be inverted elementwise leading to very little computational overhead, as expected from an explicit timestepping method. Note that the finite element space $\mathbb{V}^n$ does not change from one RK stage to another. See Algorithm 3 for further details.

3.3 Moving mesh methods

Moving mesh strategies are characterised by three major components: the mesh adaptation strategy, the method used to discretise the physical PDE, and the approach used to couple the moving mesh to the evolving solution of the PDE.

The effect of the mesh movement in the time discretisation of the physical PDE can be treated with either the *quasi-Lagrange* approach or the *rezoning* approach [BHR09; HR11]. Quasi-Lagrange methods ensure the mesh points move continuously in time and physical time derivatives are transformed into time derivatives along mesh trajectories with an additional convection term [HR11]. If an explicit time integrator is used for solving the physical PDE, this extra convection term can pose a critical constraint on the choice of the timestep size due to the Courant-Friedrichs-Lewy (CFL) condition. Indeed such methods are often unstable and very stiff [LP96].

Rezoning approaches require that the mesh is updated at each time step by solving a Moving mesh PDE (MMPDE), introduced in §2.2 of Chapter 2. This procedure treats the MMPDE and the physical PDE separately, offering flexibility for coding each component with any preferred numerical scheme. The physical solution must then be transferred from the old to the new mesh. This step is critical for the success of this approach, as a poor choice can lead to loss of both accuracy and stability. A conservative interpolation procedure [FM10; Zha06] is desirable to preserve those properties and ensure that conservation laws are inherited by the numerical scheme.

Given $U^n \in \mathbb{V}^n$, the solution of (3.2.12) at the time t_n, Algorithm 3 describes the *adaptive-conservative dG scheme* that will be studied in the numerical experiments of §3.5. Note that it inherently calls another algorithm defined by Algorithm 4.

This algorithm is similar to the rezoning approach originally formulated in [HR11, Section 2.6], with the only addition of the local Galerkin projection procedure.

Algorithm 3 Adaptive-conservative dG scheme.

Require: $u_0, \mathscr{T}^0, T$
Ensure: U^{N_T}
1: Set $n := 0, t_0 := 0$
2: Let $U^0 = \Pi u_0$ be the L^2 projection of the initial data
3: Adapt the mesh $\mathscr{T}^0$ by solving the MMPDE (Algorithm 4)
4: **while** $t_n \leq T$ **do**:
5: Step forward the SSPRK-dG scheme (3.2.12) to compute $U^{n+1} \in \mathbb{V}^n$
6: Adapt the mesh $\mathscr{T}^{n+1}$ by solving the MMPDE (Algorithm 4)
7: Construct an appropriate projector $\Pi : \mathbb{V}^n \to \mathbb{V}^{n+1}$ and let $\widehat{U}^{n+1} = \Pi U^n$
8: Update the timestep τ_{n+1} to ensure the CFL condition is satisfied
9: Set $t_n = t_n + \tau_{n+1}$, $U^n := \widehat{U}^{n+1}$, n += 1
10: **end while**

3.3.1 Winslow's variational-based diffusion method

In §2.2.1 of Chapter 2 we introduced Winslow's adaptive diffusion method. We now discretise the equation (2.2.15) using the finite element method and apply it to the linear advection problem.

1 Numerical Solution of Winslow's MMPDE In this section, we illustrate the numerical procedure to solve the MMPDE (2.2.15) by using a finite element discretisation. In particular, we discretise the weak formulation (2.2.15) of the MMPDE in space and time using finite element space comprised of linear elements and a semi-implicit Euler integrator. The computational domain is still referred by the coordinates $\boldsymbol{\xi} = (\xi, \eta)$, while the continuous coordinates of Ω are expressed by $\boldsymbol{x} = (x, y)$.

2 Weak formulation We begin by splitting the boundary of Ω_c into two components, Γ_D and Γ_N. The weak formulation of Winslow' MMPDE is obtained by multiplying both sides of (2.2.15) by an appropriate test function and integrate by parts over Ω_c. We seek $\boldsymbol{x} \in \mathrm{H}^1_{0,\Gamma_D}(\Omega_c)^2$

$$\begin{aligned}\int_{\Omega_c} A(\boldsymbol{x})\frac{\partial \boldsymbol{x}}{\partial s}\Phi \,\mathrm{d}\boldsymbol{\xi} = \int_{\Omega_c} \rho(\boldsymbol{x})\Bigg\{ -\frac{\partial}{\partial \xi}(\alpha_1(\boldsymbol{x})\Phi)\frac{\partial \boldsymbol{x}}{\partial \xi} + \Bigg[\frac{\partial}{\partial \eta}(\alpha_2(\boldsymbol{x})\Phi)\frac{\partial \boldsymbol{x}}{\partial \xi} + \\ \frac{\partial}{\partial \xi}(\alpha_2(\boldsymbol{x})\Phi)\frac{\partial \boldsymbol{x}}{\partial \eta}\Bigg] - \frac{\partial}{\partial \eta}(\alpha_3(\boldsymbol{x})\Phi)\frac{\partial \boldsymbol{x}}{\partial \eta}\Bigg\}\mathrm{d}\boldsymbol{\xi} \quad \forall\, \Phi \in \mathrm{H}^1_{0,\Gamma_D}(\Omega_c)^2,\end{aligned} \tag{3.3.1}$$

where $\alpha_1(\boldsymbol{x}) = x_\eta^2 + y_\eta^2$, $\alpha_2(\boldsymbol{x}) = x_\xi x_\eta + y_\xi y_\eta$, $\alpha_3(\boldsymbol{x}) = x_\xi^2 + y_\xi^2$ and $A(\boldsymbol{x}) = |J|^2\rho^2(\boldsymbol{x})$. We denote by s the temporal variable for the MMPDE, not to be confused with t used in equation (3.2.3).

Note that the weak form (3.3.1) requires the solution $\boldsymbol{x}(\boldsymbol{\xi}, t)$ to be twice weakly differentiable. This condition is delicate to interpret in a primal form, so we circumvent the difficulties by computing $\alpha_{1,2,3}$

in the following fashion:

$$\begin{aligned}\widetilde{\alpha}_1(\boldsymbol{x}) &= \Pi\Big(x_\eta^2 + y_\eta^2\Big)\,,\\ \widetilde{\alpha}_2(\boldsymbol{x}) &= \Pi\big(x_\xi x_\eta + y_\xi y_\eta\big)\,,\\ \widetilde{\alpha}_3(\boldsymbol{x}) &= \Pi\Big(x_\xi^2 + y_\xi^2\Big)\,.\end{aligned} \tag{3.3.2}$$

Time discretisation. The discretisation of (3.3.1) in the temporal variable is realised with a *semi-implicit Euler scheme*. To that end, we divide the domain $[0, S]$ into a partition of N_S consecutive subintervals $0 = s_0 < s_1 < \cdots < s_{N_S} = S$, with $s_i - s_{i-1} = k$ for all i. We treat the non-linear terms through evaluation at the current time step s_i, while the linear terms are evaluated at the next time step s_{i+1}. We denote $\boldsymbol{x}_{i+1}$ to be the solution of (3.3.1) at s_{i+1}. Specifically, we seek $\boldsymbol{x}_{i+1} \in \mathrm{H}^1_{0,\Gamma_D}(\Omega_c)^2$ such that

$$\begin{aligned}\int_{\Omega_c} A(\boldsymbol{x}_i)\left(\frac{\boldsymbol{x}_{i+1}-\boldsymbol{x}_i}{k}\right)\Phi\,\mathrm{d}\boldsymbol{\xi} = \int_{\Omega_c}\rho(\boldsymbol{x}_i)\Bigg\{ &-\frac{\partial}{\partial\xi}(\widetilde{\alpha}_1(\boldsymbol{x}_i)\Phi)\frac{\partial \boldsymbol{x}_{i+1}}{\partial\xi}\\ &+\left[\frac{\partial}{\partial\eta}(\widetilde{\alpha}_2(\boldsymbol{x}_i)\Phi)\frac{\partial\boldsymbol{x}_{i+1}}{\partial\xi}+\frac{\partial}{\partial\xi}(\widetilde{\alpha}_2(\boldsymbol{x}_i)\Phi)\frac{\partial\boldsymbol{x}_{i+1}}{\partial\eta}\right]-\frac{\partial}{\partial\eta}(\widetilde{\alpha}_3(\boldsymbol{x}_i)\Phi)\frac{\partial\boldsymbol{x}_{i+1}}{\partial\eta}\Bigg\}\,\mathrm{d}\boldsymbol{\xi}\quad \forall\,\Phi\in\mathrm{H}^1_{0,\Gamma_D}(\Omega_c)^2.\end{aligned} \tag{3.3.3}$$

Fully discrete scheme. Let Ω_c be tessellated with a conforming, shape regular mesh $\mathscr{T}_c$ that is fixed in time. We denote by $\boldsymbol{\Phi} \in \mathbb{R}^P$ the vector of basis functions such that $\mathbb{P}^1(\mathscr{T}_c) = \mathrm{span}(\boldsymbol{\Phi})$. The algebraic system for the numerical solution $\boldsymbol{X}_i = (X_i, Y_i)$ of (3.3.3) can be decoupled and takes the form

$$(\boldsymbol{M}_i - k\boldsymbol{S}_i)\boldsymbol{X}_{i+1} = \boldsymbol{M}_i\boldsymbol{X}_i, \tag{3.3.4}$$

where $\boldsymbol{X}_i$ denotes the array of coefficients at time s_i and the weighted mass and stiffness matrices $\boldsymbol{M}_i, \boldsymbol{S}_i$ are given by

$$(\boldsymbol{M}_i)_{a,b} = \int_{\Omega_c} A(\boldsymbol{X}_i)\boldsymbol{\Phi}_a(\boldsymbol{\xi})\boldsymbol{\Phi}_b(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_{\Omega_c}|J(\boldsymbol{\xi})|^2 W(\boldsymbol{X}_i)^2\boldsymbol{\Phi}_a(\boldsymbol{\xi})\boldsymbol{\Phi}_b(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}, \tag{3.3.5}$$

$$\begin{aligned}(\boldsymbol{S}_i)_{a,b} = \int_{\Omega_c} W(\boldsymbol{X}_i)\Bigg(&-\frac{\partial}{\partial\xi}(\widetilde{\alpha}_1(\boldsymbol{X}_i)\boldsymbol{\Phi}_a)\frac{\partial\boldsymbol{\Phi}_b}{\partial\xi}+\frac{\partial}{\partial\eta}(\widetilde{\alpha}_2(\boldsymbol{X}_i)\boldsymbol{\Phi}_a)\frac{\partial\boldsymbol{\Phi}_b}{\partial\xi}+\\ &\frac{\partial}{\partial\xi}(\widetilde{\alpha}_2(\boldsymbol{X}_i)\boldsymbol{\Phi}_a)\frac{\partial\boldsymbol{\Phi}_b}{\partial\eta}-\frac{\partial}{\partial\eta}(\widetilde{\alpha}_3(\boldsymbol{X}_i)\boldsymbol{\Phi}_a)\frac{\partial\boldsymbol{\Phi}_b}{\partial\eta}\Bigg)\mathrm{d}\boldsymbol{\xi}.\end{aligned} \tag{3.3.6}$$

Practically, the matrices (3.3.5) and (3.3.6) are evaluated by a three point Gaussian quadrature rule and the non-symmetric system (3.3.4) is solved by using sparse LU decomposition [BH74]. For large problems resulting from, for example a high-resolution mesh, iterative methods can be used as faster

and less memory-demanding options, e.g. GMRES and MINRES.

3 Boundary conditions of the MMPDE For the sake of clarity, the numerical solution of (3.3.1) computes the shift of the new mesh coordinates rather than the coordinates themselves. Homogeneous Dirichlet BCs fix the mesh points over Γ_D. However, this condition generally does not ensure good mesh quality, especially close to the boundary, and a combination of Dirichlet and Neumann BCs result in less skewed meshes [BHR09]. For $\Omega = [0,1]^2$ we enforce essential Dirichlet conditions on X_i on the east-west sides and natural Neumann conditions on the top-bottom sides. Concerning Y_i, Dirichlet conditions are applied on the top-bottom sides and Neumann conditions on the east-west sides. Importantly, this allows the mesh nodes to move along the boundaries.

4 The monitor function To assemble the system in (3.3.4), the discrete monitor function $W(\boldsymbol{X}_i) \in \mathbb{P}^1(\mathscr{T}_i)$ must be defined over Ω_c. We choose to represent $\widehat{W}_i \in \mathbb{P}^1(\mathscr{T}_c)$. The discrete map $\boldsymbol{X}_i(\boldsymbol{\xi})$ allows us to compute the coefficients $\widehat{W}_i$ as

$$\widehat{W}_i = \widehat{W}_i(\boldsymbol{\xi}) = W(\boldsymbol{X}_i(\boldsymbol{\xi})) = \sqrt{1 + \frac{1}{\delta}\,\|G[U_i](\boldsymbol{X}_i)\|^2_{\mathrm{L}^2}}, \tag{3.3.7}$$

where the optimal intensity parameter, δ [HR11], is chosen as

$$\delta = \frac{1}{|\Omega|}\int_\Omega \|G[U_i]\|_{\mathrm{L}^2}\,\mathrm{d}\boldsymbol{x}. \tag{3.3.8}$$

The term $G[V] \in \mathbb{V}\times\mathbb{V}$ is a *discrete gradient operator* of a generic finite element object $V \in \mathbb{V}$, defined through

$$\int_\Omega G[V]\Phi\,\mathrm{d}\boldsymbol{x} = \int_\Omega \nabla V\,\Phi\,\mathrm{d}\boldsymbol{x} - \int_{\mathcal{E}} [\![V]\!]\,\{\!\{\,\Phi\,\}\!\}\,\mathrm{d}s \quad \forall\,\Phi \in \mathbb{V}. \tag{3.3.9}$$

The monitor function can be directly defined with respect to the vector U_i and denoted by $\widehat{W}(U_i)$.

A common practice that we utilise in our numerical experiments to reduce the skewness in a mesh is to smooth $\widehat{W}$ by solving a diffusion PDE [HK15b]. This procedure can be realised by considering the equation to find $\widetilde{W} \in \mathbb{P}^1(\mathscr{T}_c)$ such that

$$\int_{\Omega_c} \Phi\widetilde{W}\,\mathrm{d}\boldsymbol{\xi} + \int_{\Omega_c} \beta\nabla\Phi\cdot\nabla\widetilde{W}\,\mathrm{d}\boldsymbol{\xi} = \int_{\Omega_c} \Phi\widehat{W}\,\mathrm{d}\boldsymbol{\xi} \quad \forall\,\Phi \in \mathbb{P}^1(\mathscr{T}_c), \tag{3.3.10}$$

where $\beta \ll 1$ is a diffusion parameter and homogeneous Neumann boundary conditions are applied on $\partial\Omega_c$.

The resulting MMPDE scheme provides an adapted mesh $\mathscr{T}_{N_S}$ to the solution U^{n+1} of (3.2.12). This represents a subfunction called by Algorithm 3.

The errors in the solution of (3.3.4) are critically dependent on the nature of the resulting mesh. Essentially, the error is a combination of the *skewness*, or lack of *shape regularity*, and the *scale* of the

Algorithm 4 MMPDE

Require: N_S, TOL, k, β, b, U^{n+1}
Ensure: $\mathcal{T}_{N_S}$
1: Set $i := 0$ and $\varepsilon := 2\text{TOL}$
2: **while** $i < N_S$ or $\epsilon <$TOL **do**:
3: Solve the fully discretised MMPDE (3.3.4) to obtain $\boldsymbol{x}_{i+1}$ and generate the mesh $\mathcal{T}_{i+1}$
4: Compute $\widehat{W}(U^{n+1})$ using (3.3.7)
5: Smooth $\widehat{W}(U^{n+1})$ to obtain $\widetilde{W}(U^{n+1})$ by (3.3.10)
6: Set $\varepsilon = \|\boldsymbol{x}_{i+1} - \boldsymbol{x}_i\|_{\mathrm{L}^2(\Omega_c)} / \|\boldsymbol{x}_i\|_{\mathrm{L}^2(\Omega_c)}$
7: Set $\boldsymbol{x}_i := \boldsymbol{x}_{i+1}$ and i += 1
8: **end while**

mesh. The skewness indicates how far mesh elements are from being equilateral, while the scale checks how well the mesh is adapted to the solution. For triangular elements, the skewness can be computed as the ratio between the minimum and maximum vertex angle. The quality measure for the element K and the *global quality measure* Q were defined in Chapter 2 as (2.3.7) and (2.3.31), respectively.

We remark that a high value of Q corresponds to skewed mesh elements that can accurately align to the small-scale features of the solution. In the context of hyperbolic PDEs, this can pose issues for the efficiency and numerical stability of the algorithm 3. The CFL condition restricts the time step for explicit time discretisation schemes with skewed elements and increases the runtime of the simulation. As a result, the quality measure for such family of PDEs can only be used as useful metric for mesh adaptation to the initial condition u_0.

5 Sensitivity of the MMPDE on the user-specified parameters We highlight that the stability of the algorithm 4 and the quality of the resulting mesh are strongly dependent on the smoothing parameters β and the number of iterations N_S. A high value of β increases the stability of the algorithm at the cost of the level of adaptivity. The monitor function is more diffused over the domain Ω and the mesh points are unlikely to collide. However, this implies that the mesh elements might not accurately align to small-scale features.

Concerning the tuning of N_S, it is recommended to first run the algorithm for the initial condition u_0 with a high number of iterations, and keep track of the relative tolerance ε. In terms of computational time, the optimal value of N_S can be set to the iteration number that guarantees a level of tolerance close to ε. Although this approach is reasonable for solutions that preserve the structure of small-scale features, there is no general guidance about initially smooth solutions of hyperbolic PDEs that develop such features over time. In this case, the safest approach that ensures a good mesh adaptation is to fix a high value of N_s and rely more on the tolerance ε as criterion to stop the algorithm.

3.4 Data transfer over timesteps

Data transfer of the field U^{n+1} from the mesh $\mathscr{T}^n$ to the new adapted mesh $\mathscr{T}^{n+1} := \mathscr{T}_{N_S}$ is an important step for the accuracy and stability of the solution of a physical PDE. A Lagrange interpolator in this role suffers from several drawbacks. In particular, the integral of the interpolated function is not conserved, monotonicity is preserved only for linear interpolation, and is not suitable for discontinuous fields.

The *local Galerkin projection* does not suffer from the same drawbacks and we will use this to represent the scalar field onto the new mesh [FM10]. This process involves the construction of a conforming triangulation that captures the features of $\mathscr{T}^{n+1}$ and $\mathscr{T}^n$. In the literature this is referred to as a *supermesh*.

3.4.1 Local Galerkin Projection

Consider the projection of the scalar field $U^{n+1} \in \mathbb{V}^n$ to $\mathbb{V}^{n+1}$. Let $\mathbb{V}^n = \text{span}(\mathbf{\Phi}^n)$ and $\mathbb{V}^{n+1} = \text{span}(\mathbf{\Phi}^{n+1})$. We make use of an L^2 projection; to that end we seek $\widehat{U} \in \mathbb{V}^{n+1}$ such that

$$\int_\Omega \widehat{U}\mathbf{\Phi}_i^{n+1}\,\mathrm{d}\boldsymbol{x} = \int_\Omega U^{n+1}\mathbf{\Phi}_i^{n+1}\,\mathrm{d}\boldsymbol{x} \quad \forall\, i \in \{1,\dots,M\}, \tag{3.4.1}$$

Consider the projection of the scalar field U^{n+1} from the old mesh $\mathscr{T}^n$ to the new mesh $\mathscr{T}^{n+1}$. By definition, let $\widehat{U}^{n+1}$ be the optimal function in the L^2 norm:

$$\left\|U^{n+1} - \widehat{U}^{n+1}\right\|_{\mathrm{L}^2(\Omega)} = \min_{U\in\mathbb{V}^{n+1}} \left\|U^{n+1} - U\right\|_{\mathrm{L}^2(\Omega)} \tag{3.4.2}$$

The L^2 norm in (3.4.2) is minimised if the derivative of $\left\|U^{n+1} - U\right\|_{\mathrm{L}^2}$ with respect to every coefficient of U is zero. This leads to the system

$$\int_\Omega U\mathbf{\Phi}_i^{n+1}\,\mathrm{d}\boldsymbol{x} = \int_\Omega U^{n+1}\mathbf{\Phi}_i^{n+1}\,\mathrm{d}\boldsymbol{x} \quad \forall\, i \in \{1,\dots,M\}. \tag{3.4.3}$$

From eq.(3.4.3) global mass conservation follows naturally as the constant function $\mathbb{1}_\Omega$ is contained in $\mathbb{V}^n$ for each $n \in \mathbb{N}$.

Expanding U^{n+1} and $\widehat{U}^{n+1}$ with respect to their basis functions yields

$$\int_\Omega \sum_{i=1}^{M} \widehat{U}_j^{n+1}\mathbf{\Phi}_i^{n+1}\mathbf{\Phi}_j^{n+1}\,\mathrm{d}\boldsymbol{x} = \int_\Omega \sum_{i=1}^{M} U_i^{n+1}\mathbf{\Phi}_i^{n}\mathbf{\Phi}_j^{n+1}\,\mathrm{d}\boldsymbol{x} \quad \forall\, j \in \{1,\dots,M\}. \tag{3.4.4}$$

If we abuse notation slightly and let $U^{n+1}, \widehat{U}$ denote the array of the finite element coefficients and

$$(\boldsymbol{M}_1)_{i,j} = \int_\Omega \boldsymbol{\Phi}_i^{n+1} \boldsymbol{\Phi}_j^{n+1} \,\mathrm{d}\boldsymbol{x}, \tag{3.4.5}$$

$$(\boldsymbol{M}_2)_{i,j} = \int_\Omega \boldsymbol{\Phi}_i^{n+1} \boldsymbol{\Phi}_j^{n} \,\mathrm{d}\boldsymbol{x} \quad \forall\, i,j \in \{1,\dots,M\}. \tag{3.4.6}$$

then $\widehat{U}$ solves

$$\boldsymbol{M}_1 \widehat{U} = \boldsymbol{M}_2 U^{n+1}. \tag{3.4.7}$$

Note that the computation of $\widehat{U}$ is not trivial as it involves the computation of products between basis functions of $\mathbb{V}^n$ and $\mathbb{V}^{n+1}$. Over each element $\hat{K} \in \mathscr{T}^{n+1}$, the basis functions of $\mathbb{V}^n$ are in general discontinuous piecewise polynomials. Since the integrals over $\hat{K}$ are evaluated exactly at the quadrature points only for polynomials, each entry of $\boldsymbol{M}_2$ must be computed at the region of intersection between $\hat{K}$ and $K \in \mathscr{T}^n$. The collection of those meshed regions defines a conforming triangulation called *supermesh*.

The algorithm for constructing the supermesh can be summarized as follows:

Algorithm 5 Supermesh construction between $\mathscr{T}^n$ and $\mathscr{T}^{n+1}$

Require: $\mathscr{T}^n$,$\mathscr{T}^{n+1}$,$\mathcal{V}_{\mathscr{T}}$
Ensure: Scalar field $\widehat{U}^{n+1}$ transferred to adapted mesh $\mathscr{T}^{n+1}$
1: **for** $\widehat{K} \in \mathscr{T}^{n+1}$ **do**:
2: Identify all the intersecting elements $K \in \mathscr{T}^n$
3: Create the mesh $\mathscr{T}_{K,\widehat{K}}$ in $\widehat{K} \cap K$
4: Assemble $\boldsymbol{M}_2$ by integrating $\boldsymbol{\Phi}^n|_K$ and $\boldsymbol{\Phi}^{n+1}|_{\widehat{K}}$ over $\mathscr{T}_{K,\widehat{K}}$
5: Assemble locally the matrix $\boldsymbol{M}_1|_{\widehat{K}}$ and compute $\widehat{U}^{n+1}|_{\widehat{K}}$ from (3.4.7)
6: If $\mathcal{V}_{\mathscr{T}}$ is continuous, solve eq.(3.4.7) globally
7: **end for**
8: Assemble the vector $\widehat{U}^{n+1} \in \mathbb{V}^{n+1}$

For a given element $\hat{K} \in \mathscr{T}^{n+1}$, we must ensure all possible intersecting elements in $\mathscr{T}^n$ are found efficiently. The greedy algorithm performs $O(|\mathscr{T}^{n+1}||\mathscr{T}^n|)$ intersection tests. These can be first filtered by an Axis Aligned Bounding Box tree algorithm [Har16], which only consider pairs with intersecting bounding boxes. The construction takes $O(|\mathscr{T}^n| \log |\mathscr{T}^n|)$, and the query for each element takes $O(\log |\mathscr{T}^n|)$. This is the strategy we utilise.

The vertices of the intersecting polygon can be computed with the *Sutherland-Hodgman clipping algorithm* [SH74]. The meshing procedure is performed connecting with an edge a generic vertex with all the other vertices except the adjacent ones. If $K \cap \hat{K}$ is formed by n vertices, with $4 \leq n \leq 6$, the intersecting region can be decomposed in $n-2$ triangles. For each triangle K the integral $\int_{K\cap\hat{K}} \boldsymbol{\Phi}_i^{n+1} \boldsymbol{\Phi}_j^n \,\mathrm{d}\boldsymbol{x}$ can be computed exactly using the Gaussian quadrature rule. An illustration of the procedure is given in Figure 3-1. Note that, for the lowest order dG space, only the area $|K \cap \hat{K}|$, which is computed with the *shoelace method* [Mei69], is needed to assemble $\boldsymbol{M}_2$.

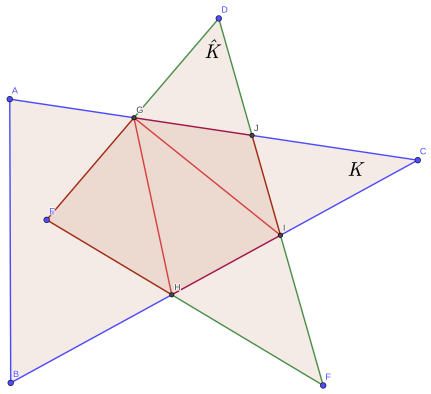


Figure 3-1: An illustration of the intersection between two triangles. The resulting polygon is subdivided into a shape regular triangulation which represents the supermesh over the intersection.

3.5 Numerical Experiments

In this section we test the methods described in §3.2–3.4. The implementation has been carried out using the DOLFIN interface of FEniCS [LW10]. The software Paraview has been used as visualisation tool. For each test, we used a Gaussian quadrature rule of order 6 and took $k = 0$ in $\mathbb{V}$ for simplicity.

3.5.1 Test 1 - Mesh adaptation to a scalar function

In this first experiment, we test the effectivity of Algorithm 4 to construct meshes that resolve certain prescribed functions with differing properties and regularity (Figure 3-3). With that in mind, let $\mathscr{T}_i$ denote the sequence of meshes generated by Algorithm 4 from an initial uniform criss-cross mesh. Let $\Pi_i u \in \mathbb{V}_i := \mathbb{V}(\mathscr{T}_i)$ be the L^2 projection of a function u.

Consider the following three functions:

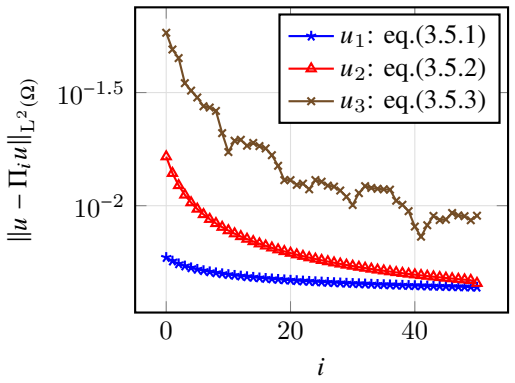
$$\text{Gaussian}: \quad u_1(x,y) = \exp\left(-50\Big((x-0.5)^2 + (y-0.5)^2\Big)\right), \tag{3.5.1}$$

$$\text{Smooth function}: \quad u_2(x,y) = 3 - \tanh\left(30\Big((x-0.3) + (y-0.2)\Big)\right), \tag{3.5.2}$$

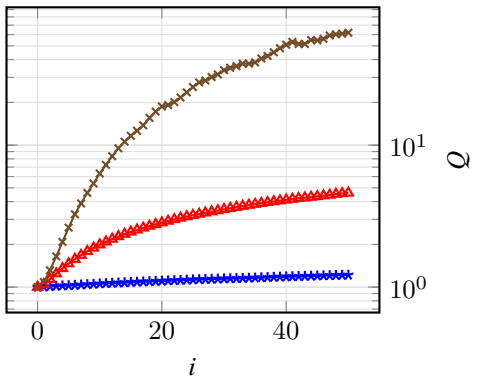
$$\text{Discontinuous function}: \quad u_3(x,y) = \begin{cases} 3.0 & \text{if } y + 0.8(x-0.2) <= 0, \\ 1.0 & \text{otherwise.} \end{cases} \tag{3.5.3}$$

These have been chosen to show numerically the difference in accuracy of the approximate solutions for decreasing smoothness in the initial condition, from u_1 to u_3.

In Figure 3-2a we show the behaviour of $\|u - \Pi_i u\|_{\mathrm{L}^2(\Omega)}$ and in Figure 3-2b the quality measure Q for each of the meshes produced for the functions (3.5.1), (3.5.2) and (3.5.3) using Algorithm 4.



(a) The projection error as a function of iterations of the discretisation of the MMPDE.



(b) The mesh quality as a function of iterations of the discretisation of the MMPDE.

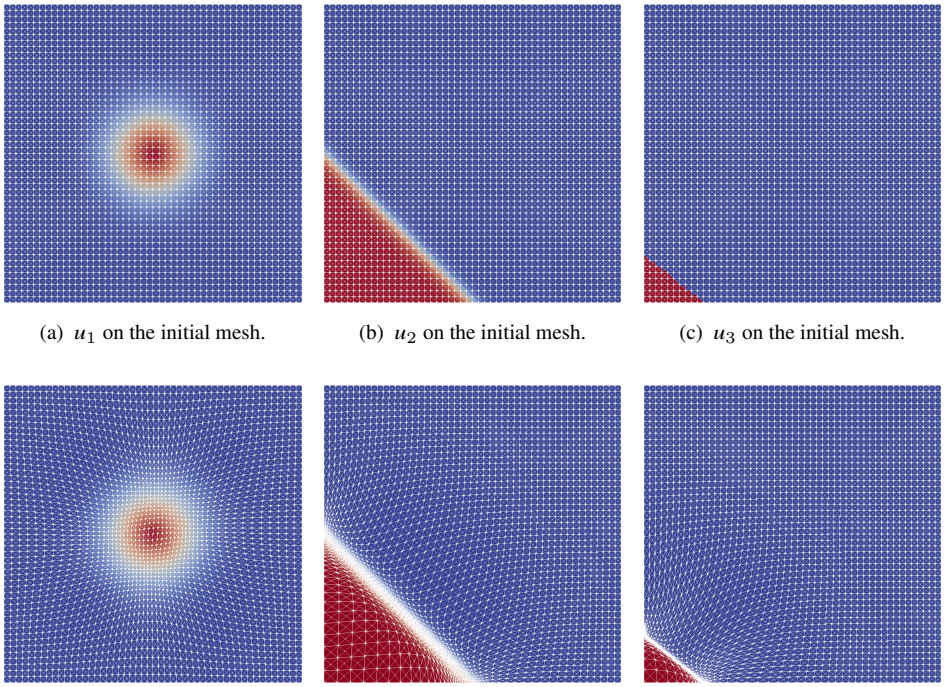
Figure 3-2: Test 1 - Here we show the projection error and the mesh quality measure Q for u_1, u_2, u_3 defined in eq.(3.5.1)–(3.5.3) using Algorithm 4 for 50 iterations. The initial criss-cross computational mesh is fixed and composed of 10^4 elements, whereas the physical mesh is then iteratively adapted. In this experiment we took the time step $k = 10^{-3}$ (3.3.4) and the smoothing parameter $\beta = 7 \times 10^{-4}$ (3.3.10). Note that the discontinuous function is approximated almost as well as the other functions despite the discontinuity not being aligned with the mesh. To compensate, as expected, the mesh quality measure increases dramatically. This results in quite anisotropic elements.

3.5.2 Test 2 - Convergence on the adaptively generated grids

In this test, we examine the approximability of the finite element spaces over the meshes generated by the moving mesh procedure in Algorithm 4. We consider a family of uniform criss-cross computational meshes, each one a uniform refinement of the previous mesh. For each of these initial meshes, we use Algorithm 4 to generate a mesh $\mathcal{T}_{N_S}$, which is able to resolve the features of the functions (3.5.1)–(3.5.3). For each realisation of $\mathcal{T}_{N_S}$, we let Πu denote the L^2 projection onto the finite element space associated to this mesh and we test the asymptotic convergence rates, measured in terms of the degrees of freedom.

Figure 3-3 shows the functions together with a visualisation of the result of applying Algorithm 4. Figure 3-4 shows the convergence of the best approximation over the uniform and warped meshes. Notice that the functions are better approximated over the warped meshes, but also that the discontinuous solution is approximated optimally despite the lack of regularity. This is consistent with other studies of irregular functions on warped meshes [GMP17].

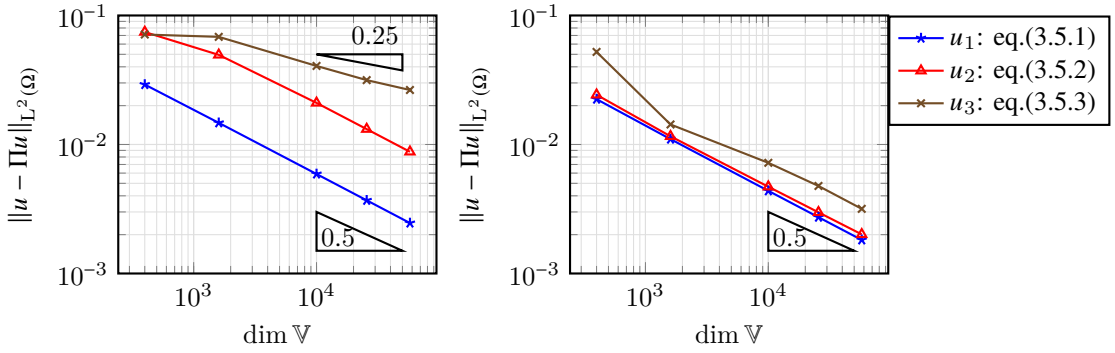
1 Adaptive-conservative dG advection equation with upwinding This last set of experiments detail the validation of the fully coupled adaptive conservative moving mesh dG scheme for the numerical solution of the advection equation (3.2.3) via Algorithm 3. It employs both the MMPDE (Algorithm 4)



(a) u_1 on the initial mesh. (b) u_2 on the initial mesh. (c) u_3 on the initial mesh.

(d) u_1 on the final mesh. (e) u_2 on the final mesh. (f) u_3 on the final mesh.

Figure 3-3: Test 1 - Visualisations of the functions u_1, u_2, u_3 defined in (3.5.1))-(3.5.3)) discretised in $\mathbb{V}_i$, a criss cross mesh on the domain $\Omega = [0,1]^2$ at initial iteration ($i = 0$), and final iteration ($i = 50$). The physical mesh $\mathscr{T}$ is comprised by 10^4 triangular elements and for $i = 0$ equal to the criss-cross computational mesh. Note that the mesh distorts locally to ensure good approximation of the hard-to-approximate features.



(a) A family of uniform meshes. (b) A family of adaptive meshes.

Figure 3-4: Test 2 - Asymptotic convergence rates for the best approximation on the finite element space over the meshes generated by Algorithm 4. Note that for uniform meshes the approximation of the discontinuous function is slower due to the lack of regularity. The adapted meshes are able to resolve this to ensure optimal rates for the piecewise constant approximations [GMP17].

and the conservative Galerkin projection as a data transfer operator.

2 Test Problems We prescribe two different initial conditions

$$u_0(\boldsymbol{x}) = 3 - \tanh\big(30\big((x-0.2)+(y-0.2)\big)\big), \tag{3.5.4}$$

$$u_0(\boldsymbol{x}) = \begin{cases} 3.0 & \text{if } y + 0.8(x-0.2) <= 0, \\ 1.0 & \text{otherwise,} \end{cases} \tag{3.5.5}$$

for $\Omega := [0,1]^2$ and set a constant velocity field

$$\boldsymbol{v} = \left(\tfrac{1}{2}, \tfrac{\sqrt{2}}{2}\right), \tag{3.5.6}$$

as not to align with the mesh. We impose an inflow boundary condition consistent with the exact solution

$$u(\boldsymbol{x}, t) = u_0(\boldsymbol{x} - \boldsymbol{v}t) \tag{3.5.7}$$

over $\partial\Omega_-$, which corresponds to the west and south sides of Ω. The solutions we consider are a smooth travelling front and a discontinuous travelling front.

3.5.3 Test 3 - Error accumulation in time

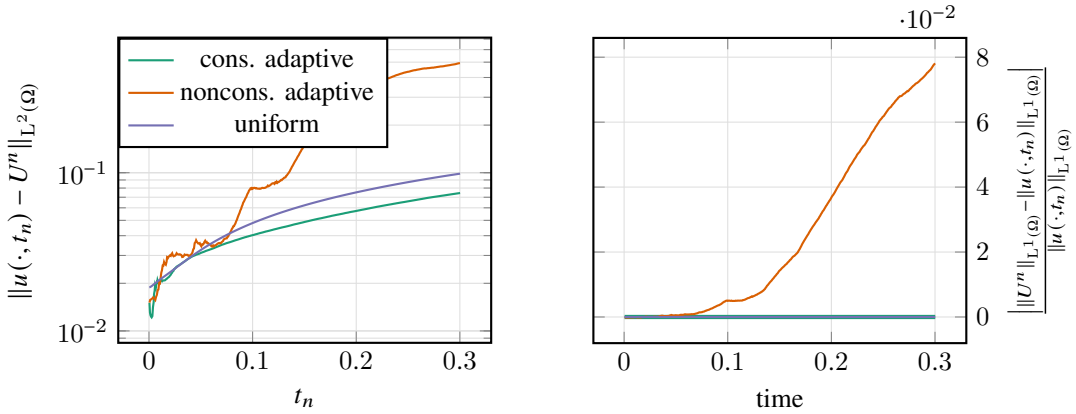
In this test, for the smooth front initial condition given by (3.5.2), we compare the performance of the proposed adaptive scheme with and without the conservative data transfer projection to that one of the uniform scheme. As such, we are able to show the importance of the conservative data transfer operator for the success of the scheme. To that end, we take an initial uniform triangulation formed of 10^4 criss-cross elements. The mesh is then adapted to the initial condition using Algorithm 4. The advection equation is then solved using the three-stage SSPRK-dG scheme (3.2.12).

The results are shown in Figure 3-5, where it is apparent that the L^2 error over time grows at a smaller rate for the conservative adaptive scheme than its uniform and non-conservative counterparts.

3.5.4 Test 4 - Sensitivity of the MMPDE algorithm

In this test, we explore the sensitivity of the MMPDE described in Algorithm 4. We fix the initial triangulation to be of criss-cross type with 10^4 elements. We fix the parameters $k = 10^{-3}, \beta = 10^{-4}$ and vary the parameter S within Algorithm 4.

Figure 3-6 shows results by varying the parameter S for both initial conditions. The SSPRK-dG scheme is kept the same. The key point is that the MMPDE must be solved in the time interval $[0, S]$ for S large enough to ensure that (3.3.3) is close enough to reach a steady state.



(a) Error accumulation in time for the uniform, adaptive nonconservative and adaptive conservative schemes.

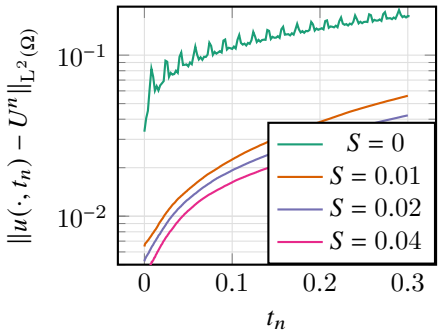
(b) Relative mass error for the uniform, adaptive nonconservative and adaptive conservative schemes.

Figure 3-5: Test 3 - Here we show the error accumulation in time for the smooth initial condition (3.5.4). The parameters used for the mesh adaptation are: $N_S = 15$, $k = 10^{-3}$ and $\beta = 10^{-5}$. In each case the initial mesh was chosen such that $\dim \mathbb{V} = 10^4$. The uniform scheme outperforms the non-conservative adaptive one as the error introduced by Lagrange interpolation from $\mathscr{T}^n$ to $\mathscr{T}^{n+1}$ is not counterbalanced by the gain in accuracy on the adapted mesh.

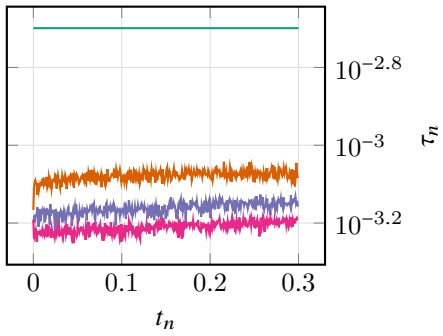
3.5.5 Test 5 - Asymptotic convergence rates

In this test, we compare the adaptive-conservative scheme to an equivalent SSPRK-dG scheme running on the initial uniform mesh. In both cases, the time step has been chosen adaptively to ensure a CFL condition is satisfied. For the uniform scheme, we take a family of consecutively refined initial meshes where $\dim \mathbb{V} \in [100, 6400]$ and run the SSPRK-dG scheme (3.2.12). For the adaptive scheme, we take the same initial meshes; however this time apply Algorithm 4. In both cases we track the L^2 error evaluated as a function of time as well as the time step.

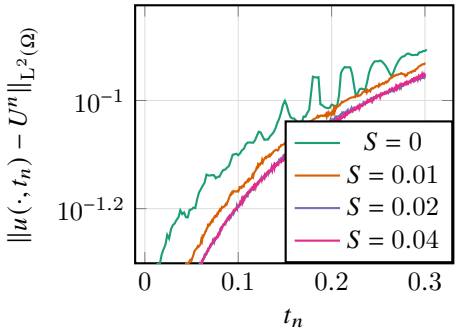
We test two exact solutions, a smooth moving front and a discontinuous moving front. The results are shown in Figures 3-7 –3-8, where we are able to see an improved convergence rate for the adaptive scheme applied to both test problems over the uniform one.



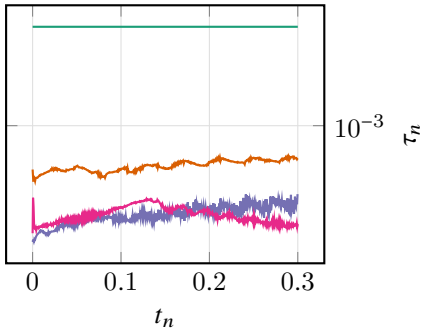
(a) The L^2 error as a function of time for the adaptive approximation of (3.5.7) with smooth initial conditions (3.5.4).



(b) The timestep as a function of time for the adaptive approximation of (3.5.7) with smooth initial conditions (3.5.4).

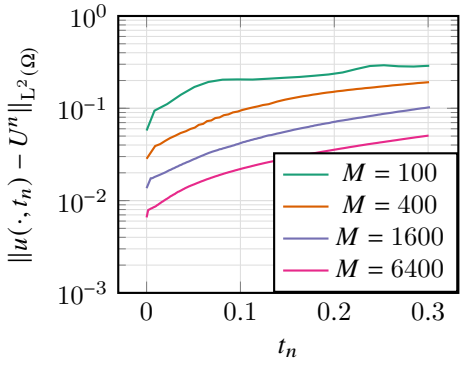


(c) The L^2 error as a function of time for the adaptive approximation of (3.5.7) with discontinuous initial conditions (3.5.5).

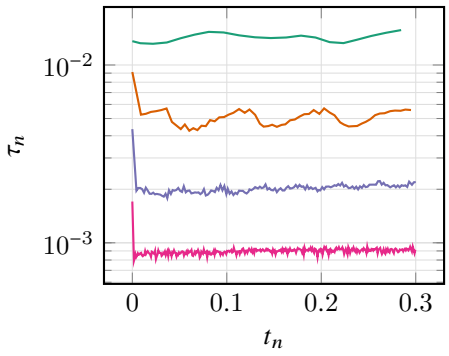


(d) The timestep as a function of time for the adaptive approximation of (3.5.7) with discontinuous initial conditions (3.5.5).

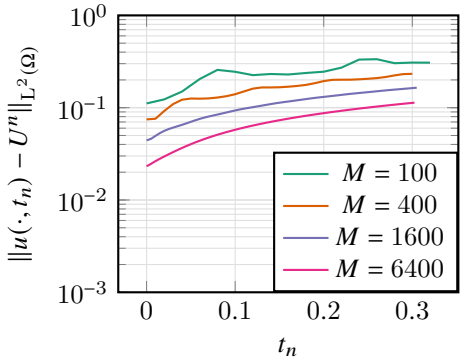
Figure 3-6: Test 4 - Here we show the effect of varying S, the end time of the moving mesh algorithm, within Algorithm 4. We fix $k = 10^{-3}$ and $\beta = 10^{-4}$. We examine the effect of modifying S for the solutions given by (3.5.7) with initial conditions (3.5.4) and (3.5.5). We show that the error stagnates as S gets large, meaning that the solution of the MMPDE is close to a steady state for that particular S.



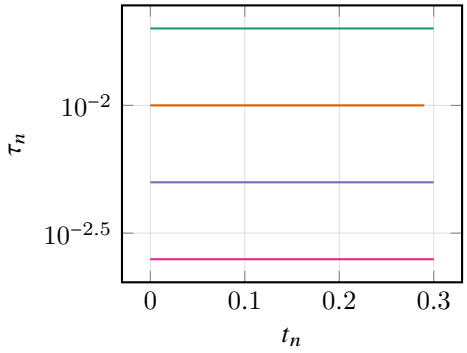
(a) The L^2 error as a function of time for the adaptive approximation.



(b) The time step as a function of time for the adaptive approximation.

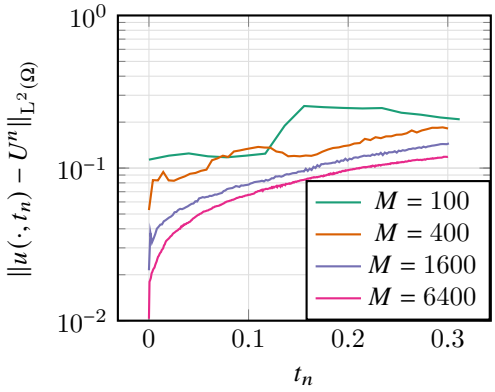


(c) The L^2 error as a function of time for the uniform approximation.

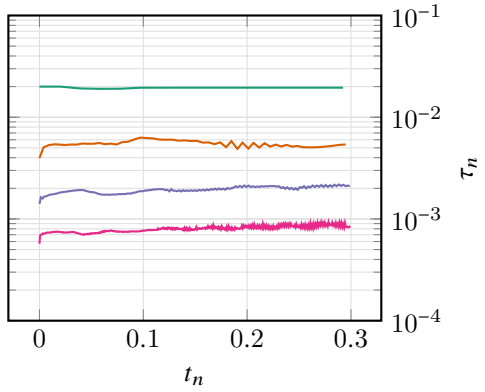


(d) The time step as a function of time for the uniform approximation.

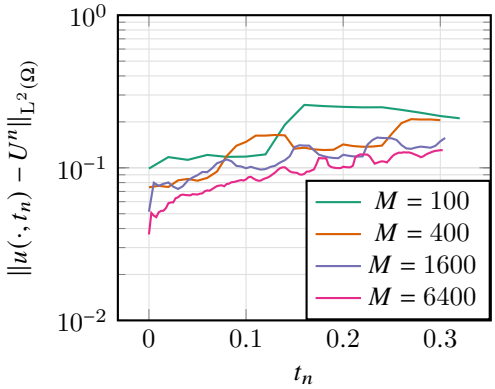
Figure 3-7: Test 5 - A comparison of the adaptive scheme given by Algorithm 3 and an equivalent uniform scheme, both initialised with the same uniform meshes. The family of uniformly refined initial grids ensures $\dim \mathbb{V} \in [100, 6400]$. We are examining the approximation of the exact solution (3.5.7) with the smooth initial condition (3.5.4). Notice that the adaptive scheme is able to achieve an approximation with smaller L^2 error than the uniform scheme as well as a higher rate of convergence.



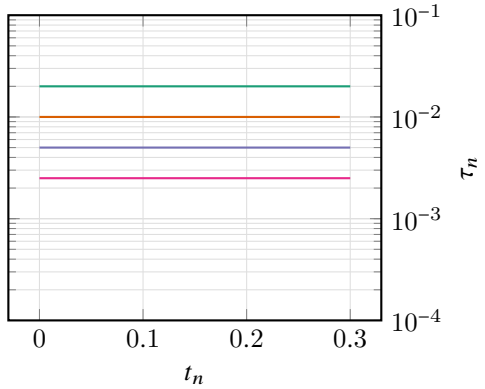
(a) The L^2 error as a function of time for the adaptive approximation.



(b) The time step as a function of time for the adaptive approximation.



(c) The L^2 error as a function of time for the uniform approximation.



(d) The time step as a function of time for the uniform approximation.

Figure 3-8: Test 5 - A comparison of the adaptive scheme given by Algorithm 3 and an equivalent uniform scheme, both initialised with the same uniform meshes. The family of uniformly refined initial grids ensures $\dim \mathbb{V} \in [100, 6400]$. We are examining the approximation of the exact solution (3.5.7) with the discontinuous initial condition (3.5.5). Notice that the adaptive scheme is marginally more accurate than the uniform scheme.

3.6 Summary

In this chapter, we have proposed and studied an r-adaptive algorithm for the solution of a linear hyperbolic PDE. The moving mesh method is able to adapt the mesh to increase the spatial accuracy of the solution in the L^2 norm. We achieved this by making use of a conservative Galerkin projection as a data transfer operator via the construction of a *supermesh*.

While the scheme itself shows good error reduction properties, it has a number of shortcomings. In particular, for each time step the solution of a quasilinear parabolic PDE is required to compute the solution to the Winslow' mesh adaptation step. This dramatically increases computational cost over, say an h- or p-adaptive method, to achieve the same error tolerance. That being said, the method has the potential for massive parallelisation since at any time step, the mesh is just an affine transformation of a uniform grid.

The next chapter will use r-adaptivity for the solution of the two-dimensional Poisson's equation in non-convex domains. Moreover, we will expand our analysis by employing a h-refinement strategy and comparing the two adaptive mesh strategies. Since the equation is elliptic, the rezoning approach is not required, but the non-convexity of the domain raises several issues for the accurate solution of the PDE. We will also formulate an algorithm to construct an Optimal Transport mesh as alternative to the use of Winslow's method.

Chapter 4

Optimal Transport and h-adaptive based mesh generation for Poisson's equation in non-convex domains

Abstract

We solve the two-dimensional Poisson's equation using the symmetric interior penalty discontinuous Galerkin method on non-convex domains. The solution exhibits a singularity at the re-entrant corner and lies in a weighted Sobolev space. Under this framework, we derive a *novel* L^2 a-posteriori estimate.

We discretise a L-shaped and crack domain on meshes using an h-adaptive and an Optimal Transport based mesh strategy. The h-refinement strategy will be guided by the proposed L^2 and L^∞ a-posteriori estimate, each displaying numerically optimal convergence rate. The Optimal Transport mesh is derived semi-analytically, by considering the local behaviour of the solution near the re-entrant corner. This second procedure exhibits the same accuracy as the h-refinement strategy but is computationally less expensive. On the contrary, the Winslow's method generates meshes that yields low order of convergence for different monitor functions.

Numerical experiments show that the *skewness* of the Optimal Transport mesh elements does depend only on the angle of the re-entrant corner. Moreover, the resulting Optimal Transport mesh is able to equidistribute the a-posteriori estimates. This hints at the close relationship between Optimal Transport and h-adaptation for the common goal of equidistributing the local interpolation error.

4.1 Introduction

Solutions of second order elliptic boundary value problems defined on non-convex domains typically display singular behaviour at the re-entrant corners and lie in Sobolev spaces H^k for small $k > 1$ [BG88; BG89]. The theoretical framework of weighted Sobolev spaces, which were originally studied in [BR72; BKP79] for elasticity and potential problems, can deal with the low regularity of the solution and will be used for discretising Poisson's equation. To obtain accurate solutions of such problems, several procedures have been proposed. In [SH16; ZS02; ZSG02] the solution of Poisson's equation is treated via a singular decomposition $u = w + \lambda s$, where w is regular, s is a singular function, and the coefficient λ is the so called *stress intensity factor*. Alternatively, the mesh tessellating the domain can be adapted to the problem via an h-,p-adaptive strategy based on some form of solution error bound [BKP79; BG92]. In [CD02] the authors analyse a second order elliptic equation with strongly discontinuous coefficients and derive an a-posteriori error estimator which depends locally on the oscillations of the coefficients around singular points. The extension to low-order nonconforming finite element methods on both triangular and quadrilateral meshes, with hanging nodes allowed for local mesh refinement, has been addressed in [KH10]. Finally, the reduction rate of data oscillation, missed by the averaging process associated with finite element methods (FEM), together with an a-posteriori error, was used to construct an efficient adaptive FEM for two- and three-dimensional elliptic partial differential equations (PDEs) with linear rate of convergence [MNS00].

In this chapter we treat the two-dimensional Poisson's equation defined on two non-convex domains with prescribed Dirichlet boundary conditions. The discretisation is performed with the symmetric interior penalty discontinuous Galerkin (SIP-dG) method, formulated in [Arn82; Arn+00; Arn+02]. The popularity of the dG method resides in its flexibility with respect to adapted elements of various types and shapes. In order to account for the singularities, we employ appropriate adaptive mesh strategies to obtain optimal convergence rates for the dG solution.

In the first part of the Chapter we will derive a new *a-posteriori* error estimate for the solution in the L^2 norm. The proof relies on the dual weak formulation of the Poisson problem and on the properties of the weighted Sobolev spaces [Sch98; Wih03]. An L^∞ a-posteriori bound will be also used for the subsequent numerical experiments [DG12]. The a-posteriori bounds will then be incorporated into an h-adapted SIP-dG method to solve Poisson's equation on both a L-shaped and crack domain. The rates of convergence of the solution in the L^2 and L^∞ norms will be computed using the corresponding a-posteriori bounds.

In the second part of the Chapter we will construct r-adapted mesh by using Winslow's method and an Optimal Transport (OT) strategy guided by an *a-priori* estimate of the *interpolation error* of the solution. The former method performs poorly on the proposed problem with different choices of monitor functions. The latter method is an adaptation of the OT mesh adaptive methods described in [BW09], which rely on the derivation of a local map from a computational to a physical domain (in which the underlying equation

is posed) through the solution of an appropriate Monge-Ampère equation. This new OT method exploits the local structure of the solution of Poisson's equation in the vicinity of the re-entrant corner, where the solution strongly depends on the radial distance to the corner. This allows us to compute the OT mesh via solving a one-dimensional non-algebraic equation, which is much less computationally expensive than the fully two-dimensional Monge-Ampère equation [BW09; Pry12; MCB18]. To implement this method, it is crucial to choose an appropriate value of an meta-parameter γ, which controls the level of node clustering near the re-entrant corner, ultimately affecting the accuracy, convergence rate of the SIP-dG method and the quality of the mesh. The optimal values of γ, in terms of accuracy and convergence rate, can be determined through a-priori considerations of the solution interpolation error.

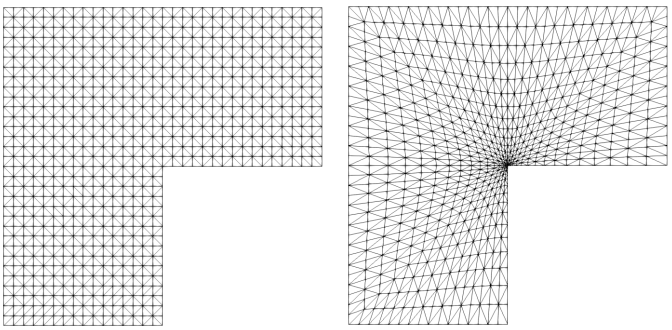
Through a series of careful numerical experiments we will demonstrate that the OT approach gives near optimal rates of convergence of the interpolation error, with results very close to those of the *h*-adaptive refinement. However, the computational cost is significantly lower. It will be also proved theoretically and by numerical tests that the adapted meshes show good regularity properties, and that the *mesh quality measure* depends only on the angle of the re-entrant corner.

The remainder of the Chapter is structured as follows. In §4.2 we introduce the notation of the weighted Sobolev spaces for the SIP-dG discretisation of Poisson's equation. In §4.3 we derive the a-posteriori error estimator in the L^2 norm from the dual formulation of the original problem. In §4.4.2 we describe the procedure for constructing the OT mesh and derive theoretically the optimal meta-parameters based on the equidistribution of the local interpolation error. We will investigate the quality of the OT mesh in terms of various metrics. In §4.5 we will conduct numerical tests on the L-shaped and crack domain. Initially, we will assess the accuracy of the SIP-dG method generated on the *h*-adaptive meshes using the a-posteriori estimates. Then, we will show that the optimal meta-parameters derived from considering the OT based strategy are close to the optimal ones showed numerically and equidistribute the a-posteriori estimates. Finally, we will draw our conclusions in §4.6.

4.2 Problem setup and discretisation

The main aim of this Chapter is to study two different adaptive strategies coupled to a discontinuous Galerkin method on a (non-uniform) mesh, for solving elliptic equations on a non-convex domain Ω with a re-entrant corner of varying angle. A re-entrant corner is defined as any internal corner that forms an angle of 180ř degrees or less. It is well known that the solution u of Poisson's equation has gradient singularity at this corner, leading to a degrading solution accuracy if a uniform mesh is used. As a result, we will be considering non-uniform meshes generated by using either an *h*-adaptation or a *r*-adaptation method.

An example of such a non-convex domain (a L-shaped domain) is given in Figure 4-1, with a mesh given by either a uniform or a non-uniform triangulation (generated by the OT method in §4.4.5).



(a) uniform triangulation. (b) OT mesh.

Figure 4-1: An example of a non convex domain Ω (a L-shaped domain), with a uniform triangulation and a non-uniform graded triangulation.

In this section we detail some definitions related to non-convex domains, weighted Sobolev spaces where the solution of Poisson's equation lies, and introduce the abstract framework used to approximate such solution.

Let $\Omega \subset \mathbb{R}^d,\ d \geq 2$ be an open, bounded, polygonal domain with outward unit normal $\boldsymbol{n}_\Omega$. Suppose that the boundary $\Gamma = \partial\Omega$ is composed by a Dirichlet part Γ_D with $|\Gamma_D| > 0$ and a Neumann part Γ_N such that:

$$\overline{\Gamma} = \overline{\Gamma}_D \cup \overline{\Gamma}_N . \tag{4.2.1}$$

All the vertices of Ω are included in the closure. The corner vertices and the points of changing boundary conditions are *singular points* and collected in the set

$$SP(\Omega, \Gamma_D, \Gamma_N) = \{A_i : i = 1, \ldots, M\}. \tag{4.2.2}$$

The interior opening angle of the domain at A_i is measured anti-clockwise and denoted by $\omega_i \in (0, 2\pi]$. If $\omega_i > \pi$ then the point A_i is a re-entrant corner and the solution will have a gradient singularity there.

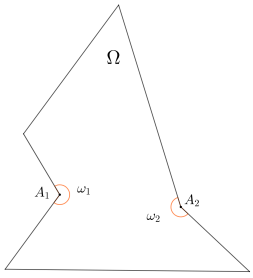


Figure 4-2: Example of a non-convex domain Ω with two re-entrant corners.

Definition 4.2.1 (L-shaped and crack domain.) *If $d = 2$ and $\omega_i = 3\pi/2$ we call A_i a* corner *(and if there is only one such point) then Ω is called an* L-shaped domain. *Similarly if $\omega_i = 2\pi$, then A_i is called a* crack *and Ω a* crack domain.

To account for the change in the regularity of the elliptic problem at the re-entrant corners, suitable Sobolev spaces will be introduced. A weight $\beta_i \in (0, 1]$ is associated with each singular point $A_i \in SP(\Omega, \Gamma_D, \Gamma_N)$ and stored in the vector $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_M)$. For any number $k \in \mathbb{R}$, we let

$$\boldsymbol{\beta} + k = (\beta_1 + k, \ldots, \beta_M + k),$$
$$C_1 \le \boldsymbol{\beta} < C_2 \;\Rightarrow\; C_1 \le \beta_i < C_2 \text{ for } i = 1, \ldots, M.$$

We then introduce the weight function on Ω:

$$\Phi_{\boldsymbol{\beta}}(\boldsymbol{x}) = \prod_{i=1}^{M} r_i(\boldsymbol{x})^{\beta_i}, \quad r_i(\boldsymbol{x}) = |\boldsymbol{x} - A_i|, \quad \boldsymbol{x} \in \Omega.$$

Then, for any integers $m \ge l \ge 0$, the weighted Sobolev spaces $\mathrm{H}^{m,l}_{\boldsymbol{\beta}}(\Omega)$ are defined as completion of the space $C^\infty(\overline{\Omega})$ with respect to the weighted Sobolev norms

$$\|u\|^2_{\mathrm{H}^{m,l}_{\boldsymbol{\beta}}(\Omega)} = \|u\|^2_{\mathrm{H}^{l-1}(\Omega)} + |u|^2_{\mathrm{H}^{m,l}_{\boldsymbol{\beta}}(\Omega)} \;\; l \ge 1, \quad \|u\|^2_{\mathrm{H}^{m}_{\boldsymbol{\beta}}(\Omega)} = \sum_{\substack{|\alpha|=k \\ k=0}}^{m} \left\| |D^\alpha u| \Phi_{\boldsymbol{\beta}+k} \right\|^2_{\mathrm{L}^2(\Omega)} \;\; l = 0, \quad (4.2.3)$$

where

$$|u|^2_{\mathrm{H}^{m,l}_{\boldsymbol{\beta}}(\Omega)} = \sum_{\substack{|\alpha|=k \\ k=l}}^{m} \left\| |D^\alpha u| \Phi_{\boldsymbol{\beta}+k-l} \right\|^2_{\mathrm{L}^2(\Omega)}, \quad D^\alpha u = \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2}}, \quad (4.2.4)$$

with $\alpha = (\alpha_1, \alpha_2) \in \mathbb{N}^2$ and $|\alpha| = \alpha_1 + \alpha_2$. When $\boldsymbol{\beta} = \mathbf{0}$, we recover the standard Hilbert spaces as $\mathrm{H}^{m,l}_{\boldsymbol{\beta}}(\Omega) \equiv \mathrm{H}^m(\Omega)$.

Dirichlet boundary conditions are determined by the existence of a suitable trace lifting of the Dirichlet data. The set of all functions in $\mathrm{L}^2(\Gamma)$, which are boundary values of functions in $\mathrm{H}^m(\Omega)$ is denoted by $\mathrm{H}^{m-1/2}(\Gamma)$ and the continuous, linear mapping $\tau : \mathrm{H}^m(\Omega) \to \mathrm{H}^{m-1/2}(\Gamma)$ given by

$$\tau : \mathrm{H}^m(\Omega) \to \mathrm{H}^{m-1/2}(\Gamma), \quad m > 1/2$$

is called the *trace operator* [Sch98, Section 1.4].

Remark 4.1 (Properties of the weighted Sobolev spaces [Wih03, Remark 1.2.2]) *The following properties hold:*

1. *If* $u \in \mathrm{H}^{m,m}_{\boldsymbol{\beta}}(\Omega)$, $m \geq 0$, *then* $u \in \mathrm{H}^m(\Omega_0)$ *for all domains* $\Omega_0 \subset \Omega$ *with*

$$P \notin \overline{\Omega_0} \quad \forall P \in SP(\Omega, \Gamma_D, \Gamma_N).$$

2. *Although* $\mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega) \not\subset \mathrm{H}^2(\Omega)$, *it was proven in [BKP79, equation 2.2] that* $\mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega) \subset \mathscr{C}^0(\overline{\Omega})$.

3. *For* $u \in \mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega)$, *there holds* $\nabla u \in \mathrm{H}^{1,1}_{\boldsymbol{\beta}}(\Omega)^2$.

We now introduce in the abstract setting the framework to derive the L^2 a-posteriori estimate. Let V be a Banach space and $\mathscr{A}(\cdot,\cdot) : V \times V \to \mathbb{R}$ a symmetric bilinear form and $l : V \to \mathbb{R}$ be a linear form. We will consider the abstract problem of finding $u \in V$ such that

$$\mathscr{A}(u, v) = l(v) \quad \forall v \in V.$$

We will also assume that the $\mathscr{A}(\cdot,\cdot)$ is bounded and coercive (Def.2.3.2 and Def.2.3.3 in Chapter 2). We will apply this setting to solve the Poisson problem with a discontinuous Galerkin (dG) method on a non-uniform mesh.

Properties of adapted mesh

Let $\mathscr{T}$ be a conforming mesh of Ω divided into elements (cells) K as in Def. 2.3.4 of Chapter 2.

We suppose that all the meshes generated by h-refinement procedures are *shape regular*, so that any individual cell does not depart too much from being regular (for example it does not have vanishingly small angles). In particular, if $\mathscr{G} = \{\mathscr{T}_i\}_{i\in\mathbb{N}}$ is a family of meshes, then $\mathscr{G}$ is characterised by the shape regularity $\mu(\mathscr{T})$ if

$$\mu(\mathscr{T}) := \min_{K\in\mathscr{T}_i} \left(\mu_K \equiv \frac{h_K}{\rho_K} \right) > 0. \tag{4.2.5}$$

The set of all edges of $\mathscr{T}$ is denoted by $\mathscr{E}$, which we partition into subsets $\mathscr{E}_D, \mathscr{E}_N, \mathscr{E}_I$, consisting of edges lying on the Dirichlet boundary Γ_D, the Neumann boundary Γ_N, and the interior edges, respectively. The corresponding quantities for each individual element K are denoted by $\mathscr{E}(K)$, $\mathscr{E}_D(K)$, $\mathscr{E}_N(K)$, $\mathscr{E}_I(K)$, respectively. We split the index set $\{1, \ldots, M\}$ of boundary edges e_i into $\mathscr{D}$ and $\mathscr{N}$, on which Dirichlet and Neumann boundary conditions are applied, respectively. This leads to $\overline{\Gamma}_D = \bigcup_{i\in\mathscr{D}} \overline{e}_i$ and $\overline{\Gamma}_N = \bigcup_{i\in\mathscr{N}} \overline{e}_i$.

The finite element space

We define the finite element space to be

$$\mathbb{V}^p := \{v \in \mathrm{L}^2(\Omega) : v|_K \in \mathbb{P}^p(K) \quad \forall K \in \mathscr{T}\}, \tag{4.2.6}$$

where $\mathbb{P}^p(K)$ is the space of polynomials of total degree p. For the remainder of the Chapter, we denote the space of piecewise linear polynomials by $\mathbb{V}$, with $p = 1$. We also remark that the space $\mathbb{V}$ does not carry any inter-element continuity and does not encode any boundary condition. These will be enforced weakly in the variational scheme.

Given an edge $e \in \mathscr{E}$, we define the *jump* and *average* operator of $v \in \mathbb{V}$ and $\boldsymbol{v} \in \mathbb{V}^2$ at $\boldsymbol{x} \in e$ by

$$[\![v]\!] := \begin{cases} v|_K \boldsymbol{n}_K + v|_{K'} \boldsymbol{n}_{K'} \\ v|_K \boldsymbol{n}_\Omega \end{cases} \qquad [\![\boldsymbol{v}]\!] := \begin{cases} \boldsymbol{v}|_K \cdot \boldsymbol{n}_K + \boldsymbol{v}|_{K'} \cdot \boldsymbol{n}_{K'} & \text{on } e = \partial K \cap \partial K' \\ \boldsymbol{v}|_K \cdot \boldsymbol{n}_\Omega & \text{on } e = \partial K \cap \Gamma_D \end{cases} \tag{4.2.7}$$

$$\{\!\{ v \}\!\} := \begin{cases} \frac{1}{2}(v|_K + v|_{K'}) \\ v|_K \end{cases} \qquad \{\!\{ \boldsymbol{v} \}\!\} := \begin{cases} \frac{1}{2}(\boldsymbol{v}|_K + \boldsymbol{v}|_{K'}) & \text{on } e = \partial K \cap \partial K' \\ \boldsymbol{v}|_K & \text{on } e = \partial K \cap \Gamma_D \end{cases} \tag{4.2.8}$$

Here $v|_K$ denotes the trace of v onto the edge $e \in K \cap K'$ and $\boldsymbol{n}_K$ is the outward unit vector relative to K on e. For $e \subset \Gamma$, we define $\{\!\{ v \}\!\} := v$, $\{\!\{ \boldsymbol{v} \}\!\} := \boldsymbol{v}$, $[\![v]\!] := v\boldsymbol{n}_\Omega$, and $[\![\boldsymbol{v}]\!] := \boldsymbol{v} \cdot \boldsymbol{n}_\Omega$.

We will use the following jump identity for the derivation of the a-posteriori error estimate:

$$[\![\boldsymbol{u}v]\!] = [\![\boldsymbol{u}]\!] \{\!\{ v \}\!\} + \{\!\{ \boldsymbol{u} \}\!\} [\![v]\!] . \tag{4.2.9}$$

Using the jump and average operators we can introduce the mesh condition as a further quality measure of the mesh:

$$q(\mathscr{T}) := \| [\![h]\!] / \{\!\{ h \}\!\} \|_{\mathrm{L}^\infty(\mathscr{E}_I)} . \tag{4.2.10}$$

The quantity $q(\mathscr{T})$ measures how 'similar' adjacent mesh cells are. It has been proved in [GMP17] that the *inf-sup stability* of the bilinear form given by the weak formulation of Poisson's equation holds if there is a constant $\alpha \in \mathbb{N}^+$ such that $q(\mathscr{T}) \leq \alpha$.

In later sections we will show how we can estimate both $\mu(\mathscr{T})$ and $q(\mathscr{T})$ for both h-adapted and r-adapted meshes on Ω.

To account for the singular behaviour of solutions at the singular points of the polygon Ω, we define the set

$$\mathscr{K}_0 = \{K \in \mathscr{T} : P \in \overline{K} \text{ for some } P \in SP(\Omega, \Gamma_D, \Gamma_N)\}. \tag{4.2.11}$$

Let $K \in \mathscr{K}_0$. We will assume the mesh is enough refined such that exactly one singular point belongs to $\overline{K}$. The corresponding vertex is denoted by A_K and the corresponding weight by Φ_{β_K}. We denote by $\mathscr{E}_{A_K}$ an edge of $\mathscr{E}$ containing the vertex A_K, with $\mathscr{E}_{A_K} \subset \mathscr{E}(K)$. For the sake of notation, if there is only one singular point in Ω, we denote the set of edges with that endpoint by $\mathscr{E}_0$ and the weight by Φ_β.

We conclude the section by stating the following Lemma, which is used to prove the consistency of the discrete dG method for the Poisson problem and to derive the relative a-posteriori estimate:

Lemma 4.2 (Continuity at the interior edges [Wih03, Lemma 1.3.4]) *Let $u \in \mathrm{H}^{1,1}_{\boldsymbol{\beta}}(\Omega)^d$ for $0 \leq \beta \leq 1$ and $d = 1, 2$. Then, for an interior edge $e \in \mathscr{E}_I$ there holds $[\![u]\!] = 0$ almost entirely (a.e.) on e.*

4.2.1 The Poisson problem in 2D

We are interested in solving the following *Poisson problem*

$$\begin{aligned} -\Delta u &= f \text{ in } \Omega, \\ u &= u_D \text{ on } \Gamma_D. \\ \nabla u \cdot \boldsymbol{n}_\Omega &= g \text{ on } \Gamma_N. \end{aligned} \tag{4.2.12}$$

Here, f is a given datum lying on the dual space of $\mathrm{H}^1_0(\Omega)$ and denoted by $\mathrm{H}^{-1}(\Omega)$, $u_D \in \mathrm{H}^{1/2}(\Gamma_D)$ and $g \in \mathrm{H}^{-1/2}(\Gamma_N)$ are prescribed Dirichlet and Neumann boundary conditions, respectively. In the framework of weighted Sobolev spaces; the existence, uniqueness and regularity of the solution of problem (4.2.12) is stated in the following Theorem:

Theorem 4.3 (Regularity of Poisson's equation [Wih03, Theorem 2.2.1]) *Let Ω be a polygon in $\mathbb{R}^2$ and $m \geq 0$ a given integer. Then, there exists a weight vector $\boldsymbol{\beta}_{min}$ with $0 \leq \boldsymbol{\beta}_{min} < 1$ depending on the opening angles at the vertices $A_i \in SP(\Omega, \Gamma_D, \Gamma_N)$, such that for weight vectors $\boldsymbol{\beta}$ with $\boldsymbol{\beta}_{min} \leq \boldsymbol{\beta} < 1$ and for*

$$f \in \mathrm{H}^{m,0}_{\boldsymbol{\beta}}(\Omega),\ u_D \in \mathrm{H}^{m+3/2,3/2}_{\boldsymbol{\beta}}(\Gamma_D),\ g \in \mathrm{H}^{m+1/2,1/2}_{\boldsymbol{\beta}}(\Gamma_N),$$

the problem (4.2.12) has a unique solution $u \in \mathrm{H}^{m+2,2}_{\boldsymbol{\beta}}(\Omega)$.

Remark 4.4 (Value of β [BG88, Theorem 2.1 - Remark 3]) *In the case of Poisson's equation the value of β_{min} is well-known at each corner A_i and given by*

$$\beta_{min,i} = \begin{cases} 1 - \frac{\pi}{\omega} & \text{if } \overline{A_i A_{i+1}} \in \Gamma_D \text{ or } \Gamma_N, \\ 1 - \frac{\pi}{2\omega} & \text{otherwise.} \end{cases} \tag{4.2.13}$$

As the re-entrant corner widens β increases. This enhances the singularity of the solution u, which must be multiplied by Φ_β with high β in equation (4.2.3) to state Theorem 4.3.

4.2.2 The variational formulation of Poisson's equation

Since more than a decade, discontinuous Galerkin methods have been attractive due to their flexibility in handling general meshes, non-uniformity in degree of approximation and capturing the rough solutions more accurately [SW05]. We aim to employ this method for the problem (4.2.12). In particular, we decide to use this over a standard FE linear discretization to derive a novel a-posteriori estimate in the L^2 norm.

We consider the standard formulation of the interior penalty method of Poisson's equation (4.2.12):

Definition 4.2.2 (SIP-dG of Poisson equation [Arn82]) *Define the discretised bilinear form $\mathscr{A}_h(\cdot,\cdot)$ for the Poisson problem by:*

$$\mathscr{A}_h(u_h, v_h) := \int_\Omega \nabla u_h \cdot \nabla v_h \,\mathrm{d}\boldsymbol{x} - \int_{\mathscr{E}_I \cup \mathscr{E}_D} \left([\![u_h]\!] \cdot \{\!\{ \nabla v_h \}\!\} + [\![v_h]\!] \cdot \{\!\{ \nabla u_h \}\!\} - \sigma h^{-1} [\![u_h]\!] \cdot [\![v_h]\!] \right) \mathrm{d}s \tag{4.2.14}$$

and the corresponding linear functional l_h by

$$l_h(v) := \int_\Omega f v_h \,\mathrm{d}\boldsymbol{x} + \int_{\mathscr{E}_N} g v \,\mathrm{d}s + \int_{\mathscr{E}_D} \left(v_h \sigma h^{-1} - \nabla v_h \cdot \boldsymbol{n}_\Omega \right) u_D \,\mathrm{d}s, \tag{4.2.15}$$

where $\sigma > 0$ is the discontinuity penalisation parameter used to ensure coercivity of the bilinear form.

The SIP-dG method of the Poisson problem (4.2.12) then reads as:

$$\begin{aligned} &\text{Find } u_h \in \mathbb{V} \text{ such that:} \\ &\mathscr{A}_h(u_h, v_h) = l_h(v_h) \quad \forall v_h \in \mathbb{V}. \end{aligned} \tag{4.2.16}$$

Some basic properties of the SIP-dG method, such as existence and uniqueness of the result, have been proved in [Wih03]. In particular, we are interested in following Proposition:

Proposition 4.5 (Consistency of the SIP-dG method [Wih03, Proposition 2.3.2]) *Let Ω be a polygon and $\beta_{min} \leq \beta < 1$ a weight vector. Then, for $f \in \mathrm{H}^{0,0}_{\boldsymbol{\beta}}(\Omega), u_D \in \mathrm{H}^{3/2,3/2}_{\boldsymbol{\beta}}(\Gamma_D)$, the bilinear form and the linear functional in definition 4.2.2 are well-defined and the SIP-dG method is consistent:*

$$\mathscr{A}_h(u, v_h) - l_h(v_h) = 0 \quad \forall v_h \in \mathbb{V}, \tag{4.2.17}$$

where $u \in \mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega)$ is the exact solution of (4.2.12).

The proof is based on the fact that $u \in \mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega) \subset \mathscr{C}^0(\overline{\Omega})$, $\nabla u \in \mathrm{H}^{1,1}_{\boldsymbol{\beta}}(\Omega)^2$ by 4.1 and Lemma 4.2.

Remark 4.6 *The SIP-dG method given in Def.4.2.2 is only consistent for $u \in \mathrm{H}^{3/2+\varepsilon}(\Omega)$, since we require $\{\!\{ \nabla u \}\!\}$ to be well-defined in Proposition 4.5. Since $u \in \mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega)$, we assume that $\mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega) \subset \mathrm{H}^{3/2+\varepsilon}(\Omega)$.*

4.3 A-posteriori estimates for the SIP-dG method

In this section, we derive a *novel* a-posteriori bound for the solution of the SIP-dG method in the L^2 norm. Previous works have addressed a similar task. In particular, the authors in [MW14] derive a-posteriori error estimates in the energy norm, while in [ARW07] the authors treat an optimal control problem for a 2D elliptic equation with pointwise control constraints. The domain is assumed to be polygonal but non-convex and corner singularities are treated by a priori mesh grading. Our proposed formulation in the L^2 norm does not involve any weight Φ_β, resulting in an useful expression for testing the validity of our numerical tests. At the end of this section, we will also introduce an upper bound of the error in the L^∞ norm [DG12] for further comparison between the h- and r-adaptive methods. The L^2 and L^∞ a-posteriori estimates are given by (4.3.14) and (4.3.15).

The h-adaptive strategy described in the subsequent numerical experiments uses these estimators in an iterated fashion. Under this strategy, cells which have large a-posteriori errors are identified and then refined. The solution is then recalculated until in all the cells the a-posteriori estimator is almost the same. This leads to an adapted mesh which approximately equidistributes the a-posteriori error estimators.

We will also give numerical evidence that these same bounds are automatically equidistributed when using the optimal OT based r-adaptive strategy.

We now state element-wise trace inequalities and interpolation error bounds based on the local regularity of the solution:

Lemma 4.7 (Trace inequalities [Sch98, Lemma 4.55]) *Let $K \in \mathscr{T}$ and denote by $e \subseteq \partial K$ the boundary or some sides of K. Further we define the space*

$$\mathrm{H}^k(e) = \prod_{i=1}^{3} \mathrm{H}^k(e_i),\ k = 0, 1,$$

where e_i are the sides of K and assume that one vertex of K coincides with the origin. We have

1. *If $u \in \mathrm{H}^2(K)$ and $u = 0$ at the vertices of K, then $u \in \mathrm{H}^1(e)$ and*

$$\begin{aligned} \|u\|^2_{\mathrm{H}^1(e)} &\leq Ch_K |u|^2_{\mathrm{H}^2(K)}, \\ \|u\|^2_{\mathrm{L}^2(e)} &\leq Ch_K^3 |u|^2_{\mathrm{H}^2(K)}. \end{aligned} \tag{4.3.1}$$

2. *If $u \in \mathrm{H}^{2,2}_{\beta}(K)$, $0 < \beta < 1$, and $u = 0$ at the vertices of K, then*

$$\begin{aligned} \|u\|^2_{\mathrm{H}^1(e)} &\leq Ch_K^{1-2\beta} |u|^2_{\mathrm{H}^{2,2}_{\beta}(K)}, \\ \|u\|^2_{\mathrm{L}^2(e)} &\leq Ch_K^{3-2\beta} |u|^2_{\mathrm{H}^{2,2}_{\beta}(K)}. \end{aligned} \tag{4.3.2}$$

We now state the following Lemma used to bound linear interpolants:

Lemma 4.8 (Errors for linear interpolant Π_K [Wih03, Lemma 2.5.2]) *Let K be a triangle with vertices A_1, A_2, A_3. Further, Let $v \in \mathrm{H}^{2,2}_{\beta_K}(K)$, with $\Phi_{\beta_K}(\boldsymbol{x}) = |\boldsymbol{x} - A_1|^{\beta_K}$, $0 < \beta_K < 1$. In corner elements $K \in \mathscr{K}_0$, the linear interpolant of v in the vertices of K, denoted by $\Pi_K v$ satisfies*

$$\begin{aligned} \|v - \Pi_K v\|_{\mathrm{L}^2(K)} &\leq Ch_K^{2-\beta_K} |v|_{\mathrm{H}^{2,2}_{\beta_k}(K)}, \\ \|v - \Pi_K v\|_{\mathrm{H}^1(K)} &\leq Ch_K^{1-\beta_K} |v|_{\mathrm{H}^{2,2}_{\beta_k}(K)}, \\ \|v - \Pi_K v\|_{\mathrm{H}^{2,2}_{\beta_K}(K)} &\leq C |v|_{\mathrm{H}^{2,2}_{\beta_k}(K)}. \end{aligned} \tag{4.3.3}$$

The above results holds also for $\beta_K = 0$, i.e. $v \in \mathrm{H}^2(K)$.

4.3.1 Derivation of the L^2 a-posteriori error estimate

We can now proceed with the derivation of the a-posteriori estimate. We will employ the dual formulation of the original problem.

Theorem 4.9 (Dual formulation [BG88, Theorem 2.1]) *There exists $v \in \mathrm{H}^{2,2}_{\beta}(\Omega)$, where β depends on the opening angles at the vertices of Ω, such that*

$$\begin{aligned} -\Delta z &= u - u_h \ \ \textit{in } \Omega, \\ z &= 0 \ \ \textit{on } \Gamma_D, \\ \nabla z \cdot \boldsymbol{n}_\Omega &= 0 \ \ \textit{on } \Gamma_N, \end{aligned} \tag{4.3.4}$$

and

$$\|z\|_{\mathrm{H}^{2,2}_{\beta}(\Omega)} \leq C \, \|u - u_h\|_{\mathrm{L}^2(\Omega)} \, . \tag{4.3.5}$$

Theorem 4.9 and the Galerkin orthogonality resulting from Proposition 4.5 leads to:

$$\begin{aligned} \mathscr{A}_h(\phi, z) &= \langle u - u_h, \phi \rangle \quad \textit{Dual formulation}, \\ \mathscr{A}_h(u - u_h, z_h) &= 0 \quad \textit{Galerkin orthogonality}, \\ \|u - u_h\|^2 &= \mathscr{A}_h(u - u_h, z) = \mathscr{A}_h(u - u_h, z - z_h). \end{aligned} \tag{4.3.6}$$

Lemma 4.10 *The L^2 a-posteriori estimator for Poisson's equation* (4.2.12) *on a non-convex domain with one re-entrant corner using the SIP-dG method is*

$$\begin{aligned} \|u - u_h\|_{\mathrm{L}^2(\Omega)} = \Big[& C_1 \Big(\sum_{K \in \mathscr{T}/\mathscr{K}_0} h_K^4 \, \|f + \Delta u_h\|^2_{\mathrm{L}^2(K)} + \sum_{K \in \mathscr{K}_0} h_K^{4-2\beta} \, \|f + \Delta u_h\|^2_{\mathrm{L}^2(K)} \Big) \\ & + C_2 \Big(\sum_{e \in \mathscr{E}_I/\mathscr{E}_0} h_{K_e}^3 \, \|[\![\nabla u_h]\!]\|^2_{\mathrm{L}^2(e)} + \sum_{e \in \mathscr{E}_0} h_{K_e}^{3-2\beta} \, \|[\![\nabla u_h]\!]\|^2_{\mathrm{L}^2(e)} \Big) \\ & + C_3 \Big(\sum_{e \in \mathscr{E}_I/\mathscr{E}_0} h_{K_e} \, \|[\![u_h]\!]\|^2_{\mathrm{L}^2(e)} + \sum_{e \in \mathscr{E}_0} h_{K_e}^{1-2\beta} \, \|[\![u_h]\!]\|^2_{\mathrm{L}^2(e)} \Big) \Big]^{1/2} \end{aligned} \tag{4.3.7}$$

where C_i, $i = 1, \ldots, 3$ are positive constants depending on the mesh and shape regularity, h_K, h_{K_e} is the cell size of element K, with K_e having an edge e, and β is the parameter of the weighted Sobolev space depending on the width of the re-entrant corner.

1 Proof: We use eq.(4.3.6) to bound the error in the L^2 norm through the bilinear form $\mathscr{A}_h(z - z_h, u - u_h)$:

$$\begin{aligned}
\mathscr{A}_h(z - z_h, u - u_h) &= \int_\Omega \nabla(z - z_h) \cdot \nabla(u - u_h)\,\mathrm{d}\boldsymbol{x} - \int_{\mathscr{E}_D \cup \mathscr{E}_I} \Big([\![z - z_h]\!] \{\!\{ \nabla(u - u_h) \}\!\} + [\![u - u_h]\!] \{\!\{ \nabla(z - z_h) \}\!\} \\
&\quad - \sigma h^{-1} [\![z - z_h]\!] [\![u - u_h]\!] \Big)\,\mathrm{d}s \\
&= \int_{\mathscr{E}_I} \Big([\![\nabla(u - u_h)]\!] \{\!\{ z - z_h \}\!\} + \{\!\{ \nabla(u - u_h) \}\!\} [\![z - z_h]\!] \Big)\,\mathrm{d}s + \int_\Omega (z - z_h)(f + \Delta u_h)\,\mathrm{d}\boldsymbol{x} \\
&\quad - \int_{\mathscr{E}_I} \Big([\![z - z_h]\!] \{\!\{ \nabla(u - u_h) \}\!\} + [\![u - u_h]\!] \{\!\{ \nabla(z - z_h) \}\!\} - \sigma h^{-1} [\![z - z_h]\!] [\![u - u_h]\!] \Big)\,\mathrm{d}s \\
&= \int_\Omega (z - z_h)(f + \Delta u_h)\,\mathrm{d}\boldsymbol{x} + \int_{\mathscr{E}_I} \Big([\![\nabla(u - u_h)]\!] \{\!\{ z - z_h \}\!\} - [\![u - u_h]\!] \{\!\{ \nabla(z - z_h) \}\!\} \\
&\quad + \sigma h^{-1} [\![z - z_h]\!] [\![u - u_h]\!] \Big)\,\mathrm{d}s \\
&=: \mathcal{R}_1 + \mathcal{R}_2 + \mathcal{R}_3 + \mathcal{R}_4.
\end{aligned} \tag{4.3.8}$$

For the first term we start applying the Cauchy-Schwarz inequality:

$$\begin{aligned}
\mathcal{R}_1 &= \int_\Omega (z - z_h)(f + \Delta u_h)\,\mathrm{d}\boldsymbol{x} \\
&\le \sum_{K \in \mathscr{T}/\mathscr{K}_0} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} \|z - z_h\|_{\mathrm{L}^2(K)} + \sum_{K \in \mathscr{K}_0} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} \|z - z_h\|_{\mathrm{L}^2(K)} \\
&\le C_1^{(1)} \sum_{K \in \mathscr{T}/\mathscr{K}_0} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} h_K^2 |z|_{\mathrm{H}^2(K)} + C_1^{(2)} \sum_{K \in \mathscr{K}_0} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} h^{2-\beta} |z|_{\mathrm{H}^{2,2}_\beta(K)} \\
&\le C_1 \|z\|_{\mathrm{H}^{2,2}_\beta(\Omega)} \Big(\sum_{K \in \mathscr{T}/\mathscr{K}_0} h_K^2 \|f + \Delta u_h\|_{\mathrm{L}^2(K)} + \sum_{K \in \mathscr{K}_0} h_K^{2-\beta} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} \Big) \\
&\le C_1 \|u - u_h\|_{\mathrm{L}^2(\Omega)} \Big(\sum_{K \in \mathscr{T}/\mathscr{K}_0} h_K^2 \|f + \Delta u_h\|_{\mathrm{L}^2(K)} + \sum_{K \in \mathscr{K}_0} h_K^{2-\beta} \|f + \Delta u_h\|_{\mathrm{L}^2(K)} \Big),
\end{aligned} \tag{4.3.9}$$

where $z_h = \Pi_K z$, the linear interpolant of z in the vertices of K. For $K \in \mathscr{T}/\mathscr{K}_0$ and $K \in \mathscr{K}_0$ we used Lemma 4.8. The last inequality is then given by eq.(4.3.5) in Theorem 4.9.

For the second term we obtain

$$
\begin{aligned}
\mathcal{R}_2 &= \sum_{e\in\mathscr{E}_I/\mathscr{E}_0}\int_e [\![\nabla(u-u_h)]\!]\,\{\!\{\, z-z_h \,\}\!\} + \sum_{e\in\mathscr{E}_0}\int_e [\![\nabla(u-u_h)]\!]\,\{\!\{\, z-z_h \,\}\!\}\,\mathrm{d}s \\
&\le C_2^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)}\, \|\{\!\{\, z-z_h \,\}\!\}\|_{\mathrm{L}^2(e)} + C_2^{(2)} \sum_{e\in\mathscr{E}_0} \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)}\, \|\{\!\{\, z-z_h \,\}\!\}\|_{\mathrm{L}^2(e)} \\
&\le C_2^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_{K_e}^{3/2}\, |z-z_h|_{\mathrm{H}^2(K_e)}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)} + C_2^{(2)} \sum_{e\in\mathscr{E}_0} h_{K_e}^{3/2-\beta}\, |z-z_h|_{\mathrm{H}^{2,2}_{\beta_K}(K_e)}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(K)} \\
&\le C_2^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_{K_e}^{3/2}\, |z|_{\mathrm{H}^2(K_e)}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)} + C_2^{(2)} \sum_{e\in\mathscr{E}_0} h_{K_e}^{3/2-\beta}\, |z|_{\mathrm{H}^{2,2}_{\beta}(K_e)}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)} \\
&\le C_2\, \|u-u_h\|_{\mathrm{L}^2(\Omega)} \Big(\sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_{K_e}^{3/2}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)} + \sum_{e\in\mathscr{E}_0} h_{K_e}^{3/2-\beta}\, \|[\![\nabla u_h]\!]\|_{\mathrm{L}^2(e)} \Big),
\end{aligned}
\tag{4.3.10}
$$

where we have used Remark 4.1 and Lemma 4.2 for $\nabla u \in \mathrm{H}^{1,1}_{\beta}(\Omega)^2$, and Lemma 4.8 for $|z-z_h|_{\mathrm{H}^2(K_e)}$ and $|z-z_h|_{\mathrm{H}^{2,2}_{\beta_K}(K_e)}$. Note that for any edge $e := K_1 \cap K_2 \in \mathscr{E}$, with elements $K_1, K_2 \in \mathscr{T}$, the quantity $K_e := K_1 \cup K_2$ is well defined as the mesh is conforming.

For the third term we apply again the Cauchy-Schwarz and the trace inequality to obtain

$$
\begin{aligned}
\mathcal{R}_3 &= -\sum_{e\in\mathscr{E}_I/\mathscr{E}_0}\int_e [\![u-u_h]\!]\,\{\!\{\, \nabla(z-z_h) \,\}\!\} - \sum_{\gamma\in\mathscr{E}_0}\int_e [\![u-u_h]\!]\,\{\!\{\, \nabla(z-z_h) \,\}\!\} \\
&\le C_3^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} \|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\, \|\{\!\{\, \nabla(z-z_h) \,\}\!\}\|_{\mathrm{L}^2(e)} + C_3^{(2)} \sum_{e\in\mathscr{E}_0} \|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\, \|\{\!\{\, \nabla(z-z_h) \,\}\!\}\|_{\mathrm{L}^2(e)} \\
&\le C_3^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} \|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\, h_{K_e}^{1/2}\, |z|_{\mathrm{H}^2(K_e)} + C_3^{(2)} \sum_{e\in\mathscr{E}_0} \|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\, h_{K_e}^{1/2-\beta}\, |z|_{\mathrm{H}^{2,2}_{\beta}(K_e)} \\
&\le C_3\, \|u-u_h\|_{\mathrm{L}^2(\Omega)} \Big(\sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_{K_e}^{1/2}\, \|[\![u_h]\!]\|_{\mathrm{L}^2(e)} + \sum_{e\in\mathscr{E}_0} h_{K_e}^{1/2-\beta}\, \|[\![u_h]\!]\|_{\mathrm{L}^2(e)} \Big),
\end{aligned}
\tag{4.3.11}
$$

where we have used the property $\mathrm{H}^{2,2}_{\beta} \subset \mathrm{H}^{1,1}_{\beta}$ and Lemma 4.2 to cancel the term $[\![u]\!]$.

For the fourth term we have

$$
\begin{aligned}
\mathcal{R}_4 &= \sum_{e\in\mathscr{E}_I/\mathscr{E}_0}\int_e \sigma h_e^{-1}\,[\![u-u_h]\!]\,[\![z-z_h]\!]\,\mathrm{d}s + \sum_{e\in\mathscr{E}_0}\int_e \sigma h_e^{-1}\,[\![u-u_h]\!]\,[\![z-z_h]\!]\,\mathrm{d}s \\
&\le C_4^{(1)} \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} \sigma h_e^{-1}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\,\|[\![z-z_h]\!]\|_{\mathrm{L}^2(e)} + C_4^{(2)} \sum_{e\in\mathscr{E}_0} \sigma h_e^{-1}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\,\|[\![z-z_h]\!]\|_{\mathrm{L}^2(e)} \\
&\le C_4^{(1)}\sigma \sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_e^{-1}h_{K_e}^{3/2}\,\|z-z_h\|_{\mathrm{H}^2(K_e)}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)} + C_4^{(2)}\sigma \sum_{e\in\mathscr{E}_0} h_e^{-1}h_{K_e}^{3/2-\beta}\,\|z-z_h\|_{\mathrm{H}^{2,2}_\beta(K_e)}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)} \\
&\le C_4\sigma\,\|u-u_h\|_{\mathrm{L}^2(\Omega)}\Big(\sum_{e\in\mathscr{E}_I/\mathscr{E}_0} h_{K_e}^{1/2}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)} + \sum_{e\in\mathscr{E}_0} h_{K_e}^{1/2-\beta}\,\|[\![u_h]\!]\|_{\mathrm{L}^2(e)}\Big).
\end{aligned}
\tag{4.3.12}
$$

Summing up the terms (4.3.9)–(4.3.12), we divide by $\|u-u_h\|_{\mathrm{L}^2(\Omega)}$ and apply the arithmetic-mean geometric-mean inequality [HR11, Theorem B.0.11] to obtain the final result. □

Corollary 4.11 *The expression derived in Lemma 4.10 can be simplified if $u_h \in \mathbb{V}$, with $p = 1$ and $f = 0$. We further introduce the indicator function $\mathbb{1}_{e\in\mathscr{E}_0}$ and rewrite* (4.3.7) *in the more compact form*

$$
\|u-u_h\|_{\mathrm{L}^2(\Omega)} \le \Big(\widehat{C_1}\sum_{e\in\mathscr{E}_I} h_{K_e}^{3-2\mathbb{1}_{e\in\mathscr{E}_0}\beta}\,\|[\![\nabla u_h]\!]\|^2_{\mathrm{L}^2(e)} + \widehat{C_2}\sum_{e\in\mathscr{E}_I} h_{K_e}^{1-2\mathbb{1}_{e\in\mathscr{E}_0}\beta}\,\|[\![u_h]\!]\|^2_{\mathrm{L}^2(e)}\Big)^{1/2}, \tag{4.3.13}
$$

where the term with constant $\widehat{C_2}$ is the sum of the terms with constant C_3 in eq.(4.3.7)*, as they all depend on $[\![u_h]\!]$ with the same power of h_{K_e}.*

For practical purposes, we reformulate (4.3.13) *as a sum of cell-wise L^2 error estimates:*

$$
\begin{aligned}
\|u-u_h\|^2_{\mathrm{L}^2(\Omega)} &\le \sum_{K\in\mathscr{T}}\Big(\widehat{C}_1\sum_{e\in\mathscr{E}_I(K)} h_{K_e}^{3-2\mathbb{1}_{e\in\mathscr{E}_0}\hat\beta}\,\|[\![\nabla_h u_h]\!]\|^2_{\mathrm{L}^2(e)} + \widehat{C}_2\sum_{e\in\mathscr{E}_I(K)} h_{K_e}^{1-2\mathbb{1}_{e\in\mathscr{E}_0}\hat\beta}\,\|[\![u_h]\!]\|^2_{\mathrm{L}^2(e)}\Big) \\
&=: \sum_{K\in\mathscr{T}} \eta^2_{\mathrm{L}^2,K},
\end{aligned}
\tag{4.3.14}
$$

where $\hat\beta \in [0,1]$ replaces the weight β (Remark 4.4). The term $\hat\beta$ is included in the upper bound for the L^2 error but does not prevent the solution u of problem 4.2.12 to lie in $\mathrm{H}^{2,2}_\beta$ with $\beta > \hat\beta$.

We also present an L^∞ a-posteriori estimator for the SIP-dG method [DG12, equation 1.2] by assuming that $p = 1$ and $f = 0$ as in (4.3.13). The domain Ω is not required to be convex, thus the following bound is appropriate for our numerical calculations:

$$\|u - u_h\|_{\mathrm{L}^\infty(\Omega)} \leq C\, l_{h,d}\Big(\|h\, [\![\nabla u_h]\!]\|_{\mathrm{L}^\infty(\mathscr{E}_I)} + \|[\![u_h]\!]\|_{\mathrm{L}^\infty(\mathscr{E}_I)} \Big) =: \eta_{\mathrm{L}^\infty,K}, \tag{4.3.15}$$

where $l_{h,d} = \big(\ln(1/\underline{h})\big)^2$.

We conclude this section by introducing the global error indicator $\eta^2_{\mathrm{L}^2} := \sum_{K\in\mathscr{T}} \eta^2_{\mathrm{L}^2,K}$ and $\eta_{\mathrm{L}^\infty} := \max_{K\in\mathscr{T}} \eta_{\mathrm{L}^\infty,K}$, where $\eta^2_{\mathrm{L}^2,K}$ and $\eta_{\mathrm{L}^\infty,K}$ refers to each cell K in equation (4.3.14) and (4.3.15), respectively.

4.4 r-adapted meshes

In this section we implement two r-adaptive strategies: Winslow's diffusion method and an OT based strategy. For the former, we propose different monitor functions and test the accuracy of the SIP-dG method over the adapted meshes. We then describe our semi-analytical approach to construct an OT mesh by exploiting the behaviour of the solution near the re-entrant corner. This results in a non-algebraic equation that maps a uniform computational mesh to the OT mesh. The implementation of the algorithm depends on the meta-parameter γ. The optimal value that minimises the local interpolation error will be computed analytically and numerical tests will corroborate the validity of the results.

4.4.1 Winslow's diffusion method

The Winslow's diffusion method is a moving mesh PDE (MMPDE), defined in §1 of Chapter 2. We consider the functional

$$I_{Win}[\boldsymbol{\xi}] = \frac{1}{2}\int_\Omega \frac{1}{\rho(\boldsymbol{x})} \sum_i (\nabla\xi_i)^T\,(\nabla\xi_i)\,\mathrm{d}\boldsymbol{x}, \tag{4.4.1}$$

where $\rho(\boldsymbol{x}) > 0$ is a given monitor function. This function can be chosen to depend on the discrete solution u_h of the physical PDE. We consider as possible choices:

1. Gradient:

$$\rho|_K = \left(1 + \frac{1}{\delta}\,|\nabla_h u_h|^2\right)^{1/2}. \tag{4.4.2}$$

2. Curvature:

$$\rho|_K = \left(1 + \frac{1}{\delta}\,|\Delta_h u_h|\right)^{1/2}. \tag{4.4.3}$$

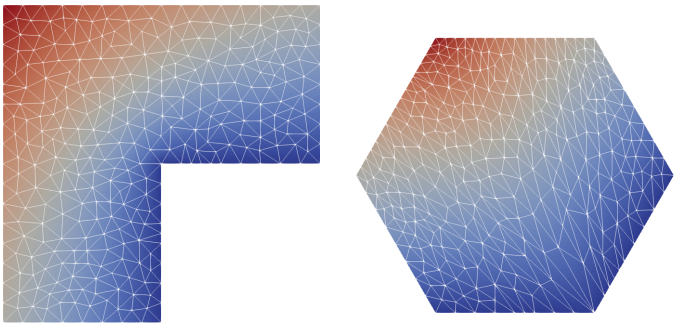
3. A-posteriori:

$$\rho|_K = \left(1 + \frac{1}{\delta}\eta^2_{\mathrm{L}^2,K}\right)^{1/2}. \tag{4.4.4}$$

where δ is a prescribed intensity parameter for $\mathcal{T}$ [HR11; BHR09]. The MMPDE associated with the functional (4.4.1) has been derived and discretised in Chapter 3. Dirichlet boundary conditions are imposed on all the sides of Ω except for that ones having the re-entrant corner as extremum. For those, the monitor function $\rho(\boldsymbol{\xi})$ has been first projected onto the two sides and the solution of the MMPDE5 [HR11] in one dimension is computed. This results in a effective movement of the boundary mesh points towards the origin.

The non-convex domain Ω poses a practical complication for the application of Winslow's MMPDE. In fact, it is well known that if the computational domain Ω_c is not convex, the resulting physical mesh might feature very skewed or even overlapping elements. A natural solution for that issue is to define a computational mesh that is convex and solve for the coordinate transformation $\boldsymbol{x}(\boldsymbol{\xi})$ on this domain [CHR99; LTZ01]. Once Ω_c is selected, a mesh $\mathcal{T}_c$ with the same topology as $\mathcal{T}$ can be constructed by first specifying a correspondence between the boundaries $\partial\Omega$ and $\partial\Omega_c$ by a mapping $g(\boldsymbol{x})$, and then let $\mathcal{T}_c$ be the image of $\mathcal{T}$ under the mapping $\boldsymbol{\xi}(\boldsymbol{x})$ satisfying

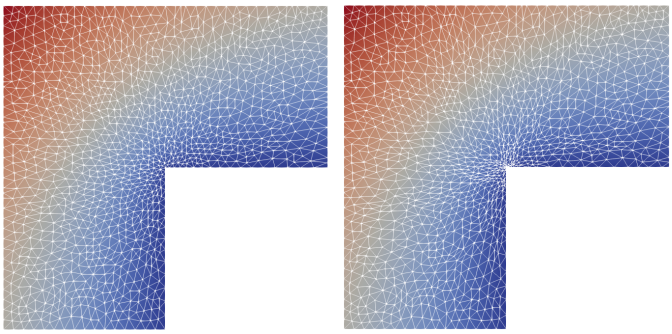
$$\begin{aligned} \Delta\boldsymbol{\xi}(\boldsymbol{x}) &= 0 \ \text{ in } \ \Omega, \\ \boldsymbol{\xi} &= g(\boldsymbol{x}) \ \text{ on } \ \partial\Omega. \end{aligned} \tag{4.4.5}$$



(a) L-shaped domain. (b) Computational domain.

Figure 4-3: Solution of equation (4.5.1) represented in the physical and computational domain.

By mapping the L-shaped domain to a hexagon, it is possible to solve Winslow's MMPDE without incurring the risk of tangling elements. We remind that also the choice of δ heavily affects the quality of the final mesh. In particular, for $\delta \gg 1$, the mesh does not adapt to the target function. On the other hand, $\delta \ll 1$ yields an adapted mesh, but the mesh elements might result too skewed.



(a) Gradient ρ. (b) Curvature ρ.

(c) A-posteriori ρ ($\hat{\beta} = 0.0$). (d) A-posteriori ρ ($\hat{\beta} = 0.99$).

Figure 4-4: Solution of eq.(4.5.1) on the mesh given by the Winslow's MMPDE ($\dim \mathbb{V} = 7005$) with monitor functions ρ in eq.(4.4.2)–(4.4.4).

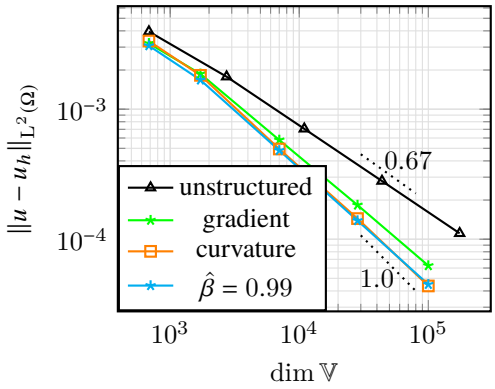
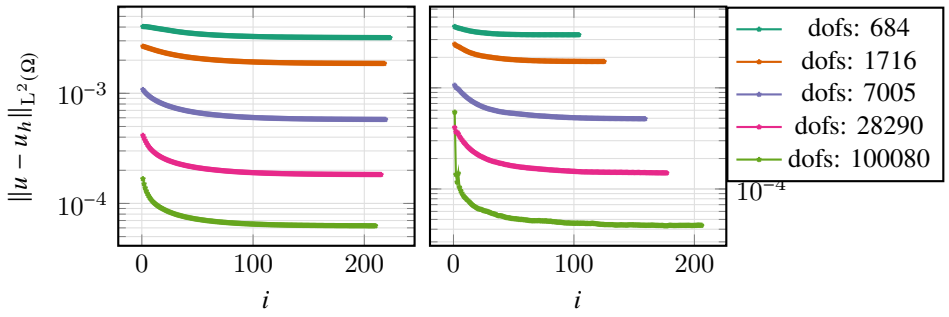


Figure 4-5: Convergence rates for Winslow's MMPDE with monitor functions ρ in eq.(4.4.2)–(4.4.4).

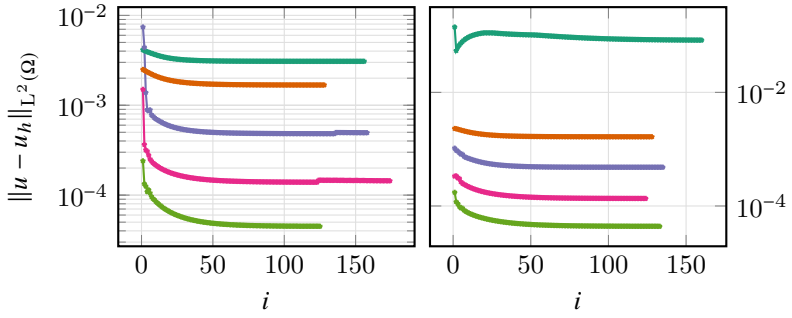
Figure 4-3 shows the solution of Poisson's equation on the L-shaped domain and the mapping to an hexagonal computational domain using equation (4.4.5). Given the convex nature of the computational domain, the Winslow's MMPDE becomes stable and converges to a solution for all the monitor functions in (4.4.2)–(4.4.4), as visible in Figure 4-6.

The monitor functions in (4.4.2)–(4.4.4) generate different meshes. In Figure 4-4, we notice that all



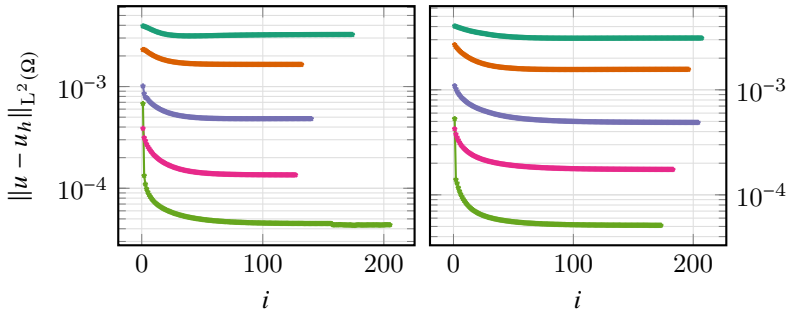
(a) Gradient ρ.

(b) Curvature ρ.



(c) A-posteriori ρ ($\hat{\beta} = 0.0$).

(d) A-posteriori ρ ($\hat{\beta} = 0.33$).



(e) A-posteriori ρ ($\hat{\beta} = 0.5$).

(f) A-posteriori ρ ($\hat{\beta} = 0.99$).

Figure 4-6: The solution error decreases monotonically over the iterations until reaching convergence for all the monitor functions. The relative tolerance has been fixed to 1×10^{-5}, while the timestep has been set to $k = 10^{-3}$.

the meshes have elements compressed towards the re-entrant corner. In particular, the gradient and the a-posteriori ($\hat{\beta} = 0$) monitor functions yield a mesh with few skewed elements (Figure 4-4a and 4-4c). In contrast, the curvature and the a-posteriori ($\hat{\beta} = 0.99$) monitor functions produce anisotropic meshes with more skewed elements (Figure 4-4b and 4-4d). The higher level of adaptation for the latter monitor functions explains the optimal order of convergence showed in Figure 4-5.

4.4.2 Mesh generation using an OT strategy

The Optimal Transport (OT) is a r-adaptive strategy that takes a different approach than the h-adaptive method, as discussed in the introduction of Chapter 2. This process can be interpreted as taking a computational domain Ω_c, with a fixed mesh $\mathscr{T}_c$, and then mapping this with a diffeomorphic map $\mathbf{F}$ to a physical domain Ω. The physical mesh $\mathscr{T}$ is then image of $\mathscr{T}_c$ under the action of $\mathbf{F}$. The properties of the mesh (such as the mesh regularity and the mesh grading) can then be found from looking at the nature of the map. In the OT strategy this map is the gradient $\mathbf{x} = \nabla\phi$ of a (mesh) potential, ϕ which (in general) is the solution of a Monge-Ampère equation. The main assumption is that there is a *monitor function* $\rho(\mathbf{x})$ of the solution at the point $\mathbf{x}$ in the physical space, which is large where we need to cluster the mesh points. The OT method then constructs a mesh by equidistributing the integral of ρ over the mesh cells, whilst minimising the Wasserstein distance between the mesh and a uniform mesh. This approach gives both good mesh scaling and good mesh regularity [BRW15; BHR09; Wel+16]. The mesh potential satisfies the Monge-Ampère equation in the form

$$\rho(\nabla\phi)D^2\phi = C, \tag{4.4.6}$$

where $D^2\phi$ is the Hessian of ϕ and C is a constant. In this section we consider a function $\rho(\mathbf{x})$ which monitors the a-priori *linear cell interpolation error* of the solution in L^∞ or in L^2 norm. This is a proxy for the FE error estimated in the previous sections, but we will show that it leads to good error estimates.

In general the fully nonlinear PDE (4.4.6) is hard to solve for a general domain, however, by making certain observations about the nature of the re-entrant corner problem, we can make some significant simplifications, which allows us to find an approximate solution for ϕ. To do this we focus on the area close to the re-entrant corner, which is where the mesh needs to be refined. We will assume that both the computational and the physical domain have the same re-entrant geometry, and will consider a map from one to the other which concentrates the mesh points close to the corner of the physical domain.

To do this we will take the origin $\mathbf{0}$ at the re-entrant corner, and take polar coordinates (s, ψ) and (r, θ) in the respective domains. We will also assume that $\psi = \theta = 0$ on one edge of the corner and $\psi = \theta = 2\pi/\omega$ on the other. Now we consider the local form of the solution $u(r, \theta)$. At the corner this has the leading order expression $u(r, \theta) \sim r^\delta f(\theta)$ where the function f is *slowly varying*. As $\delta < 1$ in the problems of interest, we have that the partial derivatives u_r and u_{rr} are all singular as $r \to 0$, whereas the partial derivatives of u with respect to θ are all bounded in this limit. The large values of the partial derivatives with respect to r lead to large interpolation and FE errors. In other words, the difficulty in representing u is due to its variation in r rather than in θ. Accordingly it makes sense to consider a mesh which concentrates points in the r-direction, but not in the θ-direction. We then consider maps $\mathbf{F}$ such that $\mathbf{F}(s, \theta) = (r(s), \theta)$. Note that this map trivially preserves the geometry of the re-entrant corner. Following the framework described above, we can consider a radially symmetric mesh potential

function $\phi(s)$ so that $r = \mathrm{d}\phi/\,\mathrm{d}s$ and $\rho(\mathbf{x}) \equiv \rho(r)$. It follows from the reasoning in [BW10] that the two-dimensional Monge-Ampère equation (4.4.6) takes the simpler form

$$\rho(\phi_s)\phi_{ss}r = Cs \tag{4.4.7}$$

or more simply

$$\rho(r)r\frac{dr}{ds} = Cs. \tag{4.4.8}$$

We now consider how the solution of this equation generates a mesh. Suppose that there is a uniform mesh in the computational domain of constant mesh width (cell diameter) Δ. Provided that Δ is small, then the corresponding mesh width h in the physical domain is given by

$$h = \frac{\mathrm{d}r}{\mathrm{d}s}\Delta. \tag{4.4.9}$$

We will now consider solutions of (4.4.8) for various forms of the monitor function $\rho(r)$.

4.4.3 Local mesh scaling

Consider the general solution of equation (4.2.12). The local (linear) interpolation error E on a small cell K of width h in various norms is bounded by

$$E_K < K_1 h^{\delta} \left\| u^{(\beta)} \right\|_{\mathrm{L}^p(K)}. \tag{4.4.10}$$

Here δ and β depend upon the order p being used, although $\beta = 2$ is typical for the norms and problems we consider. Using the expression $u \sim r^{\alpha} f(\theta)$, where f is a regular function of θ, and again noting that as $r \to 0$ the partial derivatives of u with respect to r dominate those with respect to θ, it follows that for small r

$$E_K < K_2 h^{\delta} r^{\alpha-\beta}.$$

In the OT formulation, we aim to *equidistribute* the estimate E over each cell. It follows that there is a constant K_3, and an exponent γ (which is a meta-parameter which depends on the norm), such that

$$h = K_3 r^{\gamma}. \tag{4.4.11}$$

Lemma 4.12 *Let h be given by* (4.4.11)*, then this is equivalent to taking a monitor function $\rho(r)$ given by:*

$$\rho(r) = Br^{-2\gamma} \tag{4.4.12}$$

for some constant B.

1 Proof: From (4.4.11) we have $h \sim dr/ds = r^\gamma$. Hence $s \sim r^{1-\gamma}$. From the Monge-Ampère equation (4.4.8) we then have, for some constant B

$$\rho(r) r r^\gamma = s = B r^{1-\gamma} \quad \text{so that} \quad \rho(r) = B r^{-2\gamma}.$$

□

The choice of γ required to equidistribute the interpolation error E is dependent upon the norm used to calculate the error. In particular, we have the following result:

Lemma 4.13 *The value of γ required to equidistribute the interpolation error is:*

$$\gamma = 1 - \frac{\pi}{2\omega} \quad \textit{for the} \quad \mathrm{L}^\infty \quad \textit{norm}, \quad \gamma = \frac{2}{3} - \frac{\pi}{3\omega} \quad \textit{for the} \quad \mathrm{L}^2 \quad \textit{norm}.$$

2 Proof: See Appendix A. □

We remark that this strategy for finding γ aims to equidistribute the two-dimensional interpolation error. However, this is likely to be sub-optimal for the SIP-dG framework, which seeks instead to minimise the local FE error. However, we will show in the numerical results of the next section, that the estimate of γ above is very close to being optimal for the FE error.

4.4.4 Solution of Monge-Ampère equation

The monitor function in (4.4.12) gives excellent mesh compression close to the corner when r is small. However we also seek a mesh which becomes asymptotically regular as r becomes larger, so that it can be matched to a near uniform computational mesh away from the re-entrant corner. This is equivalent to having a monitor function which is close to a constant A for large values of r. A monitor function $\rho(r)$ which satisfies both conditions is given simply by:

$$\rho(r) = A + B r^{-2\gamma}. \tag{4.4.13}$$

To find the mesh over all ranges of r we then substitute eq.(4.4.13) into eq.(4.4.8) to give

$$\left(A + B r^{-2\gamma}\right) r \frac{\mathrm{d}r}{\mathrm{d}s} = s.$$

Integrating this expression we find that r satisfies the nonlinear algebraic equation

$$Ar^2 + \frac{B}{1-\gamma} r^{2(1-\gamma)} = s^2. \tag{4.4.14}$$

The equation (4.4.14) determines the mesh transformation from s to r (noting that θ is mapped to θ). Note that the parameter A and B controls the level of mesh compression near the corner. In fact, for high values of B and small r we have $r \sim s^{1/(1-\gamma)}$ and for large A and r we have $r \sim s$. For the remainder of the analysis, we choose $B = (1-\gamma)$, as this is the maximum admissible value that respects the boundary condition $A + B/(1-\gamma) = 1$ when $r, s = 1$. The nature of the mesh then depends on the meta-parameter γ and we will find that the solution error is sensitive to its value.

4.4.5 Computation of the final mesh

The transformation (4.4.14) transforms a regular mesh in the computational domain to a mesh in the physical domain, which is compressed close to $r = 0$ and which evolves into an almost uniform mesh as r increases. However, the final transformation does not (quite) match the outer boundary of the region to itself, and we need to make a small adjustment to the transformation for larger values of r to make this possible. The following procedure can be used if there is a single re-entrant corner at the origin, and the domain is 'star-shaped' so that a straight line from the origin to each point on the boundary lies wholly within Ω. Suppose we have a uniform regular mesh $(\xi_{i,j}, \eta_{i,j})$ in Ω_c which we want to map into an adapted non-uniform mesh $(x_{i,j}, y_{i,j})$ in the (star-shaped) physical domain, with $i, j = 1, \ldots, N$. The application of the adjusted OT method provides the desired mesh as follows:

1. For each pair (i, j) compute $s_{i,j}^2 = \xi_{i,j}^2 + \eta_{i,j}^2$.
2. Compute the angle $\theta_{i,j}$ with respect to the semi-positive x axis by $\arctan\left(\frac{\eta_{i,j}}{\xi_{i,j}}\right)$.
3. Compute the length $l_{i,j}$ of the line spanned from the origin with angle $\theta_{i,j}$ to the boundary of the domain.
4. Set the parameter $B = 1 - \gamma$ and enforce the condition $A + \frac{B}{1-\gamma} l_{i,j}^{-2\gamma} = 1$. This ensures that the new mesh boundary matches with the original boundary region.
5. Solve equation (4.4.14) to find $r_{i,j}$. Note that this can be done to low accuracy.
6. Set $(x_{i,j}, y_{i,j}) = \frac{r_{i,j}}{s_{i,j}} (\xi_{i,j}, \eta_{i,j})$.

Given the new mesh $(x_{i,j}, y_{i,j})$, we can increase the resolution by applying the previous procedure from a more graded uniform mesh.

An example of an OT mesh computed in this manner for the L-shaped domain, and $\gamma = 2/3$ has already been given in Figure 4-1 in §4.2. In this case $N = 833$, and the mesh on the right of this Figure is obtained

by solving the Monge-Ampère equation as above, with the uniform triangulation of the L-shaped domain as computational domain on the left side. It is clear from the Figure that the resulting OT mesh is regular, symmetric and smoothly graded towards the corner.

4.4.6 Mesh Quality

In the earlier discussion on h-adaptivity in §4.2 we introduced two measures of mesh quality. The shape regularity $\mu_K = h_k/\rho_K$ in 4.2.5 is a measure of the skewness of each individual cell K. The mesh condition $q(\mathscr{T}) = \|[\![h]\!]/\{\!\{ h \}\!\}\|_{\mathrm{L}^\infty(\mathscr{E}_I)}$ in equation (4.2.10) is a measure of how smoothly graded the mesh is. In the context of an h-adaptive mesh these quantities are estimated from a direct analysis of the mesh itself. In the context of r-adaptive meshes, and especially OT meshes, estimates of these quantities can be obtained directly from the properties of the map from the computational to the physical space.

Global quality measure

A common metric for regularity of the mesh elements in the context of r-adaptive methods is the *skewness* of the cell K, which indicates how far mesh elements are from being equilateral. This metric is very closely related to the shape regularity μ_K of each cell. Given the corresponding Jacobian $\boldsymbol{J}_K$ with eigenvalues λ_1 and λ_2, the quality measure Q_K for the element K is a measure of the skewness and is represented as in Def.2.3.7:

$$Q_K := \frac{1}{2}\left(\frac{\lambda_1}{\lambda_2} + \frac{\lambda_2}{\lambda_1}\right),$$

with global mesh quality measure $Q = \max_{K\in\mathscr{T}} Q_K$.

Note that $Q \geq 1$ and $Q = 1$ for a completely non-skewed mesh. We then have the following result:

Theorem 4.14 *The mesh quality measure Q is bounded as $r \to 0$ with a bound that depends only on γ:*

$$Q(\gamma) = \frac{1}{2}\left(\frac{\lambda_1}{\lambda_2} + \frac{\lambda_2}{\lambda_1}\right) = \frac{1}{2}\left((1-\gamma) + \frac{1}{(1-\gamma)}\right). \tag{4.4.15}$$

This entails two properties for Q:

1. *Independence on the mesh resolution with dimension N.*
2. *Independence on the distance from the origin* $\mathbf{0}$.

1 Proof: Assuming that the *computational mesh* is uniform and so has a mesh quality measure Q independent of N, the quality measure of the *physical mesh* is also independent of N. Under the OT

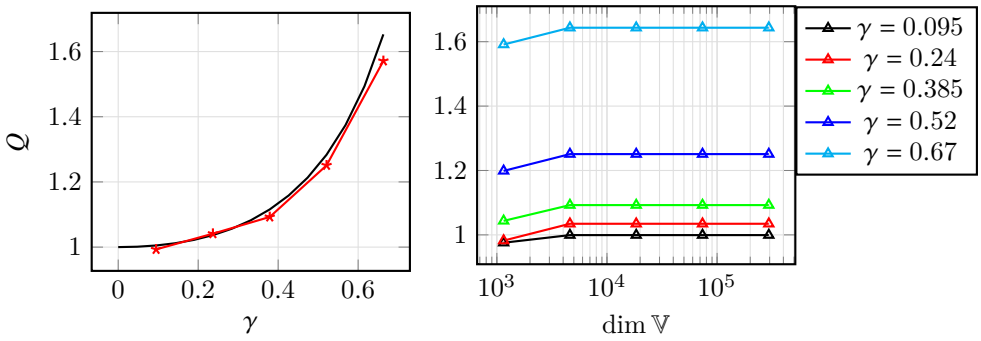
transformation $(x, y) = \frac{r}{s}(\xi, \eta)$, we can determine the value of Q analytically, this allows us to estimate rigorous bounds for Q_K. In particular, when the mesh gets more uniform away from the corner, we assume that the largest values of Q are given for cells where $r \sim 0$. In this case the algebraic expression (4.4.14) simplifies to $r^{1-\gamma} = s$. We then compute the Jacobian of the map $(x, y) = s^{\frac{\gamma}{1-\gamma}}(\xi, \eta)$:

$$\boldsymbol{J} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} = \frac{\gamma}{1-\gamma}(\xi^2+\eta^2)^{\frac{\gamma}{2(1-\gamma)}-1} \begin{bmatrix} \xi^2 + (\xi^2+\eta^2)^{\frac{1-\gamma}{\gamma}} & \xi\eta \\ \xi\eta & \eta^2 + (\xi^2+\eta^2)^{\frac{1-\gamma}{\gamma}} \end{bmatrix}.$$

The eigenvalues of this Jacobian matrix are then $\lambda_1 = \frac{1-\gamma}{\gamma}(\xi+\nu)^2$ and $\lambda_2 = \frac{1}{\gamma}(\xi+\nu)^2$. A simple calculation gives the skewness measure (4.4.15) expressed only in terms of the coefficient γ.

□

We can see from (4.4.15) that we have a good bound on the mesh quality measure. For example, if $\gamma = 2/3$, then $Q = 5/3$, which is still close to one. This implies that the OT meshes are very regular, even as $r \to 1$. However, as γ increases, Q also increases and the mesh quality deteriorates. In Figure 4-7 we plot the resulting values of Q for a variety of meshes for the L-shaped domain, computed using the method of the previous section, which confirms these results. These observations are also qualitatively suggested by looking at the OT mesh produced for the L-shaped and the crack domain and illustrated in Figs. 4-12 and 4-17.



(a) The mesh quality measure Q of the OT mesh as a function of γ.

(b) Quality measure of the OT mesh as a function of $N = \dim \mathbb{V}$.

Figure 4-7: (Left) We show that the dependence of the quality measure on γ computed numerically fits the theoretical formula in (4.4.15) on the L-shaped domain ($\dim \mathbb{V} = 4608$). (Right) The value of $Q(\gamma)$ is independent on the dimension of the FE space.

Mesh condition

The mesh condition $q(\mathcal{T})$ is a measure of how similar adjacent mesh cells are. For the OT meshes constructed above, the primary cell variation is in the radial direction. If we assume as before that we have a radial coordinate $r(s)$, then for some constant Δ we have $\{\!\{ h \}\!\} = \Delta(dr/ds)$ and $[\![h]\!] = \Delta(\mathrm{d}^2 r/\mathrm{d}s^2)\delta s$,

where δs is the increment in s from one cell to the next *in the radial direction*. The continuous approximation for the condition q_K at the K-th cell, is then given by

$$\bar{q}_K = \frac{\mathrm{d}^2 r/\,\mathrm{d}s^2}{\mathrm{d}r/\,\mathrm{d}s}\delta s.$$

Suppose now that we have an OT mesh with $\mathrm{d}r/\,\mathrm{d}s = r^\gamma$. A simple calculation gives

$$\bar{q}(\mathscr{T}) = \frac{\gamma}{(1-\gamma)}\frac{\delta s}{s}.$$

The largest value of $\bar{q}(\mathscr{T})$ arises at the boundary of the first and second cell from the origin, where $s = \delta s$. This gives

$$\hat{q}(\mathscr{T}) = \frac{\gamma}{1-\gamma} \tag{4.4.16}$$

as an estimate for $q(\mathscr{T})$. Observe that $\hat{q}(\mathscr{T})$ increases as $\gamma \to 1$ and $\bar{q}_K$ decreases as s increases.

4.5 Numerical Results

In this section we compare results obtained with the a-posteriori estimate for h-adaptivity and the OT-based r-adaptive method on similarly sized meshes. In particular, we consider as a model problem the following Poisson's equation in the non-convex domain:

$$\begin{aligned} -\Delta u &= 0 \text{ in } \Omega, \\ u_D(r,\theta) &= r^{\pi/\omega}\sin(\pi\theta/\omega) \text{ on } \Gamma, \end{aligned} \tag{4.5.1}$$

where $\omega \in [\pi, 2\pi]$ is the angle of the re-entrant corner located at $\mathbf{0} = (0,0) \in \mathbb{R}^2$. If $\omega = 3\pi/2$ this is an 'L-shaped' domain, and if $\omega = 2\pi$ this is a 'crack' domain. We note that u is analytic in $\overline{\Omega}/\{\mathbf{0}\}$ and $u \in \mathrm{H}^{2,2}_{\boldsymbol{\beta}}(\Omega) \subset \mathscr{C}^0(\overline{\Omega})$, so that $SP(\Omega, \Gamma_D, \Gamma_N) = \{A_1 := \mathbf{0}\}$. Therefore, the weight function is defined as $\Phi_\beta(\boldsymbol{x}) = |\boldsymbol{x}|^\beta$, where $\beta \in (1 - \frac{\pi}{\omega}, 1]$, according to Remark 4.4. Let Ω be the L-shaped domain $(-1,1)\times(-1,1)/([0,1)\times(-1,0])$ with interior angle $\omega = 3\pi/2$ at the re-entrant corner. Then, the harmonic solution of problem (4.5.1) is

$$u(r,\theta) = r^{2/3}\sin(2\theta/3)\,. \tag{4.5.2}$$

We compare the accuracy of the SIP-dG method applied to problem (4.5.1) using the different adaptive mesh strategies and assess the quality of the resulting meshes. For all the tests, we compute the numerical solution $u_h \in \mathbb{V}$ with $p = 1$. All the tests have been performed with the FEniCS library [LW10].

4.5.1 Results for the L-shaped domain

4.5.1.1 h-refinement

In the h-adaptive process, we use the global error indicator $\eta^2_{\mathrm{L}^2}$ and η_{L^∞} defined in (4.3.14) and (4.3.15). As a widely accepted criterion, elements with large error estimator are marked according to longest edge bisection [BR14]. Here, we employ the so called *maximum strategy* (*ms*) starting from a uniform triangular mesh, which can be described as follows:

$$\eta_{\mathrm{L}^2,K} \geq \alpha \max_{K' \in \widetilde{\mathcal{T}}} \eta_{K'} \iff K \text{ is marked for refinement,} \tag{4.5.3}$$

where $\alpha \in (0, 1)$ is a predetermined parameter. The same criterion holds for $\eta_{\mathrm{L}^\infty,K}$. Under this condition, we ensure that the mesh is regular enough in order to satisfy the assumptions in §4.3.1 for the derivation of the a-posteriori estimate.

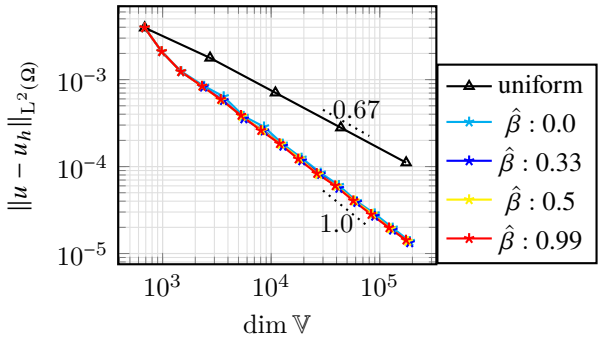
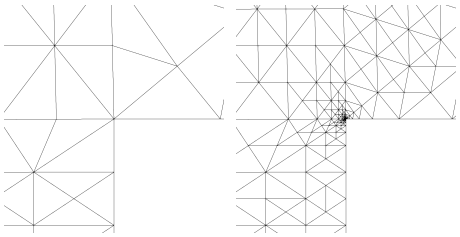


Figure 4-8: Asymptotic convergence rates for the h-refinement strategy on the finite element space $\mathbb{V}$. Note that the rate of convergence is optimal and independent on $\hat{\beta}$. Uniform mesh refinement yields a convergence rate of $2/3$ due to the singular behaviour of the solution at the origin [GMP17].

The *ms* h-refinement strategy has been tested for different $\hat{\beta}$. We observe that the level of refinement is increasing near the corner as a function of $\hat{\beta}$ (Figure 4-9) but does not affect the order of convergence (Figure 4-8).

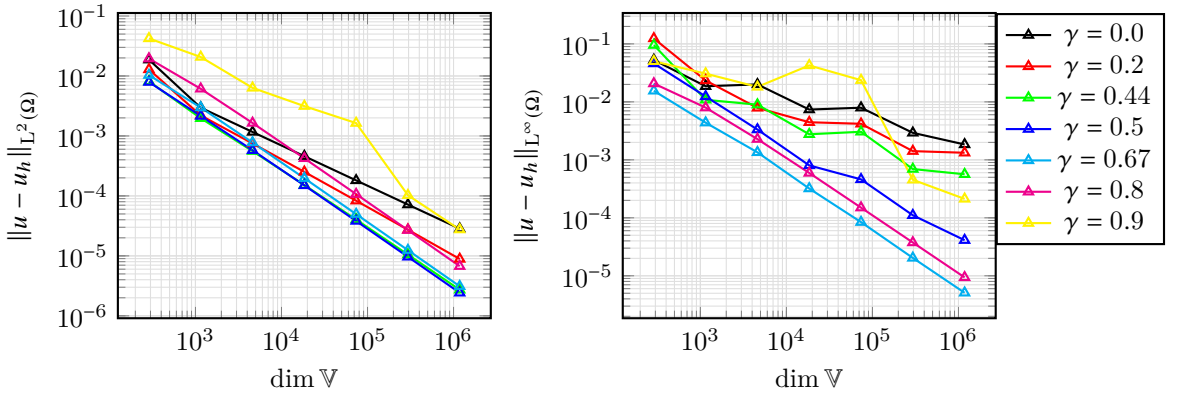


(a) $\hat{\beta} = 0.0$. (b) $\hat{\beta} = 0.99$.

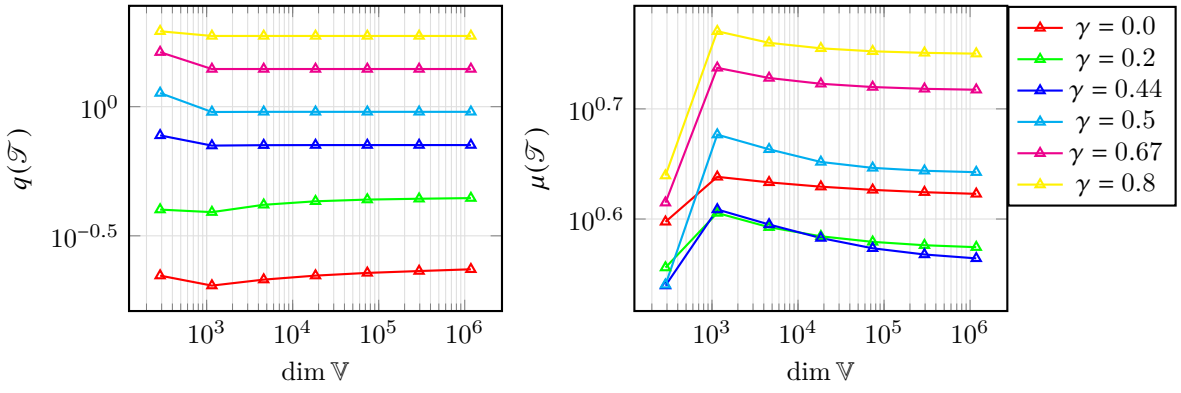
Figure 4-9: Mesh obtained via the L^2 a-posteriori bound (4.3.14) after 15 refinements.

4.5.1.2 OT based meshes for different values of γ

We next consider the results of using a SIP-dG on an OT mesh with $\dim \mathbb{V} = N$ nodes, generated by using different values of γ. In each case we calculate the L^2 error of the solution, and also various mesh quality measures. The results are summarised in Figs. 4-10 and 4-11.

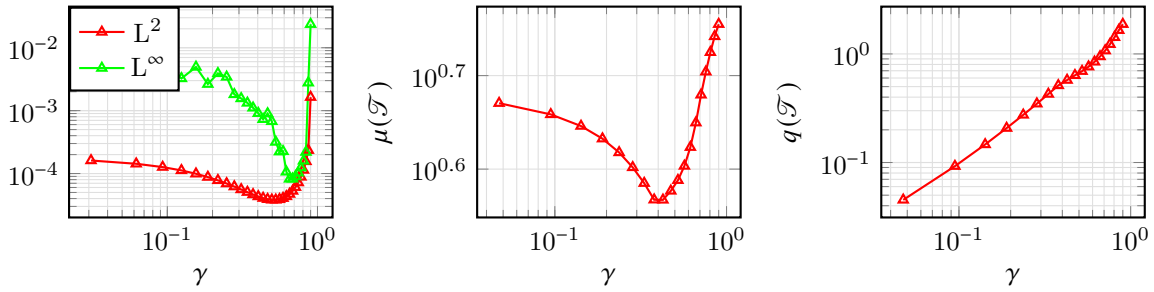


(a) L^2 error which is minimised when $\gamma \approx 4/9$. (b) L^∞ error which is minimised when $\gamma \approx 2/3$.



(c) Mesh condition. (d) Shape regularity.

Figure 4-10: Error and quality measures of the OT mesh for different γ.



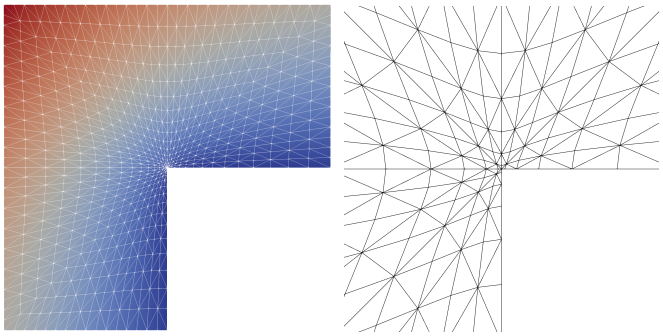
(a) Error. (b) Shape regularity. (c) Mesh condition.

Figure 4-11: Error and quality measures for the a-priori OT mesh as function of γ ($\dim \mathbb{V} = 73728$).

In Figure 4-10a and 4-10b we show the solution error in the L^2 and L^∞ norms plotted as a function of $N = \dim \mathbb{V}$ for a number of values of γ. It is clear from these Figures that the optimal values of γ are very close to those obtained in Lemma 4.13, which equidistribute the respective interpolation errors. Note that for these optimal values of γ, the L^2 error varies (optimally) as $1/N$ as $N \to \infty$. In Figure 4-11a we observe that the L^2 error changes smoothly with respect to γ with a shallow minimum close to the predicted value of $\gamma = 4/9$. In contrast the L^∞ error has a pronounced minimum close to $\gamma = 2/3$, where only a small range of γ values yields a small error.

In Figure 4-11b and 4-11c we plot measures of the mesh quality. We see that the shape regularity $\mu(\mathscr{T})$ given in (4.2.5) decreases as a function of γ until reaching the optimal value found in Lemma 4.13. For higher values of γ, the mesh regularity deteriorates. The mesh condition $q(\mathscr{T})$ given in (4.2.10) increases monotonically as $\gamma \to 1$. This is in agreement with equation (4.4.16).

In Figure 4-12 we present the resulting mesh for $\gamma = 0.44$.



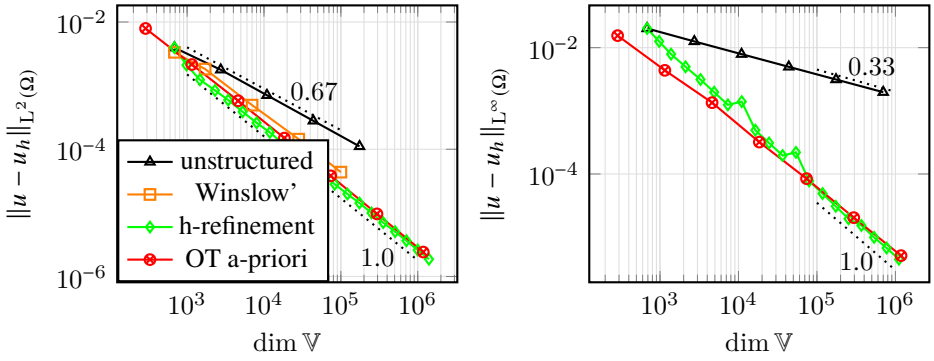
(a) Solution of Poisson's equation. (b) Zoom on the re-entrant corner.

Figure 4-12: Solution of equation (4.5.1) on the OT mesh with $\gamma = 0.44$. Note the smooth grading and symmetry of this mesh even with $q(\mathscr{T}) > 1$.

4.5.1.3 Comparison between the h-refinement and OT refinement methods

Finite Element errors

In Figure 4-13a and 4-13b we compare the convergence rate of the FE error with an h-refined mesh, the OT mesh and a uniform mesh for the same value of $N = \dim \mathbb{V}$. We see an optimal $1/N$ convergence rate for both adaptive meshes. In contrast, we observe convergence rates of $1/N^{2/3}$ and $1/N^{1/3}$ for the uniform mesh.



(a) L^2 error. (b) L^∞ error.

Figure 4-13: Convergence rates for different adaptive strategies. The rate of convergence is optimal $(1/N)$ for both h-refinement and for the OT strategy ($\gamma = 0.44, 0.67$ for L^2 and L^∞ norm, respectively).

Run time for the computation of the solution on the two adapted meshes

In Figure 4-14 we present the CPU run time (calculated on a 16 GB RAM Computer with an Intel Core i7-10510U processor) for the computation of the solution using both the h-refinement and the OT mesh averaged for 10 runs. In the first case, the solution is started with a coarse mesh and successively refined. Additional computational time is required to compute the approximate solution of Poisson's equation and evaluate the error indicators $\eta_{L^2,K}$ (eq.(4.3.14)). In the second case, there is an upfront cost for calculating the mesh, followed by a single cost for finding the solution on that mesh. The execution time is mainly dependent on the Newton's algorithm used for solving the nonlinear algebraic equation for each mesh point (eq. (4.4.14)).

The link between the OT mesh and the a-posteriori estimate

In this section we analyse the relationship between the a-posteriori bounds η_2, η_∞ used for h-adaptivity and the OT mesh obtained in the previous subsections. We first remark that effect of h-refinement is to approximately equidistribute the a-posteriori solution error η_K throughout each cell by reducing the element mesh size until η_K is less than a prescribed tolerance. The result of this procedure leads to the minimisation of the overall estimated FE error. In contrast the OT mesh with the value of γ given in Lemma 4.13 is designed to equidistribute the linear interpolation error in each cell. To make a

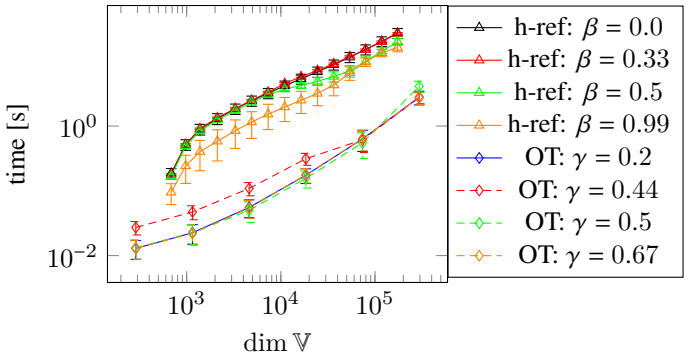
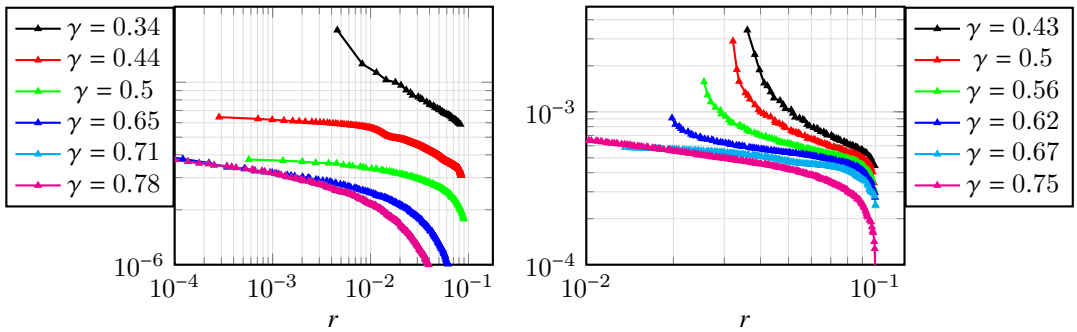


Figure 4-14: CPU-runtime comparison.

comparison we ask whether, following the calculation of the solution, the OT mesh also equidistributes η_K in the vicinity of the origin. In Figure 4-15 we present the values of η_K calculated as a function of r on the OT mesh close to the origin in both the L^2 and L^∞ norms. In this calculation we fix $\dim \mathbb{V} = 7 \times 10^4$ and consider the effect of varying γ.



(a) $\eta_{L^2,K}$.

(b) $\eta_{L^\infty,K}$.

Figure 4-15: Cell values of the a-posteriori measure $\eta_{L^2,K}, \eta_{L^\infty,K}$ for OT meshes with $\dim \mathbb{V} = 7 \times 10^4$ for different values of γ. The measure has been computed as a function of the distance from the re-entrant corner.

It is apparent from this Figure that the OT mesh with the optimal value of γ does equidistribute η_K (in both norms) as $r \to 0$. We observe for the L^∞ norm in particular that the equidistribution property is very sensitive to the value of γ. This is consistent with the FE error also having a high sensitivity on the value of γ.

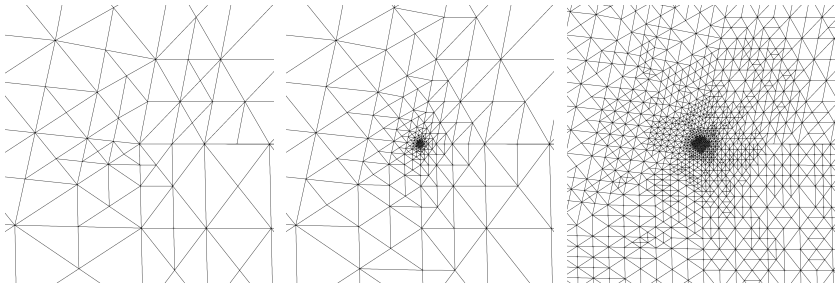
4.5.2 Results for the crack domain

We now consider the results of using the same approaches as in the previous section but in a domain Ω with a crack $(-1,1)\times(1,1)/([(0,1)\times\{0\}])$. The solution of equation 4.5.1 in this case is:

$$u(r,\theta) = r^{1/2}\sin(\theta/2)\,. \tag{4.5.4}$$

Numerically, we set the interior angle ω to be $2\pi-\epsilon$, with $\epsilon = 10^{-3}$. We then, as before, calculate the solution using an h-adaptive SIP-dG method, and compare this with the solution on the OT meshes for various values of γ.

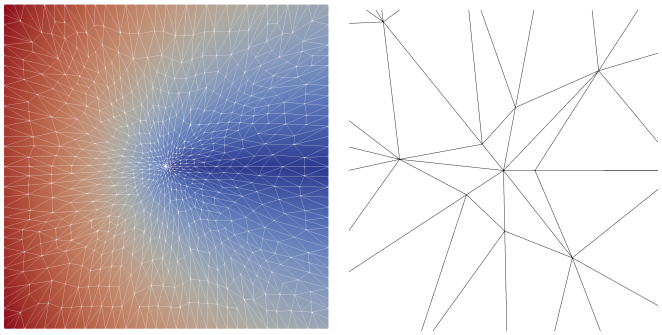
In Figure 4-16 we show the resulting h-adapted meshes for various values of β. In comparison, we



(a) L^2 a-posteriori ($\beta = 0.0$). (b) L^2 a-posteriori ($\beta = 0.99$). (c) L^∞ a-posteriori bound.

Figure 4-16: Zoom of the mesh near the re-entrant corner obtained with the a-posteriori bounds (4.3.14),(4.3.15).

present in Figure 4-17 the OT mesh taking $\gamma = 1/2$ together with the solution on this mesh. This again shows good regularity and symmetry close to the origin.

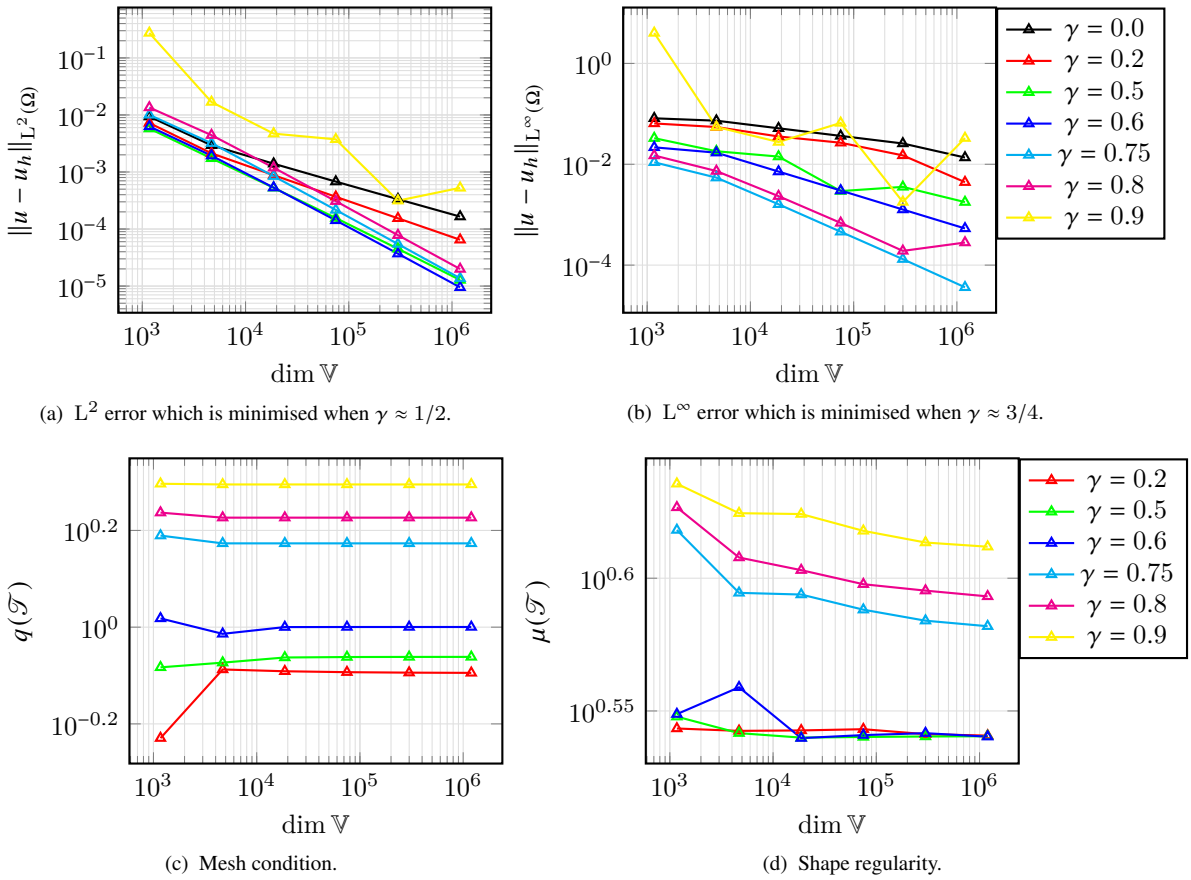


a Solution of Poisson's equation. b Zoom of the OT mesh near $\mathbf{0}$.

Figure 4-17: Solution of Poisson's equation (4.5.4) with the OT mesh generated as in §4.4.5.

We repeat the earlier calculations to show how the parameter γ affects the accuracy and the quality of

the resulting OT mesh. The results are shown in Figure 4-18. The optimal values of γ that minimise the FE errors in the two norms, leading to $O(1/N)$ convergence, are in broad agreement with the ones given in Lemma 4.13.



(a) L^2 error which is minimised when $\gamma \approx 1/2$.

(b) L^∞ error which is minimised when $\gamma \approx 3/4$.

(c) Mesh condition.

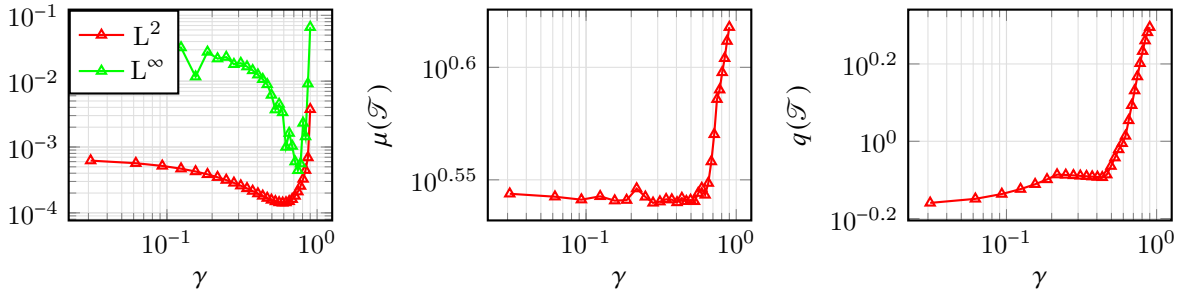
(d) Shape regularity.

Figure 4-18: Error and quality measures for the a-priori OT mesh as function of γ ($\dim \mathbb{V} = 74880$).

Similarly, the results presented in Figure 4-20 show that the convergence rate of the solutions on the optimal OT mesh and the *h*-adapted mesh are very comparable.

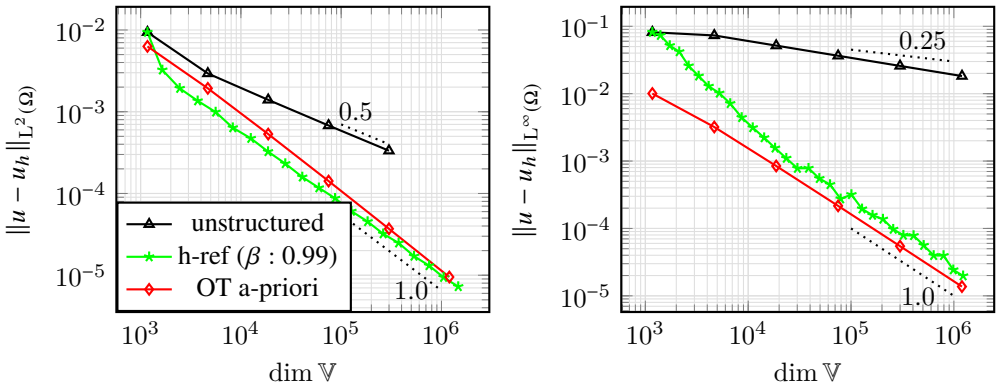
In Figure 4-19, we see that the trend error trend is similar to the L-shaped case (Figure 4-11). On the other hand, the shape regularity does not appear to consistently decrease for $\gamma \in (10^{-1}, 5\times10^{-1})$. Also, the mesh condition significantly increase for γ higher than the right extremum of the previous interval.

Finally, the plots presented in Figure 4-21 show that, as in the L-shaped domain, the a-posteriori estimator is equidistributed near the origin for the value of γ providing the highest accuracy.



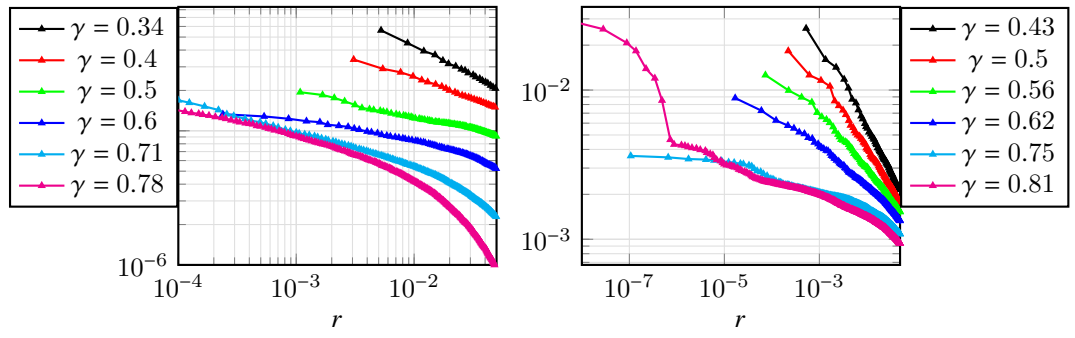
(a) Error. (b) Shape regularity. (c) Mesh condition.

Figure 4-19: Error and quality measures for the a-priori OT mesh as function of γ ($\dim \mathbb{V} = 74880$).



(a) L^2 error. (b) L^∞ error.

Figure 4-20: Asymptotic convergence rates for different adaptive strategies. Note that the rate of convergence is optimal for the h-refinement and the OT strategy ($\gamma = 0.5, 0.75$ for the L^2 and L^∞ norm, respectively).



(a) $\eta_{L^2,K}$. (b) $\eta_{L^\infty,K}$.

Figure 4-21: Cell values of the a-posteriori measure $\eta_{L^2,K}, \eta_{L^\infty,K}$ for OT meshes with $\dim \mathbb{V} = 7 \times 10^4$ for different values of γ. The measure has been computed as a function of the distance from the re-entrant corner.

4.6 Summary

In this Chapter, we have analysed the relationship between *h*-adaptivity and an OT based strategy for the solution of Poisson's equation on non-convex domains with the SIP-dG method. Under this framework, we derived a new L^2 a-posteriori error estimate and adopted an L^∞ error estimate from [DG12] for the application of the *h*-refinement strategy.

We have then constructed an OT mesh, which accounts for the singular behaviour of the solution at the re-entrant corner. We evidenced that the *skewness* of this mesh increases locally only as a function of the re-entrant angle and is independent on the resolution of the mesh and on the location of the cell. This indicates that the OT mesh does mainly depend on the radial variable and is possible to formulate the OT problem in one-dimension, avoiding the numerical treatment of the fully two-dimensional Monge-Ampère equation. Optimality of the convergence rate and accuracy requires the specification of a meta-parameter γ, which controls the level of node clustering close to the origin. The optimal values have been derived theoretically from local interpolation error estimates using an equidistribution argument for L^2 and L^∞ norms. Although the estimate might be sub-optimal, as does not target directly the FE error, it has been showed that the resulting OT mesh provides the same level of accuracy as the *h*-refined one. Both the methods achieve optimal convergence, in contrast to Winslow's method supplied with different monitor functions. The major advantage of the OT based method is that the solution is obtained in one step, whereas the *h*-refinement strategy demands the computation of the numerical solution at each stage of the mesh refinement. We also remark the close relationship between the two adaptive strategies, as the optimally-tuned OT mesh equidistributes the a-posteriori estimates used for *h*-refinement in order to minimise the global FE error.

The next Chapter will address the same Poisson problem on the L-shaped domain with the deep learning framework. We will detail the procedure to train a neural network with two different loss (objective) functions and use as training points the OT mesh vertices and randomly sampled points. We will observe that the location of those points and the choice of quadrature rule for integral approximation is crucial in order to obtain an accurate result.

Chapter 5

The Deep Galerkin and the deep Ritz method for Poisson's equation on the L-shaped domain

Abstract

Following up on Chapter 4, we propose alternative approaches to solve Poisson's equation on the L-shaped domain by means of a neural network. We will detail the network structure, training procedure and define two different loss functions, based on the Galerkin and Ritz method used in finite element method. Numerical experiments reveal that the location of the training points and a quadrature rule are crucial for the accuracy of the solution. If these points are chosen as vertices of the OT mesh derived in Chapter 4, the accuracy is much higher compared to the usual case of functions using points randomly sampled. Moreover, the deep Ritz method is also naturally adaptive and contributes to increase the overall accuracy. We will compare the convergence rate of the DRM and of the SIP-dG method discussed in the previous Chapter. We will observe that neural network is more accurate than the SIP-dG only for solution with small-resolution, but the error does not decrease as for the FE based method with additional degrees of freedom (*dofs*). A slightly better performance in terms of convergence rate is achieved over a squared domain.

5.1 Introduction

In the field of numerical analysis, the strong representability of functions using deep neural networks (DNNs) means that they are becoming increasingly popular for solving partial differential equations

(PDEs) [LLF98; Lon+18; RK18; SS18].

Compared to standard PDE solvers, such as Finite Element (FE) [BS08] and Finite Difference (FD) methods [Smi86], deep learning (DL) methods provide a functional approximation of the solution of the PDE using a combination of linear and non-linear transformations, as shown in §2.4.

In the current Chapter, we extend the work in the previous Chapter by studying the solution of the Poisson's equation on the L-shaped domain using the deep Galerkin method (DGM) [SS18] and the deep Ritz method (DRM) [WB18]. Both train the network by minimising a loss (objective) function. For the DGM, this is defined as the L^2 norm of the residual of the PDE. In contrast, the latter method defines the loss function as the energy functional of the PDE. This brings two advantages. Firstly, the approximate solution can admit lower regularity when compared to the the DGM. Secondly, although the discretised loss is not convex, its variational formulation is supposed to be naturally adaptive under the optimisation process [WB18].

In certain simple cases, the integrals defining the loss functions can be approximated in a *meshless* fashion [BBO03], by computing the mean squared error (MSE) directly using the functional approximation, or over a mesh with a quadrature rule in more complex cases. The evaluation/quadrature points are usually sampled uniformly at random [Che20] or via Latin Hypercube sampling (LHS) [RCS05]. The uniformly random sampling poses critical issues in the training process, as the integrals defining the loss functions are not approximated well with the resulting quadrature method. Also, the random collocation of points inside the domain makes the network unaware of any singular behaviour of the solution, such as that one arising in a L-shaped domain. As numerically shown in Chapter 4, posing more points on the area close to the singularity, considerably increases the accuracy of the approximate solution.

In classical numerical methods, boundary conditions can be exactly enforced for mesh points at the boundary [Eva10]. However, it is very difficult to impose exact boundary conditions for a DNN representation [WTP21; Che20; Liy+21]. Therefore, in the loss function, it is common to add a penalty term in L^2 norm which penalises the difference between the DNN representation on the boundary and the exact boundary condition.

Our contribution in this Chapter is twofold. At first, we compare the solutions given for Poisson's equation on a L-shaped domain using the DGM and the DRM trained over a fixed OT mesh (quadrature rule) and over fixed collocation points (MSE). In the second place, we seek the optimal values of the penalty term to ensure a positive convergence rate and compare this with the performance of the SIP-dG method shown in §4.5.1.3.

The remainder of this Chapter is as follows. Section 5.2 describes the components for training the neural network, with details about the network structure, the loss function of the DGM and DRM, and the optimisation procedure. Section 5.3 proposes numerical experiments for the solution of the Poisson

equation using the two neural networks. We draw our conclusions in §5.5.

5.2 Methodology

The usage of a DNN to solve a PDE problem comprises of three parts: the neural network structure used to approximate the solution of the PDE, the loss function, and the way it is optimized over the parameter space. We first illustrate the network structure used to approximate the PDE solution. Then we introduce the DGM and DRM, the corresponding loss function and explain how boundary conditions are treated using the penalty method. Finally, we describe the optimisation method used to update the parameters of the DNN and find the approximate solution of the PDE.

5.2.1 Network structure

The basic component of the training of the DGM and DRM is a nonlinear transformation $\boldsymbol{x} \in \mathbb{R}^d \to \mathbb{R}^m$, where $\boldsymbol{x}$ is the usual spatial variable defined on the open bounded domain $\Omega \subset \mathbb{R}^d$. Each layer of the network is made up of blocks stacked on each other. Each block consists of two linear transformations, two activation functions and a residual connection. We follow the same notation as introduced in §2.4.1 of Chapter 2. Given L blocks, we denote by $\boldsymbol{W}_0 \in \mathbb{R}^{m\times d}, \boldsymbol{b}_0 \in \mathbb{R}^m$ the weight matrices and the bias vectors for the first layer, with output $u^0_\Theta = \boldsymbol{W}_0\boldsymbol{x} + \boldsymbol{b}_0$. For a generic l-th block, let $u^l_\Theta(\boldsymbol{x}) \in \mathbb{R}^m$ be the input, and let $\boldsymbol{W}_{l,1}, \boldsymbol{W}_{l,2} \in \mathbb{R}^{m\times m}, \boldsymbol{b}_{l,1}, \boldsymbol{b}_{l,2} \in \mathbb{R}^m$ be the weight matrices and the bias vectors. The element-wise activation function is denoted by $\sigma(\cdot)$, and $u^{l+1}_\Theta(\boldsymbol{x})$ is the output expressed as

$$u^{l+1}_\Theta(\boldsymbol{x}) = \sigma\Big(\boldsymbol{W}_{l,2} \circ \sigma(\boldsymbol{W}_{l,1}u^l_\Theta(\boldsymbol{x}) + \boldsymbol{b}_{l,1}) + \boldsymbol{b}_{l,2}\Big) + u^l_\Theta(\boldsymbol{x}). \tag{5.2.1}$$

The final output is $u_\Theta(\boldsymbol{x}) = \boldsymbol{W}_L u^L_\Theta(\boldsymbol{x}) + \boldsymbol{b}_L$, where $\boldsymbol{W}_L \in \mathbb{R}^{m\times d}$ and $\boldsymbol{b}_L \in \mathbb{R}^m$.

Skip Connections (or Shortcut Connections), as the name suggests, skips some of the layers in the neural network and feeds the output of one layer as the input to the next layers. Skip Connections were introduced to avoid the problem of vanishing gradients [Hoc98]. Additionally, the loss surface of the neural network with skip connections is smoother and thus leading to faster convergence than the network without any skip connections [Li+18].

For this type of architecture, possible choices of activation function are

- $\text{sigmoid}(x) = \frac{1}{1+\exp(-x)}$.
- $\text{swish}(x) = \frac{x}{1+\exp(-x)}$.
- $\tanh(x)$.

- $\sigma_{sin}(x) = (\sin(x))^3$.

Overall, the output $u_\Theta(\boldsymbol{x})$ of the DNN is expressed as the composition

$$u_\Theta(\boldsymbol{x}) = u_\Theta^{L+1} \circ \cdots \circ u_\Theta^0(\boldsymbol{x}), \tag{5.2.2}$$

where Θ is the full set of all weight and bias parameters in the neural network, i.e. $\Theta = \{\boldsymbol{W}_0, \boldsymbol{b}_0, \boldsymbol{W}_L, \boldsymbol{b}_L\} \cup \{\boldsymbol{W}_{l,1}, \boldsymbol{W}_{l,2}, \boldsymbol{b}_{l,1}, \boldsymbol{b}_{l,2}\}_{l=1}^{L-1}$. In each case of δ the output $u_\Theta(\boldsymbol{x})$ is a smooth function defined over all Ω and can be exactly differentiated to all orders. In particular, $u_\Theta(\boldsymbol{x}) \in C^\infty(\Omega)^m$ and cannot approximate accurately functions with singularities, lying in low order Hilbert spaces.

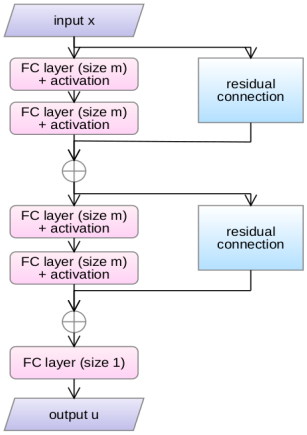


Figure 5-1: Architecture of a deep Ritz network [WB18]. Two fully-connected (FC) layers and the residual connection is described in eq. (5.2.1).

An illustrative example of the architecture is shown in Figure 5-1. We denote by $\mathcal{H}_\sigma^m(\Theta, L)$ the set of functions $u_\Theta(\boldsymbol{x}) \in \mathbb{R}^m$ given as output of the DNN with L blocks, m nodes per hidden layer and activation function $\sigma(\cdot)$.

5.2.2 The deep Galerkin method and the deep Ritz method

Consider the following well-posed Poisson's equation over a bounded domain $\Omega \subset \mathbb{R}^d$

$$\begin{aligned} -\Delta u(\boldsymbol{x}) &= f(\boldsymbol{x}) \ \text{ in } \ \Omega, \\ u(\boldsymbol{x}) &= u_D(\boldsymbol{x}) \text{ on } \partial\Omega. \end{aligned} \tag{5.2.3}$$

where $f(\boldsymbol{x})$ is a given function and u_D are Dirichlet boundary conditions in the input space.

It has been shown that DNNs are universal approximators, so that any continuous function can be approximated up to arbitrary precision based on the network parameters [HSW89]. The DGM and DRM uses a DNN to find an approximate solution $u_\Theta(x)$ to the problem 5.2.3 [WB18; ML21].

For the problem (5.2.3), we define the *strong-form* residual $r(u_\Theta(\boldsymbol{x}))$ and the *boundary* residual $r_b(u_\Theta(\boldsymbol{x}))$ by

$$\begin{aligned} r(u_\Theta(\boldsymbol{x})) &= \Delta u(\boldsymbol{x}) + f(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega, \\ r_b(u_\Theta(\boldsymbol{x})) &= u_\Theta(\boldsymbol{x}) - u_D(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega. \end{aligned} \tag{5.2.4}$$

Those residuals reflect how close the network output $u_\Theta(\boldsymbol{x})$ is to the target function $u(\boldsymbol{x})$. We compute the weighted integrals of the residuals by projecting them onto chosen test functions $v_j \in \mathbb{V}$, where V is a finite dimensional vector space with $\dim(\mathbb{V}) = N$, to obtain the variational forms

$$\begin{aligned} \mathcal{R}_j(u_\Theta) &= \int_\Omega r(u_\Theta(\boldsymbol{x}))v_j \,\mathrm{d}\boldsymbol{x} = 0, \\ \mathcal{R}_{b,j}(u_\Theta) &= \int_{\partial\Omega} r_b(u_\Theta(\boldsymbol{x}))v_j \,\mathrm{d}S(\boldsymbol{x}) = 0, \quad j = 1, \dots, N. \end{aligned} \tag{5.2.5}$$

To solve the nonlinear system resulting from these equations, we define the general loss function:

$$\mathcal{I}(u_\Theta(\boldsymbol{x}), v) = w_r \sum_{j=1}^{N_r} \mathcal{R}_j^2(u_\Theta(\boldsymbol{x})) + w_b \sum_{j=1}^{N_b} \mathcal{R}_{b,j}^2(u_\Theta(\boldsymbol{x})), \tag{5.2.6}$$

where the parameters $\{w_r, w_b\}$ denote the weight coefficients in the loss function. They are usually tuned manually or automatically learned [WTP21].

Different choices of test functions v_j correspond to different deep learning methods. For example, the *Delta dirac test functions* $v_j(\boldsymbol{x}) = \delta(\boldsymbol{x} - \boldsymbol{x}_j)$, for suitably chosen $\boldsymbol{x}_j$, correspond to the collocation methods, also known as PINNs [RPK19].

The deep Galerkin method

The Deep Galerkin Method results from the choice of $v_i = r(u_\Theta(\boldsymbol{x}_i))\delta(\boldsymbol{x} - \boldsymbol{x}_i),\ i = 1, \dots, N_r$ for the interior of Ω, and $v_j = r(u_\Theta(\boldsymbol{x}_j))\delta(\boldsymbol{x} - \boldsymbol{x}_j),\ j = 1, \dots, N_b$ for the boundary $\partial\Omega$, such that

$$\mathcal{I}_{DGM}(u_\Theta(\boldsymbol{x})) = w_r \sum_{i=1}^{N_r} \Big(\Delta u(\boldsymbol{x}_i) + f(\boldsymbol{x}_i)\Big)^2 + w_b \sum_{j=1}^{N_b} \Big(u_\Theta(\boldsymbol{x}_j) - u_D(\boldsymbol{x}_j)\Big)^2. \tag{5.2.7}$$

The deep Ritz method

The variational formulation for the Deep Ritz method is based on the homonym introduced in §2.3.3 of Chapter 2. The aim is to find $u \in H_0^1(\Omega)$ such that

$$\int_\Omega \nabla u(\boldsymbol{x}) \cdot \nabla v \, \mathrm{d}\boldsymbol{x} = \int_\Omega f(x) \cdot v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \ v \in H_0^1(\Omega), \tag{5.2.8}$$

where the left hand side is given by integration by parts and homogeneous boundary conditions.

The corresponding objective function to be minimised for the DRM is obtained by choosing $v = u_\Theta(\boldsymbol{x})$ and enforcing the boundary condition of eq.(5.2.3):

$$\mathcal{I}_{DRM}(u_\Theta(\boldsymbol{x})) = w_r \int_\Omega \left[\frac{1}{2} \nabla u_\Theta(\boldsymbol{x}) \cdot \nabla u_\Theta(\boldsymbol{x}) - f(\boldsymbol{x}) \cdot u_\Theta(\boldsymbol{x}) \right] \mathrm{d}\boldsymbol{x} + w_b \int_{\partial\Omega} \left(u_\Theta(\boldsymbol{x}) - u_D(\boldsymbol{x}) \right)^2 \mathrm{d}S(\boldsymbol{x}). \tag{5.2.9}$$

The optimal approximation $u_{\Theta^*}(\boldsymbol{x}) \in \mathbb{R}^m$ for the DGM and DRM is obtained by solving the following optimization problem:

$$u_{\Theta^*}(\boldsymbol{x}) = \underset{u_\Theta(\boldsymbol{x}) \in \mathcal{H}_\sigma^m(\Theta, L)}{\arg\min} [\mathcal{I}(u_\Theta(\boldsymbol{x}))], \tag{5.2.10}$$

where $\mathcal{H}_\sigma^m(\Theta, L)$ is the set of admissible functions given as output of the DNN.

5.2.3 The optimisation algorithm and the loss approximation

Using DNNs to solve PDEs has been restated as the minimisation problem (5.2.10). Even if the original PDE is linear, the DNN representation (5.2.2) can be highly nonlinear in the parameters $\boldsymbol{q}$ due to the successive composition of the nonlinear activation functions.

Quadrature schemes for the integrals in equation (5.2.9) are also required for a good accuracy of the network output. Nonetheless, most of papers addressing PDEs via DNN approximate (5.2.7) using a Monte-Carlo method, for both low and high dimensions [SS18; WB18; RPK19]. In this context, stochastic gradient descent (SGD) and its variants, play a key role in deep learning training. It is a first-order optimization method which naturally incorporates the idea of Monte-Carlo sampling. At each iteration, SGD updates the network parameters by evaluating the gradient of the loss function only at a batch of samples as

$$\Theta_{k+1} = \Theta_k - \eta_k \nabla_{\Theta_k} MSE\left(\mathcal{I}(\Theta_k)\right), \tag{5.2.11}$$

where

$$MSE\left(\mathcal{I}_{DGM}(\Theta_k)\right) = \frac{w_r}{N_r}\sum_{i=1}^{N_r}\Big(\Delta u_{\Theta_k}(\boldsymbol{x}_i) + f(\boldsymbol{x}_i)\Big)^2 + \frac{w_b}{N_b}\sum_{j=1}^{N_b}\Big(u_{\Theta_k}(\boldsymbol{x}_j) - g(\boldsymbol{x}_j)\Big)^2,$$
$$MSE\left(\mathcal{I}_{DRM}(\Theta_k)\right) = \frac{w_r}{N_r}\sum_{i=1}^{N_r}\Big[\frac{1}{2}\nabla u_{\Theta_k}(\boldsymbol{x}_i)\cdot\nabla u_{\Theta_k}(\boldsymbol{x}_i) - f(\boldsymbol{x}_i)\cdot u_{\Theta_k}(\boldsymbol{x}_i)\Big] + \frac{w_b}{N_b}\sum_{j=1}^{N_b}\Big(u_{\Theta_k}(\boldsymbol{x}_j) - g(\boldsymbol{x}_j)\Big)^2. \tag{5.2.12}$$

The set of parameters of the DNN at the k-th iteration is denoted by Θ_k, and η_k is the learning rate. In general, the coordinates $\{\boldsymbol{x}_i\}_{i=1}^{N_r}$ are randomly generated over Ω and $\partial\Omega$ [Che20; RCS05]. The gradient of $MSE\left(\mathcal{I}(\theta_k)\right)$ with respect to Θ_k is computed via Automatic Differentiation (AD), which has been described in §2.4.3 of Chapter 2.

The problem of this method is that neglects the geometric properties of the solution and does not necessarily approximate well the loss integrals, especially if $u(\boldsymbol{x})$ has a singularity. This issue can be addressed by choosing collocation/quadrature and boundary points to compose a mesh. To ensure good regularity properties, the set of mesh vertices $\{\boldsymbol{x}_i\}_{i=1}^{N}$ is then fixed and taken from a Delaunay triangulation or an OT mesh. The loss integrals in (5.2.9) can be then approximated on that mesh with any appropriate quadrature rule.

The network parameters are updated according to

$$\Theta_{k+1} = \Theta_k - \eta_k \nabla_{\Theta_k}\mathcal{I}(\Theta_k), \tag{5.2.13}$$

Finally, as discussed in Chapter 2, we will use the *Adam* optimization algorithm in the numerical experiments, whose flowchart (2) is in §2.4.2. This method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. The advantage of Adam over the standard SGD is that the magnitudes of parameter updates are invariant to rescaling of the gradient and the step sizes are approximately bounded by the corresponding meta-parameter [KB14; Rud16]. The algorithm flowchart 2 in Chapter 2 provides additional implementation details.

5.3 Numerical Results - 1

In this section, we present the numerical results for the Poisson problem (5.2.3). The approximate solution is computed via the DGM and DRM. As mentioned in §5.2.3, the second method approximates the loss integrals with a quadrature rule on the OT mesh derived in §4.4.5 of Chapter 4. The error is then evaluated on a Delaunay mesh constructed with *mshr*, the mesh generation component of FEniCS [LW10]. In all experiments, the block-based DNN is implemented in PyTorch [Pas+19].

The network configuration and parameters are:

- *max epochs* = 50000.
- *learning rate* (lr) = 10^{-4}.
- n° *hidden nodes* = 20.
- n° *layers* = 4.
- *activation function* $\sigma(\cdot) = \tanh(\cdot)$.

The loss value and the L^2 error is computed every 1000 epochs. The values for the above parameters are set based on multiple code runs. We noticed that 4 layers and 20 hidden nodes represent a good compromise between the accuracy of the numerical result and the CPU runtime. The $\tanh(\cdot)$ activation function has been used because it is continuously differentiable and takes real values, whereas the *sigmoid* activation function takes only positive values. The learning rate of 10^{-4} ensures that the loss function monotonically decreases such that after *max epochs* the error stagnates.

5.3.1 Poisson's equation

We consider the Poisson problem with Dirichlet boundary condition as in equation (4.5.1) of Chapter 4. We use the two-dimensional radial coordinates $(r, \theta) \in \mathbb{R}^+ \times [0, 2\pi)$ to define the problem on the L-shaped domain:

$$\begin{aligned} -\Delta u(r,\theta) &= 0 \text{ in } \Omega, \\ u_D(r,\theta) &= r^{2/3}\sin(2\theta/3) \text{ on } \partial\Omega, \end{aligned} \tag{5.3.1}$$

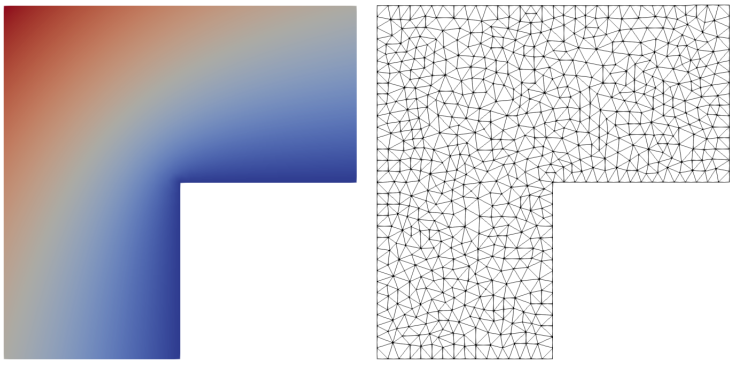
with $\Omega = (-1,1)\times(-1,1)/([0,1)\times(-1,0])$. The solution $u(r,\theta) = r^{2/3}\sin(2\theta/3)$ exhibits a corner singularity in $\{\mathbf{0}\}$.

To account for the boundary condition, we train the network using the DGM with the loss defined by (5.2.7), with $u_\Theta \in \mathrm{H}^2(\Omega)$.

Similarly, the loss function for the DRM follows from equation (5.2.9) and takes the form

$$\mathcal{I}_{DRM}(u_\Theta) = \frac{1}{2}\int_\Omega |\nabla u_\Theta(\boldsymbol{x})|^2 \,\mathrm{d}\boldsymbol{x} + w_b \int_{\partial\Omega} \Big(u_\Theta(\boldsymbol{x}) - u_D(\boldsymbol{x})\Big)^2 \,\mathrm{d}S(\boldsymbol{x}), \tag{5.3.2}$$

with $u_\Theta \in \mathrm{H}^1(\Omega)$.



(a) Solution of Poisson's equation. (b) Delaunay mesh with $N = 833$.

Figure 5-2: Exact solution of equation (5.3.1) and Delaunay mesh with $N = 833$ on which the L^2 error is computed.

We will compute the relative L^2 error, defined by

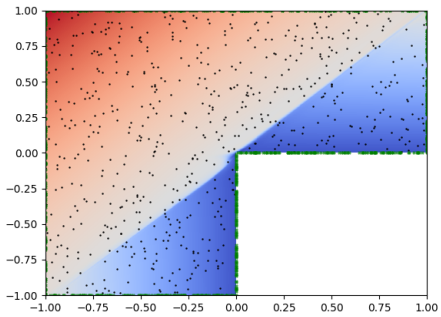
$$\text{relative } \mathrm{L}^2 \text{ error} := \frac{\|u(\boldsymbol{x}) - u_\Theta(\boldsymbol{x})\|_{\mathrm{L}^2(\Omega)} + \|u_D(\boldsymbol{x}) - u_\Theta(\boldsymbol{x})\|_{\mathrm{L}^2(\partial\Omega)}}{\|u(\boldsymbol{x})\|_{\mathrm{L}^2(\Omega)} + \|u_D(\boldsymbol{x})\|_{\mathrm{L}^2(\partial\Omega)}}, \tag{5.3.3}$$

by evaluating the approximated solution on the Delaunay mesh, as shown in Figure 5-2. The number of vertices N is the same as the number of collocation/quadrature points used to train the network with the DGM/DRM method.

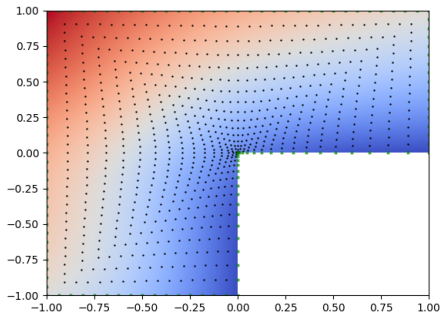
The neural network is fed with input $\boldsymbol{x} \in \Omega \subset \mathbb{R}^2$ to compute the approximate solution $u(r, \theta)$ of problem (5.3.1). The training of the DNN stops when the loss function changes below a relative tolerance of 10^{-5} every 1000 epochs. In the cases where the desired tolerance is not achieved, the training stops at *max epochs*. Finally, the penalty parameter w_b, which weights the importance of boundary residuals, takes values in the set $\{1, 10, 100, 500\}$.

The main difference in accuracy is related to the location of the collocation/quadrature points, as shown in Figure 5-3. We observe that the choice of fixed random collocation points leads to a poor solution accuracy for both methods, with a sharp transition from low (blue) and high (red) function values on the interior domain. The output obtained by the DRM is closer to the real solution (Figure 5-3c).

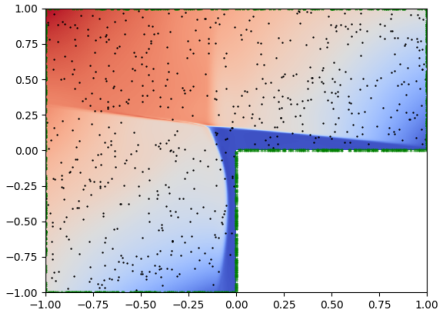
Concerning the OT points, we note that both DGM and DRM perform better than their counterparts with random collocation points and the output looks very similar to the exact solution, as shown in Figure 2-10 of Chapter 2. We will compare the L^2 error and loss of the two methods in the next sections. The former is computed by using the trapezoidal quadrature rule over the Delaunay mesh in Figure 5-4.



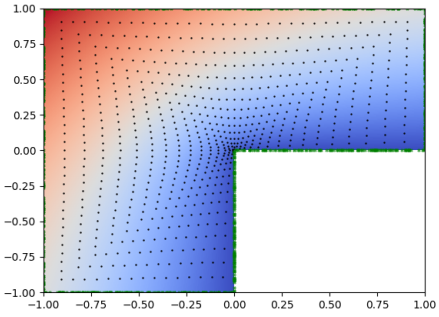
(a) DGM: random points



(b) DGM: OT quadrature points

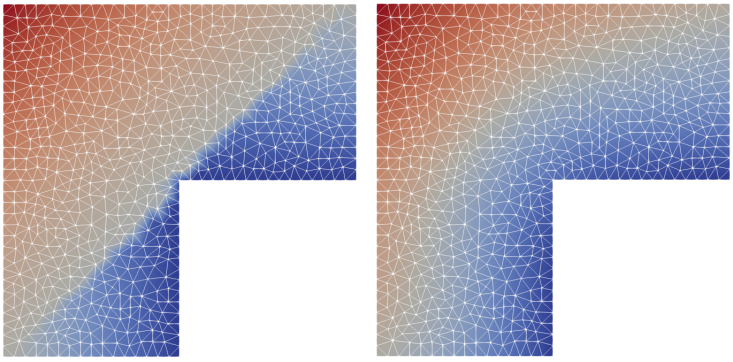


(c) DRM: random points



(d) DRM: OT quadrature points

Figure 5-3: Solution of Poisson's equation with the DGM and DRM with uniformly sampled and OT based collocation/quadrature points for $N = 833$ and $w_b = 500$.



(a) DGM: random points

(b) DGM: OT quadrature points

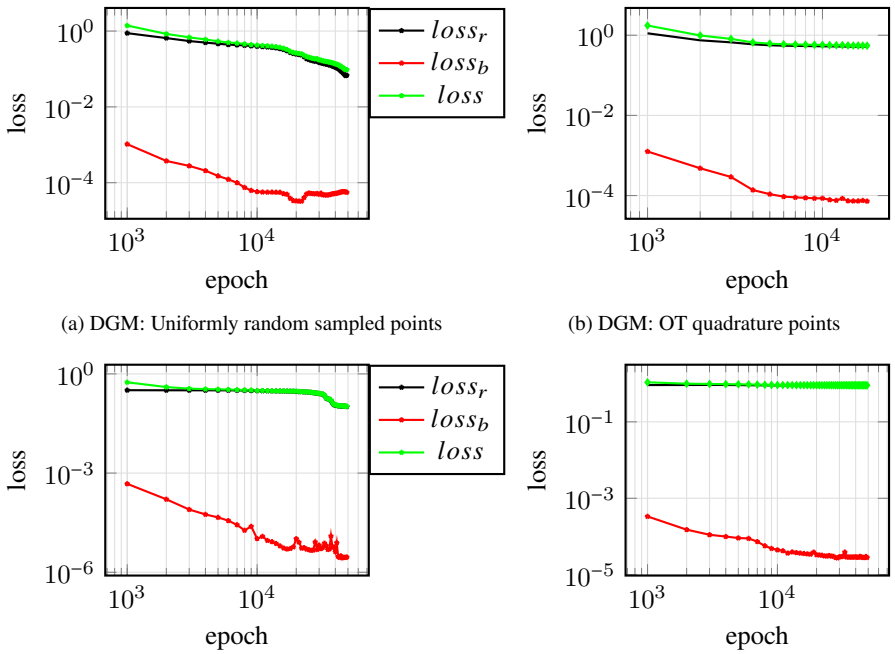
(c) DRM: random points

(d) DRM: OT quadrature points

Figure 5-4: Solution of Poisson's equation with random collocation points and OT quadrature points evaluated on the Delaunay mesh to compute the L^2 error for $w_b = 500$.

5.3.2 Loss function

In this section, we show how the loss of the DGM 5.2.7 and of the DRM 5.2.9 decreases as a function of the epochs. We evaluate loss integrals by using the MSE as in equation (5.2.12) with fixed randomly sampled points and the trapezoidal quadrature rule over the OT mesh. The locations of the randomly sampled and quadrature points are displayed in Figure 5-3.



(a) DGM: Uniformly random sampled points

(b) DGM: OT quadrature points

(c) DRM: Uniformly random sampled points

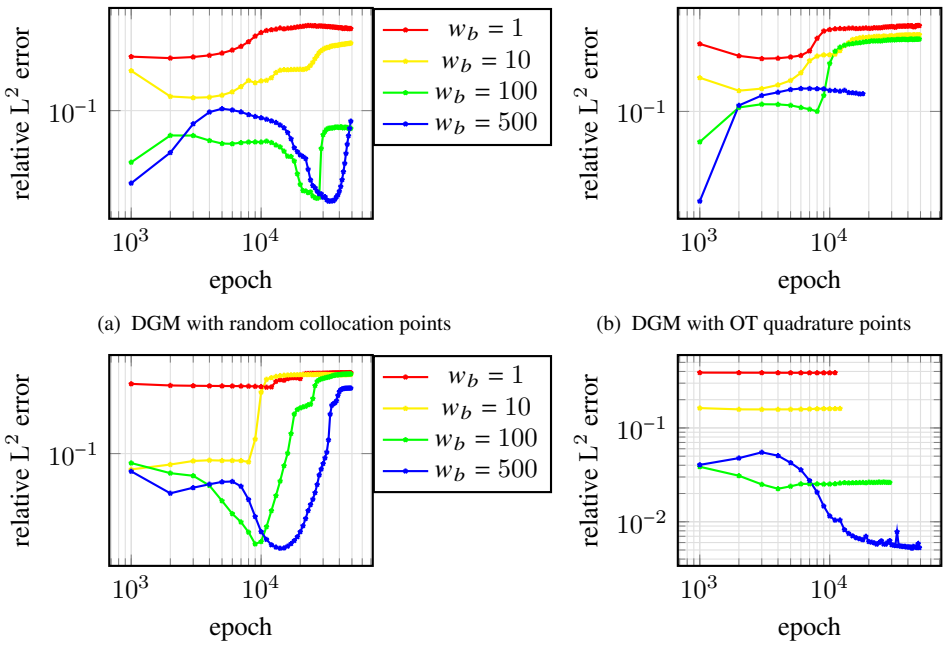
(d) DRM: OT quadrature points

Figure 5-5: Loss function given as sum of residual loss ($loss_r$) and boundary loss ($w_b \times loss_b$) for the DGM and DRM. The losses are evaluated on uniformly-sampled and OT based quadrature points for $N = 833$ and $w_b = 500$.

We note that the boundary loss ($loss_b$) decreases more steadily than the internal loss ($loss_r$), with the latter contributing most to the total loss ($loss$), The random collocation points make the loss decrease faster than using the quadrature rule for both the DGM and the DRM. Concerning those two methods, we observe that the DGM loss keeps decreasing monotonically over all the training periods, whereas the DRM stagnates towards the end of the training epochs, denoting that a minimum has been reached. In general, the final loss value computed for the DGM is lower than the DRM loss value. We will see in the next section whether this also leads to a lower L^2 error for the DGM.

5.3.3 Error analysis

In this section, we compute the relative L^2 error, defined in (5.3.3), for the DGM and the DRM. We compare the results using random collocations points and quadrature points, as done in the previous section. Furthermore, we will discuss the relationships between the losses in Figure 5-5 and the corresponding errors.



(a) DGM with random collocation points

(b) DGM with OT quadrature points

(c) DRM with random collocation points

(d) DRM with OT quadrature points

Figure 5-6: Relative L^2 error for the DGM and DRM with uniformly-sampled and OT quadrature points for $N = 833$.

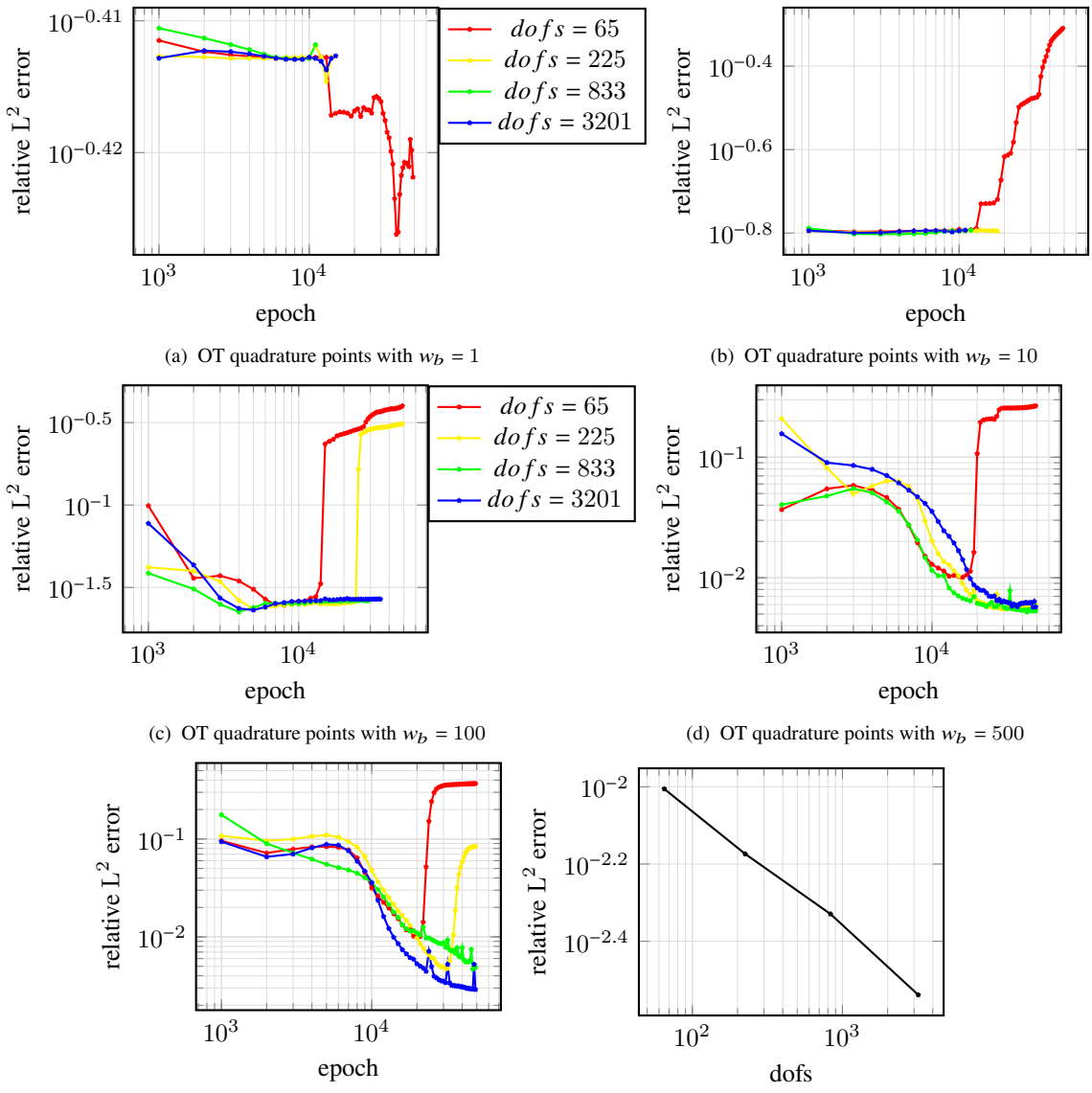
We evidence that with the DGM the error reaches a minimum after about 30000 epochs for $w_b \in \{100, 500\}$, but increases up to 10^{-1} at the end of the training process (Figure 5-6a). The choice of OT quadrature points and other values of w_b lead to a worse performance, as the relative error trend looks monotonically increasing (Figure 5-6b).

Concerning the DRM, we can distinguish clearly the error trend between collocation and OT quadrature points. The latter yields a slight not-exponential decreasing trend for all values of w_b, except when $w_b = 500$ (Figure 5-6b). In this case, the error is monotonically decreasing and stagnates towards the end of the training process, with lowest values compared to the DGM.

This suggests that decrease of the loss value does not necessarily relate to an accurate approximate solution.

5.3.4 Convergence rate analysis for DRM

As observed in the previous section, the DRM coupled with OT quadrature points is able to yield an accurate network output with decreasing relative L^2 error (Figure 5-6). Under this configuration, we compare the error for different degrees of freedom ($dofs$) under different values of w_b. Finally, we will compute the convergence rate for the best w_b.



(a) OT quadrature points with $w_b = 1$

(b) OT quadrature points with $w_b = 10$

(c) OT quadrature points with $w_b = 100$

(d) OT quadrature points with $w_b = 500$

(e) OT quadrature points with $w_b = 1000$

(f) Error rate with $w_b = 1000$

Figure 5-7: Relative L^2 error for DRM with OT quadrature points.

5.3.5 Comparison with SIP-dG method

Here, we compare the L^2 error between the SIP-dG method described in §4.2.2 and the output of the neural network trained and evaluated on the OT-mesh with $w_b = 1000$.

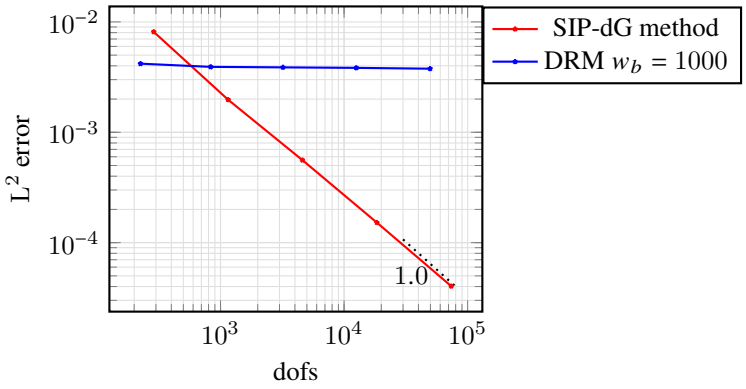


Figure 5-8: L^2 error comparison between SIP-dG solution and DGM on OT mesh.

In Figure 5-8 we observe that the SIP-dG method is only less accurate than the DRM up to 500 mesh points. The convergence rate of the SIP-dG method is optimal as shown in Chapter 4, whereas it is almost zero for the DNN method. This is not surprising, as the former method has been devised specifically for solutions with singular behaviour [ZS02]. The neural network attempts to minimise the energy loss functional of the Poisson problem, but the optimised parameters are not able to yield a solution for higher-resolution meshes.

We performed a large number of numerical experiments where we tested different scaling factors of w_b for increasing *dofs*. These show the same outcome as in Figure 5-8.

We conclude from this result that PINNs might generally outperform standard numerical solvers for linear PDEs, whose solution is sufficiently regular, but are not able to increase the resolution of the network output with more training points. A careful tuning of the parameters is required for increasing *dofs*.

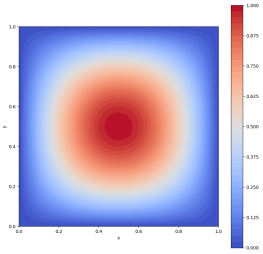
The issue related to the *convergence rate* has not been addressed in the aforementioned cited papers and is worth of further research, as this concept is essential to conduct any type of error analysis, as discussed in §5.3.3 of Chapter 2.

5.4 Numerical Results - 2

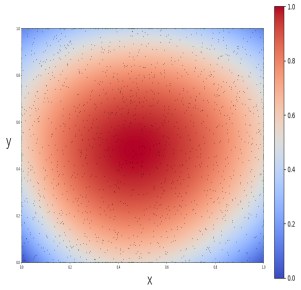
In this section, we test the DGM and the DRM with a two-dimensional Poisson problem on a squared domain:

$$\begin{aligned} -\Delta u(x,y) &= f(x,y) \text{ in } \Omega, \\ u_D(x,y) &= 0 \text{ on } \partial\Omega, \end{aligned} \tag{5.4.1}$$

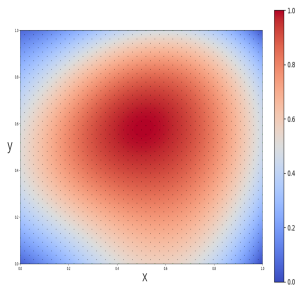
where $\Omega = [0,1] \times [0,1]$ and $f(x,y) = 2\pi^2 sin(\pi x) sin(\pi y)$, as it makes for the simple solution $u(x,y) = sin(\pi x) sin(\pi y)$.



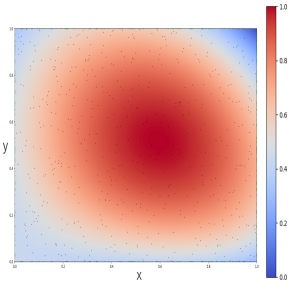
(a) Exact solution



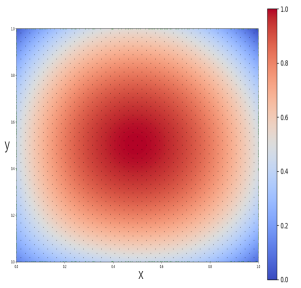
(b) DGM: random points



(c) DGM: regularly spaced points



(d) DRM: random points

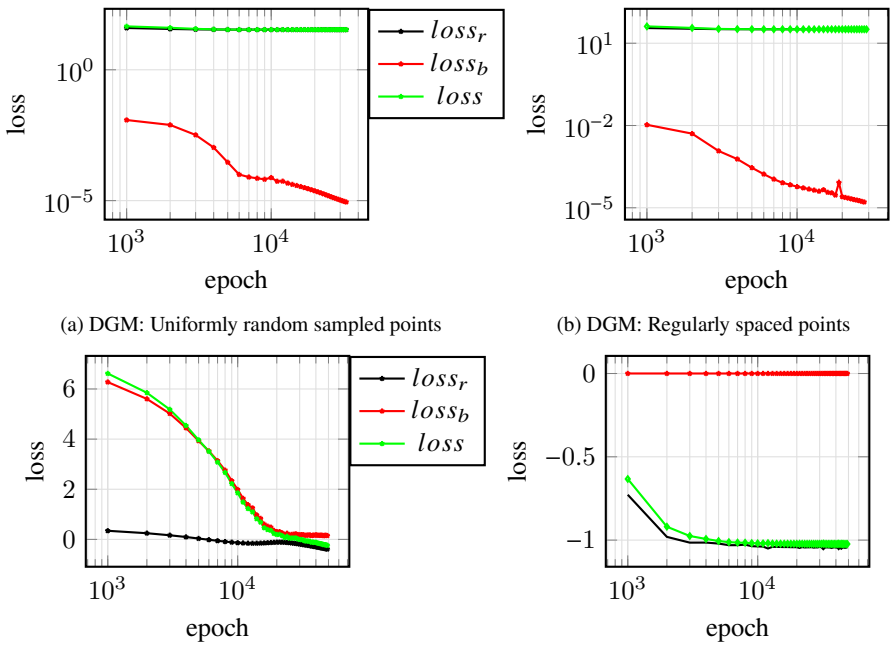


(e) DRM: regularly spaced points

Figure 5-9: Solution of Poisson's equation for the DGM and the DRM with uniformly sampled and regularly spaced collocation/quadrature points with $N = 784$ and $w_b = 500$.

Figure 5-9 shows that both the DGM and the DRM are not able to capture the radial symmetry of the exact solution. We conjecture that the solutions in Figure 5-9b–5-9d are not endowed with such property because the accuracy is locally improved only over the collocation/quadrature points, but the evaluation over the entire domain depends on the quality of the trained model. In Figure 5-9e, the uniform distribution allows for a better training, and so for a more accurate evaluation on new test points.

We proceed by plotting the DGM and the DRM loss functions for fixed *dofs* to inspect the dependence of the relative L^2 error on the penalty parameter w_b.



(a) DGM: Uniformly random sampled points

(b) DGM: Regularly spaced points

(c) DRM: Uniformly random sampled points

(d) DRM: Regularly spaced points

Figure 5-10: Loss function given as sum of residual loss ($loss_r$) and boundary loss ($w_b \times loss_b$) for the DGM and the DRM. The losses are evaluated on randomly-sampled points and on a regular grid for $N = 784$ and $w_b = 500$.

Concerning the DGM, the total loss is mainly dependent on the internal loss (Figure 5-10a–5-10b), and it is not affected by the decrease of the boundary loss. On the contrary, the DRM total loss for regularly spaced points is decreasing at the same rate as the internal loss, while the boundary loss does not change significantly.

Since the DRM is based on the Euler-Lagrange argument, the minimum of the loss function is achieved when

$$\begin{aligned} \Delta u + f &= 0 \ in \ \Omega, \\ u &= 0 \ on \ \partial\Omega. \end{aligned} \tag{5.4.2}$$

Let $\mathcal{I}(u) = \int_\Omega \left[\frac{1}{2}|\nabla u(\boldsymbol{x})|^2 - f(\boldsymbol{x}) \cdot u(\boldsymbol{x})\right] \mathrm{d}\boldsymbol{x} + \int_{\partial\Omega} u(\boldsymbol{x})^2 \, \mathrm{d}S(\boldsymbol{x})$. The minimisation of this loss and the enforcement of the boundary conditions leads to

$$\begin{aligned} \mathcal{I}(u) &= \int_\Omega \left[\frac{1}{2}\nabla u \cdot \nabla u + \frac{1}{2}u\Delta u - \frac{1}{2}u\Delta u - fu\right] \mathrm{d}\boldsymbol{x} \\ &= -\int_\Omega \left[\frac{u\Delta u}{2} + fu\right] \mathrm{d}\boldsymbol{x} \\ &= -\int_\Omega u\left[f + \frac{\Delta u}{2}\right] \mathrm{d}\boldsymbol{x} = -\frac{1}{2}\int_\Omega uf \, \mathrm{d}\boldsymbol{x}, \end{aligned} \tag{5.4.3}$$

where the first two terms in the first row integrate to zero. This follows from the divergence theorem and the fact that $u = 0$ on the boundary.

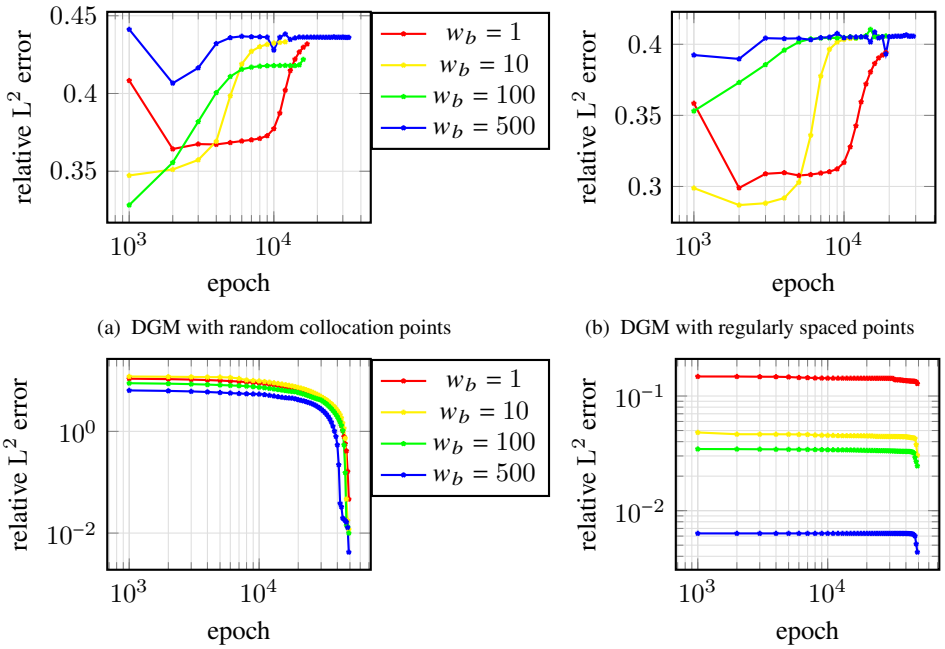
So if $u = sin(\pi x)sin(\pi y)$ and $f = 2\pi^2 sin(\pi x)sin(\pi y)$ we obtain

$$\mathcal{I}(u) = -\pi^2 \int_\Omega sin^2(\pi x)sin^2(\pi y) \, \mathrm{d}\boldsymbol{x} = -\frac{\pi^2}{4}. \tag{5.4.4}$$

Since the neural network output has enough regularity due to the activation function and the skip connection, we can assume that this argument holds for our experimental settings. The loss value at the end of the DRM training for regularly spaced points is close to -1.36, whereas the value is close to 0 for randomly sampled points. This explains why the former method yields a more accurate solution.

Finally, We remind that in Figure 5-5c–5-5d the internal loss is always positive as $f = 0$, so $\min \mathcal{I}(u) = 0$.

We plot below the relative L^2 errors for fixed $dofs$ and analyse them as done in §5.3.3.



(a) DGM with random collocation points

(b) DGM with regularly spaced points

(c) DRM with random collocation points

(d) DRM with regularly spaced points

Figure 5-11: Relative L^2 error for the DGM and the DRM with uniformly-sampled and regularly spaced quadrature points for $N = 784$.

We observe that the DRM is able to reduce the relative error as w_b increases (Figure 5-11c–5-11d). In Figure 5-11a–5-11b, the DGM does not yield satisfactory results and the relative errors converge to the same value, which are even higher compared to the first example (Figure 5-6). This might be explained by the fact that the DGM attempts to reduce (unsuccessfully) the boundary loss (Figure 5-10a–5-10b), but not the interior loss.

Finally, we compute the convergence rate for the DRM using regularly spaced training points.

In Figure 5-12, the order of convergence does not exceed 0.7, which is by far lower than the expected convergence rate obtained with DG linear elements. As a result, we conclude that the neural network framework may compete with the FEM only with a careful tuning of the configuration settings for increasing $dofs$. Furthermore, the L^2 error should be weighted by the CPU runtime required to solve the problem by using either the FEM or a neural network.

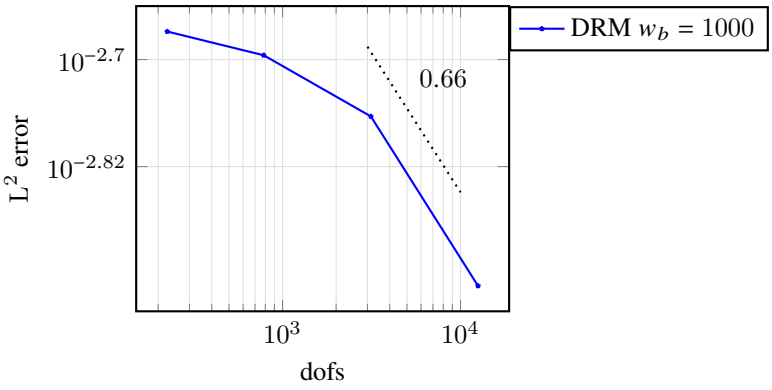


Figure 5-12: Convergence rate of L^2 error using the DRM with regularly spaced points.

5.5 Summary

In this Chapter, we have trained a block-based deep neural network to solve Poisson's equation on a L-shaped domain with two methods. The deep Galerkin method (DGM) reflects the Galerkin method in finite element to define the loss function as the integral of the PDE residual over the problem domain. The deep Ritz method (DRM) is based on the Ritz method and defines a loss energy functional to be minimised. In contrast to the first method, the DRM requires a less regular function approximation for the network output. The discretisation of the loss functions is usually conducted by randomly sampling collocation points inside the domain and at the boundary. Alternatively, the integral defining the loss function can be approximated over a mesh. The OT mesh constructed in §4.4.5-Chapter 4 has been used for both DGM and DRM in the numerical experiments, given the excellent properties in terms of quality measure (Theorem 4.14-Chapter 4) and convergence rate. The relative L^2 error has been computed over a Delaunay triangulation. The numerical results show that the coupling of DRM and OT based collocation points yields the most accurate solution. This suggests that the choice of the collocation points and the loss function is crucial for a stable and efficient training of the network parameters.

For the DRM, we have observed how the increment of the penalty parameter w_b in equation (5.3.2) improves the accuracy of the solution. However, the training of the network over the OT mesh with high resolution does not provide the desired convergence rate as for the SIP-dG method treated in Chapter 4. The stagnation of the error might be due to the fact that for high-resolution meshes, the network architecture might require more hidden units, layers, and a proper scaling of other meta-parameters, such as the learning rate (which might be adaptive and decrease as a function of epochs) and the penalty parameter w_b. A simpler Poisson problem does not show significant improvements, except for a higher convergence rate with the DRM and regularly spaced quadrature points.

Chapter 6

Conclusion

In this thesis we analysed the application of adaptive mesh strategies for the solution of hyperbolic and elliptic partial differential equations (PDEs). In Chapter 2 we first provided a general overview of adaptive mesh techniques and their applications. We mentioned moving mesh PDEs (MMPDEs) constructed from variational principles, and an Optimal Transport strategy, which has been then used in Chapter 4 to drive the mesh adaptation. In terms of h-refinement, we illustrated how the mesh can be refined/coarsened based on some criteria, such as with the use of an a-posteriori error estimator based on the Finite Element (FE) method. An example of such estimator has been shown in §5.3.3. We expanded the topic of FE by introducing the Ritz and Galerkin method and by providing consistency and well-posedness of a conforming FE discretisation using the Lax-Milgram Theorem. This has been then used to derive convergence rates in §5.3.3. We also dedicated this Chapter to discuss the mesh properties and quality measures in the context of h- and r-adaptive strategies. Finally, we introduced the deep learning framework to solve PDEs by training neural networks. We stated different approximation theorems and discussed the limitations of those.

In Chapter 3 we solved the linear advection equation by employing Winslow's moving mesh PDE as r-adaptive strategy. The physical PDE has been discretised with an upwinding discontinuous Galerkin (dG) method, while the MMPDE has been discretised with the finite element method with Lagrangian elements. The two equations have then been solved in an alternative way with an intermediary step, consisting of a data transfer operator by means of a *supermesh*.

In Chapter 4 we addressed Poisson's equation on non-convex domains. We discretised the problem with the symmetric-interior-penalty dG (SIP-dG) method. The main results are the derivation of an a posteriori estimator in the L^2 norm and the construction of an Optimal Transport (OT) mesh for r-adaptivity. Numerical experiments evidence that both methods yield optimal convergence rates in L^2 and L^∞ norm. Furthermore, exploiting the radial symmetry of the solution near the re-entrant corner, we have proved that the quality measure of the OT mesh elements are independent on their location and number of total vertices. Finally, we shown a close link between the two adaptive strategies by providing

numerical evidence that the local a-posteriori estimates used for h-refinement are equidistributed on the OT mesh.

In Chapter 5 we solved the same problem by using the neural network framework, aiming to minimise two loss functions, which resemble the standard Galerkin and Ritz finite element method introduced in §2.3.3 of Chapter 2. The main contribution is related to the location of the training points used to train the network. We used the vertices of the OT mesh derived in Chapter 4 to show that the network output features higher accuracy than a network with training points chosen uniformly at random. Also, the convex energy functional of the deep Ritz formulation guarantees a more stable optimisation procedure for the network parameters compared to the deep Galerkin formulation. However, the final numerical experiments shows that the network output is more accurate the SIP-dG method for low resolution, but is unable to increase the accuracy of the approximate solution for higher resolution unless a different configuration is employed. This is a major drawback that hinders the error analysis for the deep learning framework.

In conclusion, we present several directions of future research for each Chapter presented in this thesis.

1. In Chapter 3 we provided a novel approach to ensure mass conservation for a linear advection equation. This is not guaranteed in the standard rezoning approach as requires an interpolation step, which adds artificial diffusion to the equation. Our numerical results show that the mass is conserved at the cost of solving a quasilinear parabolic PDE at each time step. Possible ideas for expanding this work will examine:

   (a) The reduction in computational time of the 'inner loop' moving mesh strategy to achieve comparable accuracy with the (Algorithm 4).

   (b) Scalability of the algorithm through parallelisation.

   (c) The applicability of the method to more physical relevant problems, such as the shallow water equations on the sphere [Wel+16]. These require tailored methods with good conservative properties for long time simulations. In particular, for global atmospheric flows, the ability of massive parallelisation is crucial.

2. In Chapter 4 we solved Poisson's equation using the SIP-dG method and applied r- and h-adaptivity. We derived an L^2 a-posteriori error estimate used for the latter strategy. The best r-adaptive mesh in terms of accuracy and quality measures was obtained with an OT strategy. We evidenced the optimal convergence for both methods and that the a-posteriori estimate is equidistributed on the OT mesh in the area close to the re-entrant corner. In the view of the above considerations, possible future research directions are:

   (a) Application of the numerical scheme on polygons with multiple re-entrant corners.

   (b) Derivation of a L^2 a-posteriori estimate with time-dependency for relevant problems arising in physics, such as the shallow water equations.

(c) Analysis of the relationship between different mesh quality measures, such as skewness and shape regularity.

(d) Scalability of the OT generation algorithm through parallelisation.

(e) Theoretical proof of the equidistribution of the a-posteriori estimator on the OT mesh.

3. In Chapter 5 we solved Poisson's equation on the L-shaped domain by using the deep learning framework. We shown that the deep Ritz method with energy loss functional approximated with quadrature rule over the OT mesh vertices derived in 4 yields the most accurate solution. In contrast, the deep Galerkin method and uniformly random sampled points is not able to approximate the target function with singular behaviour. We concluded the Chapter by highlighting that the convergence rate of the neural network is not comparable to the finite element method and requires a proper scaling of network parameters. There are several open questions that are worth further investigation:

   (a) Applicability of the deep Ritz method to solve time-dependent PDEs.

   (b) Parallelisation of the training process by domain decomposition [KZK21].

   (c) Analysis of the relationship between convergence rate and scaling network parameters, such as number of hidden units, layers and other meta-parameters.

   (d) Implementation of the rezoning approach to equidistribute the mesh (possibly based on a MMPDE or OT based strategy) and to solve a PDE alternatively under the deep learning framework.

Appendix A

Derivation of the optimal parameters for OT-based mesh generation

In this appendix we prove the results stated in Lemma 4.13.

A.0.1 L^∞ norm

Firstly, we derive the optimal value of the exponent γ that equidistributes the L^∞ linear interpolation error norm $\left\|u - p_{1,N}\right\|_\infty$. We consider the function $u(r) = r^\alpha$, where $\alpha = \pi/\omega$, and ω is the interior angle of the re-entrant corner. As $\alpha < 1$ and $u \notin C^2([0,1])$, we split the error analysis into two parts. We split the interval $[0,1]$ into the partition $\{r_j\}_{j=0}^N$, with $r_j = \left(\frac{j}{N}\right)^\beta$ for $j = 0, \ldots, N$, and define the linear interpolant of u on those points by $p_{1,N}$. The prescribed parameter β specifies the level of grid compression towards the origin. We can then write

$$\left\|u - p_{1,N}\right\|_{\infty,[0,1]} = \max\left\{\left\|u - p_{1,N}\right\|_{\infty,[0,r_1]}, \left\|u - p_{1,N}\right\|_{\infty,[r_1,r_N]}\right\}.$$

For the first part, by the triangle inequality

$$\left\|u - p_{1,N}\right\|_{\infty,[0,r_1]} \leq \|u\|_{\infty,[0,r_1]} + \left\|p_{1,N}\right\|_{\infty,[0,r_1]} = r_1^\alpha + p_{1,N}(r_1) = 2\left(\left(\frac{1}{N}\right)^\beta\right)^\alpha = 2N^{-\alpha\beta}, \quad \text{(A.0.1)}$$

using the fact that both u and $p_{1,N}$ are increasing functions. We treat each interval $[r_{j-1}, r_j]$ for $j \geq 2$ separately. For $h_j = r_j - r_{j-1}$, we have

$$\left\|u - p_{1,N}\right\|_{\infty,[r_{j-1},r_j]} \leq \frac{1}{8} h_j^2 \left\|u''\right\|_{\infty,[r_{j-1},r_j]},$$

with

$$\left\|u''\right\|_{\infty,[r_{j-1},r_j]} = \sup_{[r_{j-1},r_j]} |u''| = |\alpha(\alpha-1)| r_{j-1}^{\alpha-2} = |\alpha(\alpha-1)| \left(\frac{j-1}{N}\right)^{\beta(\alpha-2)}.$$

Using the mean-value theorem we have $h_j = \frac{1}{N}\beta\left(\frac{t_j}{N}\right)^{\beta-1} \leq \frac{\beta}{N}\left(\frac{j}{N}\right)^{\beta-1}$ for any $t_j \in (j-1, j)$.

We then obtain

$$\left\|u - p_{1,N}\right\|_{\infty,[r_{j-1},r_j]} \leq \frac{|\alpha(\alpha-1)|}{8} \beta^2 N^{-\alpha\beta} \left(\frac{j^{2(\beta-1)}}{(j-1)^{\beta(2-\alpha)}}\right). \tag{A.0.2}$$

Finally, equations (A.0.1) and (A.0.2) give

$$\left\|u - p_{1,N}\right\|_{\infty,[0,1]} \leq \max_{j=1,\ldots,N} \{\left\|u - p_{1,N}\right\|_{\infty,[r_{j-1},r_j]}\} \leq N^{-\alpha\beta} \max_{j=2,\ldots,N} \left\{2, \frac{|\alpha(\alpha-1)|}{8} \beta^2 \frac{j^{2(\beta-1)}}{(j-1)^{\beta(2-\alpha)}}\right\}.$$

By imposing $2(\beta-1) = \beta(2-\alpha) \Rightarrow \beta = 2/\alpha$ we obtain that the order of convergence for the L^∞ norm is N^{-2}.

If s is the computational variable in Ω_c then the L^∞ norm is equidistributed with

$$r = s^\beta$$

so that $\mathrm{d}r = \beta s^{\beta-1}\,\mathrm{d}s$. Hence we have $m(r) s^\beta \beta s^{\beta-1}\,\mathrm{d}s = s\,\mathrm{d}s \Rightarrow m(r) = 1/\beta(r^{1/\beta})^{2(1-\beta)} = cr^{-2\gamma}$. Thus

$$\gamma = 1 - \frac{1}{\beta} = 1 - \frac{\alpha}{2} = 1 - \frac{\pi}{2\omega}. \tag{A.0.3}$$

For the L-shaped domain ($\omega = 3\pi/2$) we have $\gamma = 2/3$, whereas for the crack domain ($\omega = 2\pi$) we have $\gamma = 3/4$. The results derived are in agreement with both Fig. 4-10b and Fig. 4-18b in §4.5.

A.0.2 L^2 norm

We consider the image K in the physical domain Ω of a uniform triangle $\widehat{K}$ in the computational domain Ω_c, under the action of the OT map. It follows from the boundedness of the shape regularity of the map as $r \sim 0$ (Figure 4-7a–4-7b) that K has no vanishingly small angles and is close to being a regular triangle (as can be seen from the images of the generated meshes). If the original triangle $\widehat{K}$ has cell diameter Δ then, from (4.4.11) it follows that the sides h_K of the transformed triangle K are proportional to $h_K = \Delta r^\gamma$. Now consider a function of the form $u(r,\theta) = r^\alpha f(\theta)$, where $\alpha = \pi/\omega < 1$ and $f(\theta)$ is slowly varying. In this function, all r-partial derivatives will dominate θ-partial derivatives as $r \to 0$. It follows from standard results that the L^2 interpolation error $E_{2,K}$ over K is bounded by:

$$E_{2,K} < Ch_K^2 |u|_{\mathrm{H}^2(K)} \approx C\left(\Delta^2 r^{2\gamma}\right)\Delta r^\gamma \max|u_{rr}| = C\Delta^3 r^{3\gamma} r^{\pi/\omega-2} = Cr^{3\gamma+\pi/\omega-2},$$

where C is a constant independent of r. As in the case of the L^∞ norm we need to consider the two cases of when $r = 0$ lies in K or not separately.

If K does not include the point $r = 0$ then $E_{2,K}$ is equidistributed when $r^{3\gamma} r^{\pi/\omega-2}$ is constant.

Hence in this case

$$\gamma = \frac{2}{3} - \frac{\pi}{3\omega}. \tag{A.0.4}$$

If K does include the point $r = 0$ then we have by the triangle inequality

$$E_{2,K} < \left\|p_{1,N}\right\|_{\mathrm{L}^2(K)} + \|u\|_{\mathrm{L}^2(K)}.$$

For the first term we have

$$\left\|p_{1,N}\right\|_{\mathrm{L}^2(K)} \sim \left(\int_0^{r_1}\int_0^{\theta_1} |p_{1,N}|^2 r\,\mathrm{d}\theta\,\mathrm{d}r\right)^{1/2} < \left(C_0(\theta) r_1^2\left(\int_0^{r_1} r\,\mathrm{d}r\right)\right)^{1/2} < C_1(\theta) r_1^2,$$

where $C_0(\theta), C_1(\theta)$ are two constants dependent on θ. Similarly, the second term gives the bound

$$\|u\|_{\mathrm{L}^2(K)} < \left(C_0(\theta)\int_0^{r_1} |r^\alpha|^2 r\,\mathrm{d}r\right)^{1/2} < C_1(\theta) r_1^\alpha r_1 < C_1(\theta) r_1^{1+\pi/\omega}.$$

We first note that the term $\left\|p_{1,N}\right\|_{\mathrm{L}^2(K)} \to 0$ faster than the second term. We focus then on the

predominant part of the error, which is dependent on $r_1^{1+\pi/\omega}$.

Provided $\gamma < 1$, the following inequality holds:

$$1 + \pi/\omega > 3\gamma + \pi/\omega - 2$$

Thus, if the error is equidistributed over those elements not including the origin, it will be smaller on the elements including the origin.

Hence we can take γ as in (A.0.4) so that if $\omega = 3\pi/2$ (L-shaped domain) then $\gamma = 4/9$ and if $\omega = 2\pi$ (crack domain) then $\gamma = 1/2$. The results derived are consistent with both Fig. 4-10a and Fig. 4-18a in §4.5.

Appendix B

Code

The code used to obtain the numerical results presented throughout the thesis are available on GitHub:

- Chapter 3
- Chapter 4
- Chapter 5

Bibliography

[AO97] M. Ainsworth and J.T. Oden. "A posteriori error estimation in finite element analysis". In: *Computer Methods in Applied Mechanics and Engineering* 142.1 (1997), pp. 1–88. ISSN: 0045-7825. DOI: https://doi.org/10.1016/S0045-7825(96)01107-3. URL: https://www.sciencedirect.com/science/article/pii/S0045782596011073.

[Ape13] T. Apel. *Advanced Finite Element Methods and Applications*. Ed. by O. Steinbach. 1st ed. 2013. Lecture Notes in Applied and Computational Mechanics, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, p. 376.

[Arn+00] D. Arnold et al. "Discontinuous Galerkin Methods". In: vol. 11. Lecture Notes in Computational Science and Engineering. Springer Berlin, Heidelberg, Jan. 2000, pp. 89–101. ISBN: 978-3-642-64098-8. DOI: 10.1007/978-3-642-59721-3.

[Arn+02] D. Arnold et al. "Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems". In: *SIAM J. Numer. Anal.* 39 (Jan. 2002). DOI: 10.1137/S0036142901384162.

[Arn82] D. Arnold. "An Interior Penalty Finite Element Method with Discontinuous Elements". In: *SIAM Journal on Numerical Analysis* 19.4 (1982), pp. 742–760. DOI: 10.1137/0719052. eprint: https://doi.org/10.1137/0719052. URL: https://doi.org/10.1137/0719052.

[ARW07] T. Apel, A. Rösch, and G. Winkler. "Optimal control in non-convex domains: A priori discretization error estimates". In: *Calcolo* 44 (Sept. 2007), pp. 137–158. DOI: 10.1007/s10092-007-0133-0.

[AS98] M. Ainsworth and B. Senior. "An adaptive refinement strategy for hp-finite element computations". In: *Applied Numerical Mathematics* 26.1 (1998), pp. 165–178.

[Bab71] I. Babuka. "The Rate of Convergence for the Finite Element Method". In: *SIAM Journal on Numerical Analysis* 8.2 (1971), pp. 304–315. DOI: 10.1137/0708031. eprint: https://doi.org/10.1137/0708031. URL: https://doi.org/10.1137/0708031.

[Bak97] T.J. Baker. "Mesh adaptation strategies for problems in fluid dynamics". In: *Finite Elements in Analysis and Design* 25.3 (1997). Adaptive Meshing, Part 2, pp. 243–273. ISSN: 0168-874X. DOI: https://doi.org/10.1016/S0168-874X(96)00032-7. URL: https://www.sciencedirect.com/science/article/pii/S0168874X96000327.

[Bay+18] A. Baydin et al. "Automatic differentiation in machine learning: A survey". In: *Journal of Machine Learning Research* 18 (Apr. 2018), pp. 1–43.

[BBO03] I. Babuska, U. Banerjee, and J. E. Osborn. "Survey of meshless and generalized finite element methods: A unified approach". In: *Acta Numerica* 12 (2003), pp. 1–125.

[BBO99] I. Babuka, C.E. Baumann, and J.T. Oden. “A discontinuous hp finite element method for diffusion problems: 1-D analysis”. In: *Computers & Mathematics with Applications* 37.9 (1999), pp. 103–122. ISSN: 0898-1221. DOI: https://doi.org/10.1016/S0898-1221(99)00117-0. URL: https://www.sciencedirect.com/science/article/pii/S0898122199001170.

[BCW13] C.J. Budd, M.J.P. Cullen, and E.J. Walsh. “MongeAmpère based moving mesh methods for numerical weather prediction, with applications to the Eady problem”. In: *Journal of Computational Physics* 236 (2013), pp. 247–270. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2012.11.014. URL: https://www.sciencedirect.com/science/article/pii/S0021999112006912.

[BG88] I. Babuka and B. Q. Guo. “Regularity of the Solution of Elliptic Problems with Piecewise Analytic Data. Part I. Boundary Value Problems for Linear Elliptic Equation of Second Order”. In: *SIAM Journal on Mathematical Analysis* 19.1 (1988), pp. 172–203. DOI: 10.1137/0519014. eprint: https://doi.org/10.1137/0519014. URL: https://doi.org/10.1137/0519014.

[BG89] I. Babuka and B. Q. Guo. “Regularity of the Solution of Elliptic Problems with Piecewise Analytic Data. II: The Trace Spaces and Application to the Boundary Value Problems with Nonhomogeneous Boundary Conditions”. In: *SIAM Journal on Mathematical Analysis* 20.4 (1989), pp. 763–781. DOI: 10.1137/0520054. eprint: https://doi.org/10.1137/0520054. URL: https://doi.org/10.1137/0520054.

[BG92] I. Babuka and B.Q. Guo. “The h, p and hp- version of the finite element method; basis theory and applications”. In: *Advances in Engineering Software* 15.3 (1992), pp. 159–174. ISSN: 0965-9978. DOI: https://doi.org/10.1016/0965-9978(92)90097-Y. URL: http://www.sciencedirect.com/science/article/pii/096599789290097Y.

[BH74] J.R. Bunch and J.E. Hopcroft. “Triangular factorization and inversion by fast matrix multiplication”. In: *Math. Comp.* 28 (1974), pp. 231–236.

[BHR09] C.J. Budd, W. Huang, and R.D. Russell. “Adaptivity with moving grids”. In: *Acta Numerica* 18 (2009), pp. 111–241.

[BHR96] C.J. Budd, W. Huang, and R.D. Russell. “Moving Mesh Methods for Problems with Blow-Up”. In: *SIAM Journal on Scientific Computing* 17 (1996), pp. 305–327.

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition*. USA: Oxford University Press, Inc., 1995. ISBN: 0198538642.

[BKP79] I. Babuka, R. B. Kellogg, and J. Pitkäranta. “Direct and Inverse Error Estimates for Finite Elements with Mesh Refinements”. In: *Numer. Math.* 33.4 (1979), pp. 447–471. ISSN: 0029-599X. DOI: 10.1007/BF01399326. URL: https://doi.org/10.1007/BF01399326.

[Boo73] Carl de Boor. “Spline Functions and Approximation Theory”. In: ed. by A. Meir and A. Sharma. International Series of Numerical Mathematics. Birkhäuser Basel, 1973, pp. 57–72. ISBN: 978-3-0348-5979-0. DOI: 10.1007/978-3-0348-5979-0_3. URL: https://doi.org/10.1007/978-3-0348-5979-0_3.

[BR14] C. Bedregal and M. Rivara. “Longest-edge algorithms for size-optimal refinement of triangulations”. In: *Computer-Aided Design* 46 (Jan. 2014), pp. 246–251. DOI: 10.1016/j.cad.2013.08.040.

[BR72] I. Babuka and M. B. Rosenzweig. “A Finite Element Scheme for Domains with Corners”. In: *Numer. Math.* 20.1 (1972), pp. 1–21. ISSN: 0029-599X. DOI: 10.1007/BF01436639. URL: https://doi.org/10.1007/BF01436639.

[Bre91] Y. Brenier. "Polar Factorization and Monotone Rearrangement of Vector-Valued Functions". In: *Communications on Pure and Applied Mathematics* 44 (1991), pp. 375–417.

[BRW15] C.J. Budd, R.D. Russell, and E. Walsh. "The geometry of r-adaptive meshes generated using optimal transport methods". In: *Journal of Computational Physics* 282 (2015), pp. 113–137. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2014.11.007. URL: https://www.sciencedirect.com/science/article/pii/S0021999114007591.

[BS08] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Vol. 15. Texts in Applied Mathematics. Springer, 2008. ISBN: 9780387759333. DOI: 10.1007/978-0-387-75934-0. URL: http://dx.doi.org/10.1007/978-0-387-75934-0.

[BW06] C.J. Budd and J.F. Williams. "Parabolic MongeAmpère methods for blow-up problems in several spatial dimensions". In: *Journal of Physics A: Mathematical and General* 39 (Apr. 2006), p. 5425. DOI: 10.1088/0305-4470/39/19/S06.

[BW09] C.J. Budd and J.F. Williams. "Moving Mesh Generation using the Parabolic MongeAmpère Equation". In: *SIAM Journal on Scientific Computing* 31.5 (2009), pp. 3438–3465. DOI: 10.1137/080716773. eprint: https://doi.org/10.1137/080716773. URL: https://doi.org/10.1137/080716773.

[BW10] C.J. Budd and J.F. Williams. "How to adaptively resolve evolutionary singularities in differential equations with symmetry". In: *Journal of Engineering Mathematics* 66 (Mar. 2010), pp. 217–236. DOI: 10.1007/s10665-009-9343-6.

[BWG04] C.J. Budd, J. F. Williams, and V. A. Galaktionov. "Self-Similar Blow-Up in Higher-Order Semilinear Parabolic Equations". In: *SIAM Journal on Applied Mathematics* 64.5 (2004), pp. 1775–1809. DOI: 10.1137/S0036139902415 52X. eprint: https://doi.org/10.1137/S003613990241552X. URL: https://doi.org/10.1137/S003613990241552X.

[Caf90] L.A. Caffarelli. "Interior $W^{2,p}$ Estimates for Solutions of the Monge-Ampere Equation". In: *Annals of Mathematics* 131.1 (1990), pp. 135–150. ISSN: 0003486X. URL: http://www.jstor.org/stable/1971510 (visited on 06/17/2022).

[Caf96] L.A. Caffarelli. "Boundary Regularity of Maps with Convex Potentials–II". In: *Annals of Mathematics* 144.3 (1996), pp. 453–496. ISSN: 0003486X. URL: http://www.jstor.org/stable/2118564 (visited on 06/17/2022).

[CD02] Z. Chen and S. Dai. "On the Efficiency of Adaptive Finite Element Methods for Elliptic Problems with Discontinuous Coefficients". In: *Industrial and Applied Mathematics* 24 (Oct. 2002), pp. 443–462. DOI: 10.1137/S1064827501383713.

[CGH14] A. Cangiani, E. Georgoulis, and P. Houston. "hp-version discontinuous Galerkin methods on polygonal and polyhedral meshes". In: *Mathematical Models and Methods in Applied Sciences* 24.10 (2014), pp. 2009–2041.

[Che20] J. Chen. "A Comparison Study of Deep Galerkin Method and Deep Ritz Method for Elliptic Problems with Different Boundary Conditions". In: *Communications in Mathematical Research* 36 (June 2020), pp. 354–376. DOI: 10.4208/cmr.2020-0051.

[CHR99] W. Cao, W. Huang, and R. D. Russell. "An r-Adaptive Finite Element Method Based upon Moving Mesh PDEs". In: *Journal of Computational Physics* 149.2 (1999), pp. 221–244. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.1998.6151. URL: https://www.sciencedirect.com/science/article/pii/S0021999198961514.

[CKO00] L.A. Caffarelli, S. Kochengin, and V. Oliker. “On the Numerical Solution of the Problem of Reflector Design with Given Far-Field Scattering Data”. In: *Contemporary Mathematics* 226 (Dec. 2000). DOI: 10.1090/conm/226/03233.

[Coo+19] S. Cook et al. “Error estimates for semi-Lagrangian finite difference methods applied to Burgers’ equation in one dimension”. In: *Applied Numerical Mathematics* 145 (June 2019). DOI: 10.1016/j.apnum.2019.06.012.

[CRR15] B. Crestel, R.D. Russell, and S. Ruuth. “Moving Mesh Methods on Parametric Surfaces”. In: *Procedia Engineering* 124 (Dec. 2015), pp. 148–160. DOI: 10.1016/j.proeng.2015.10.129.

[Del+08] G.L. Delzanno et al. “An optimal robust equidistribution method for two-dimensional grid adaptation based on MongeKantorovich optimization”. In: *Journal of Computational Physics* 227 (Dec. 2008), pp. 9841–9864. DOI: 10.1016/j.jcp.2008.07.020.

[DF11] G. De Philippis and A. Figalli. “$W^{2,1}$ regularity for solutions of the Monge-Ampère equation”. In: *arXiv: Analysis of PDEs* (2011).

[DG12] A. Demlow and E. H. Georgoulis. “Pointwise a Posteriori Error Control for Discontinuous Galerkin Methods for Elliptic Problems”. In: *SIAM Journal on Numerical Analysis* 50.5 (2012), pp. 2159–2181. DOI: 10.1137/110846397. eprint: https://doi.org/10.1137/110846397. URL: https://doi.org/10.1137/110846397.

[DS13] J. Danczyk and K. Suresh. “Finite element analysis over tangled simplicial meshes: Theory and implementation”. In: *Finite Elements in Analysis and Design* 70 (2013), pp. 57–67.

[Dvi91] A.S. Dvinsky. “Adaptive Grid Generation from Harmonic Maps on Riemannian Manifolds”. In: *Journal of Computational Physics* 95.2 (Aug. 1991), pp. 450–476. DOI: 10.1016/0021-9991(91)90285-S.

[Eis87] P.R. Eiseman. “Adaptive grid generation”. In: *Computer Methods in Applied Mechanics and Engineering* 64.1 (1987), pp. 321–376.

[Eva10] L. C. Evans. *Partial differential equations*. American Mathematical Society, 2010. ISBN: 9780821849743 0821849743.

[FDF11] C. Fischer, A. Düster, and W. Fricke. In: *Conference: Proceedings of the 3rd International Conference on Marine Structures*. Mar. 2011, pp. 289–294. ISBN: 978-0-415-67771-4. DOI: 10.1201/b10771-34.

[Fel99] M. Feldman. “Growth of a sandpile around an obstacle”. In: *Monge Ampère Equation: Applications to Geometry and Optimization (Deerfield Beach, FL, 1997), number 226 in Contemp. Math*. American Mathematical Society, 1999, pp. 55–78.

[Fig07] A. Figalli. “Existence, Uniqueness, and Regularity of Optimal Transport Maps”. In: *SIAM Journal on Mathematical Analysis* 39.1 (2007), pp. 126–137. DOI: 10.1137/060665555. eprint: https://doi.org/10.1137/060665555. URL: https://doi.org/10.1137/060665555.

[FM10] P. Farrell and J. Maddison. “Conservative interpolation between volume meshes by local Galerkin projection”. In: *Computer Methods in Applied Mechanics and Engineering* 200 (2010), pp. 89–100.

[Gar77] A.B. Garth. “Finite element methods for elliptic equations using nonconforming elements”. In: *Mathematics of Computation* 31 (1977), pp. 45–59.

[Gav+11] N. Gavish et al. “Curvature driven flow of bi-layer interfaces”. In: *Physica D-nonlinear Phenomena - PHYSICA D* 240 (Mar. 2011), pp. 675–693. DOI: 10.1016/j.physd.2010.11.016.

[GB05] T. Grätsch and K.J. Bathe. "A posteriori error estimation techniques in practical finite element analysis". In: *Computers & Structures* 83.4 (2005), pp. 235–265. ISSN: 0045-7949. DOI: https://doi.org/10.1016/j.compstruc.2004.08.011. URL: https://www.sciencedirect.com/science/article/pii/S0045794904003165.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[GGI] S. Gottlieb, Z.J. Grant, and L. Isherwood. *Strong Stability Preserving Integrating Factor Runge-Kutta Methods*.

[GL13] K.B. Glasner and A.E. Lindsay. "The Stability and Evolution of Curved Domains Arising from One-Dimensional Localized Patterns". In: *SIAM Journal on Applied Dynamical Systems* 12.2 (2013), pp. 650–673. DOI: 10.1137/120893008. eprint: https://doi.org/10.1137/120893008. URL: https://doi.org/10.1137/120893008.

[GMP17] E. Georgoulis, C. Makridakis, and T. Pryer. "Babuka-Osborn techniques in discontinuous Galerkin methods: L^2-norm error estimates for unstructured meshes". In: *arXiv:1704.05238* (2017).

[Gri11] P. Grisvard. *Elliptic Problems in Nonsmooth Domains*. Society for Industrial and Applied Mathematics, 2011. DOI: 10.1137/1.9781611972030. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611972030. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972030.

[GW08] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Second. USA: Society for Industrial and Applied Mathematics, 2008. ISBN: 0898716594.

[Hag94] R. Hagmeijer. "Grid Adaption Based on Modified Anisotropic Diffusion Equations Formulated in the Parametric Domain". In: *Journal of Computational Physics* 115.1 (1994), pp. 169–183. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.1994.1185. URL: https://www.sciencedirect.com/science/article/pii/S0021999184711855.

[Har16] K. Hart. *AABBTree*. https://github.com/kip-hart/AABBTree. 2016.

[Her+13] C. Hertel et al. "Using a Moving Mesh PDE for Cell Centres to Adapt a Finite Volume Grid". In: *Flow, Turbulence and Combustion* 90.4 (2013), pp. 785–812.

[HK15a] W. Huang and L. Kamenski. "A geometric discretization and a simple implementation for variational mesh generation and adaptation". In: *J. Comput. Phys.* 301 (2015), pp. 322–337.

[HK15b] W. Huang and L. Kamenski. "On the mesh nonsingularity of the moving mesh PDE method". In: *Mathematics of Computation* 87 (2015).

[Hoc98] S. Hochreiter. "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6 (Apr. 1998), pp. 107–116. DOI: 10.1142/S0218488598000094.

[HR11] W. Huang and R.D. Russell. *Adaptive Moving Mesh Methods*. Vol. 174. 2011.

[HR98a] W. Huang and R. D. Russell. "A high dimensional moving mesh strategy". In: *Applied Numerical Mathematics* 26.1 (1998), pp. 63–76. ISSN: 0168-9274. DOI: https://doi.org/10.1016/S0168-9274(97)00082-2. URL: https://www.sciencedirect.com/science/article/pii/S0168927497000822.

[HR98b] W. Huang and R. D. Russell. “Moving Mesh Strategy Based on a Gradient Flow Equation for Two-Dimensional Problems”. In: *SIAM Journal on Scientific Computing* 20.3 (1998), pp. 998–1015. DOI: 10.1137/S1064827596315242. eprint: https://doi.org/10.1137/S1064827596315242. URL: https://doi.org/10.1137/S1064827596315242.

[HRR94a] W. Huang, Y. Ren, and R. D. Russell. “Moving Mesh Methods Based on Moving Mesh Partial Differential Equations”. In: *Journal of Computational Physics* 113.2 (1994), pp. 279–290. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.1994.1135. URL: https://www.sciencedirect.com/science/article/pii/S0021999184711351.

[HRR94b] W. Huang, Y. Ren, and R. D. Russell. “Moving Mesh Partial Differential Equations (MMPDES) Based on the Equidistribution Principle”. In: *SIAM Journal on Numerical Analysis* 31.3 (1994), pp. 709–730. DOI: 10.1137/0731038. eprint: https://doi.org/10.1137/0731038. URL: https://doi.org/10.1137/0731038.

[HS03] W. Huang and W. Sun. “Variational Mesh Adaptation II: Error Estimates and Monitor Functions”. In: *J. Comput. Phys.* 184.2 (2003), pp. 619–648.

[HSS02] P. Houston, C. Schwab, and E. Süli. “Discontinuous hp-Finite Element Methods for Advection-Diffusion-Reaction Problems”. In: *SIAM Journal on Numerical Analysis* 39.6 (2002), pp. 2133–2163. DOI: 10.1137/S0036142900374111. eprint: https://doi.org/10.1137/S0036142900374111. URL: https://doi.org/10.1137/S0036142900374111.

[HSW89] K. Hornik, M. Stinchcombe, and H. White. “Multilayer feedforward networks are universal approximators”. In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[Hua15] W. Huang. “A Comparative Numerical Study of Meshing Functionals for Variational Mesh Adaptation”. In: *Journal of Mathematical Study* 48.2 (2015), pp. 168–186.

[JHG18] A. Jacot, C. Hongler, and F. Gabriel. “Neural Tangent Kernel: Convergence and Generalization in Neural Networks”. In: *NeurIPS*. 2018, pp. 8580–8589. URL: http://dblp.uni-trier.de/db/conf/nips/nips2018.html#JacotHG18.

[KB14] D. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (Dec. 2014).

[KH10] K. Kwang-Yeon and L. Hyung-Chun. “A posteriori error estimators for nonconforming finite element methods of the linear elasticity problem”. In: *Journal of Computational and Applied Mathematics* 235.1 (2010), pp. 186–202. ISSN: 0377-0427. DOI: https://doi.org/10.1016/j.cam.2010.05.032. URL: https://www.sciencedirect.com/science/article/pii/S0377042710003080.

[KZK21] E. Kharazmi, Z. Zhang, and G.E. Karniadakis. “hp-VPINNs: Variational physics-informed neural networks with domain decomposition”. In: *Computer Methods in Applied Mechanics and Engineering* 374 (2021), p. 113547. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2020.113547. URL: https://www.sciencedirect.com/science/article/pii/S0045782520307325.

[Li+18] H. Li et al. “Visualizing the Loss Landscape of Neural Nets”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS’18. Montrèal, Canada: Curran Associates Inc., 2018, pp. 6391–6401.

[Liy+21] L. Liyao et al. “Enforcing Exact Boundary and Initial Conditions in the Deep Mixed Residual Method”. In: *CSIAM Transactions on Applied Mathematics* 2.4 (2021), pp. 748–775. ISSN: 2708-0579. DOI: https://doi.org/10.4208/csiam-am.SO-2021-0011. URL: http://global-sci.org/intro/article_detail/csiam-am/19991.html.

[LLF98] I.E. Lagaris, A. Likas, and D.I. Fotiadis. “Artificial neural networks for solving ordinary and partial differential equations”. In: *IEEE transactions on neural networks* 95 (1998), pp. 987–1000.

[Lon+18] Z. Long et al. *PDE-Net: Learning PDEs from Data*. Ed. by J. Dy and A. Krause. Oct. 2018. URL: https://proceedings.mlr.press/v80/long18a.html.

[LP96] S. Li and L. Petzold. “Stability of Moving Mesh Systems of Partial Differential Equations”. In: *SIAM Journal on Scientific Computing* 20 (1996).

[LTZ01] R. Li, T. Tang, and P. Zhang. “Moving Mesh Methods in Multiple Dimensions Based on Harmonic Maps”. In: *Journal of Computational Physics* 170.2 (2001), pp. 562–588. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2001.6749. URL: https://www.sciencedirect.com/science/article/pii/S002199910196749X.

[LW10] A. Logg and G. N. Wells. “DOLFIN”. In: *ACM Transactions on Mathematical Software* 37.2 (Apr. 2010), pp. 1–28. ISSN: 1557-7295. DOI: 10.1145/1731022.1731030. URL: http://dx.doi.org/10.1145/1731022.1731030.

[MCB18] A. T. T. McRae, C. J. Cotter, and C. J. Budd. “Optimal-Transport–Based Mesh Adaptivity on the Plane and Sphere Using Finite Elements”. In: *SIAM Journal on Scientific Computing* 40.2 (2018), A1121–A1148. DOI: 10.1137/16M1109515. eprint: https://doi.org/10.1137/16M1109515. URL: https://doi.org/10.1137/16M1109515.

[Mei69] A.L.F. Meister. *Generalia de genesi figurarum planarum et inde pendentibus earum affectionibus*. 1769.

[MG90] Shepherd M. and Theodore G. “Symmetries, conservation laws, and Hamiltonian structure in geophysical fluid dynamics”. In: *Advances in Geophysics*. Vol. 32. Elsevier, 1990, pp. 287–338.

[MGO05] P. Müller, C. Garrett, and A. Osborne. “Rogue waves”. In: *Oceanography* 18.3 (2005), p. 66.

[ML21] Y. Ming and P. Liao. “Deep Nitsche Method: Deep Ritz Method with Essential Boundary Conditions”. In: *Communications in Computational Physics* 29.5 (June 2021), pp. 1365–1384. ISSN: 1991-7120. DOI: 10.4208/cicp.oa-2020-0219. URL: http://dx.doi.org/10.4208/cicp.OA-2020-0219.

[MMN18] S. Mei, A. Montanari, and P.M. Nguyen. “A mean field view of the landscape of two-layer neural networks”. In: *Proceedings of the National Academy of Sciences* 115.33 (2018), E7665–E7671. DOI: 10.1073/pnas.1806579115.

[MNS00] P. Morin, R.H. Nochetto, and K. G. Siebert. “Data Oscillation and Convergence of Adaptive FEM”. In: *SIAM Journal on Numerical Analysis* 38.2 (2000), pp. 466–488. DOI: 10.1137/S0036142999360044. eprint: https://doi.org/10.1137/S0036142999360044. URL: https://doi.org/10.1137/S0036142999360044.

[MW14] J. Melenk and T. Wihler. *A Posteriori Error Analysis of hp-FEM for singularly perturbed problems*. 2014. DOI: 10.48550/ARXIV.1408.6037. URL: https://arxiv.org/abs/1408.6037.

[Nwa+18] C. Nwankpa et al. *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*. 2018. DOI: 10.48550/ARXIV.1811.03378. URL: https://arxiv.org/abs/1811.03378.

[OPS20] J. A. A. Opschoor, P. C. Petersen, and C. Schwab. "Deep ReLU networks and high-order finite element methods". In: *Analysis and Applications* 18.05 (2020), pp. 715–770. DOI: 10.1142/S0219530519410136. eprint: https://doi.org/10.1142/S0219530519410136. URL: https://doi.org/10.1142/S0219530519410136.

[OR68] L.A. Oganesyan and L.A. Rukhovets. "Variational-difference schemes for linear second-order elliptic equations in a two-dimensional region with piecewise smooth boundary". In: *USSR Computational Mathematics and Mathematical Physics* 8.1 (1968), pp. 129–152. ISSN: 0041-5553. DOI: https://doi.org/10.1016/0041-5553(68)90008-6. URL: https://www.sciencedirect.com/science/article/pii/0041555368900086.

[Pas+19] A. Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[PLW05] T. Plewa, T. Linde, and V.G. Weirs. *Adaptive Mesh Refinement - Theory and Applications: Proceedings of the Chicago Workshop on Adaptive Mesh Refinement Methods*. Lecture Notes in Computational Science and Engineering. Springer Berlin Heidelberg, 2005.

[Pry12] T. Pryer. *Applications of nonvariational finite element methods to Monge–Ampère type equations*. 2012. DOI: 10.48550/ARXIV.1203.0660. URL: https://arxiv.org/abs/1203.0660.

[Pry89] J. D. Pryce. "On the Convergence of Iterated Remeshing". In: *IMA Journal of Numerical Analysis* 9.3 (July 1989), pp. 315–335. ISSN: 0272-4979. DOI: 10.1093/imanum/9.3.315. eprint: https://academic.oup.com/imajna/article-pdf/9/3/315/2834314/9-3-315.pdf. URL: https://doi.org/10.1093/imanum/9.3.315.

[PV18] P. Petersen and F. Voigtlaender. "Optimal approximation of piecewise smooth functions using deep ReLU neural networks". In: *Neural Networks* 108 (2018), pp. 296–330. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2018.08.019. URL: https://www.sciencedirect.com/science/article/pii/S0893608018302454.

[QS07] S. Quan and D. P. Schmidt. "A moving mesh interface tracking method for 3D incompressible two-phase flows". In: *Journal of Computational Physics* 221.2 (2007), pp. 761–780. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2006.06.044. URL: https://www.sciencedirect.com/science/article/pii/S0021999106003159.

[RCS05] J. Ruichen, W. Chen, and A. Sudjianto. "An efficient algorithm for constructing optimal design of computer experiments". English (US). In: *Journal of Statistical Planning and Inference* 134.1 (Sept. 2005), pp. 268–287. ISSN: 0378-3758. DOI: 10.1016/j.jspi.2004.02.014.

[Rem+00] J.F. Remacle et al. "Anisotropic adaptive simulation of transient Flows using Discontinuous Galerkin methods". In: *International Journal for Numerical Methods in Engineering* 00 (Jan. 2000), pp. 1–6. DOI: 10.1002/nme.1196.

[RK18] M. Raissi and G.E. Karniadakis. “Hidden physics models: Machine learning of nonlinear partial differential equations”. In: *Journal of Computational Physics* 357 (2018), pp. 125–141. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2017.11.039. URL: http://dx.doi.org/10.1016/j.jcp.2017.11.039.

[RN94] I. Roulstone and J. Norbury. “A Hamiltonian structure with contact geometry for the semi-geostrophic equations”. In: *Journal of Fluid Mechanics* 272 (1994), pp. 211–234.

[RPD06] W. Rachowicz, D. Pardo, and L. Demkowicz. “Fully automatic hp-adaptivity in three dimensions”. In: *Computer Methods in Applied Mechanics and Engineering* 195.37 (2006), pp. 4816–4842. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2005.08.022. URL: https://www.sciencedirect.com/science/article/pii/S0045782505005098.

[RPK19] M. Raissi, P. Perdikaris, and G.E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (2019), pp. 686–707. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045. URL: https://www.sciencedirect.com/science/article/pii/S0021999118307125.

[Rud16] S. Ruder. “An overview of gradient descent optimization algorithms”. In: *ArXiv* abs/1609.04747 (2016).

[Sch98] C. Schwab. *p-and hp-finite element methods: Theory and applications in solid and fluid mechanics*. Oxford University Press, 1998. URL: http://www.jstor.org/stable/2698818.

[SH16] K. Seokchan and L. Hyung-Chun. “A finite element method for computing accurate solutions for Poisson equations with corner singularities using the stress intensity factor”. In: *Computers & Mathematics with Applications* 71.11 (2016). Proceedings of the conference on Advances in Scientific Computing and Applied Mathematics. A special issue in honor of Max Gunzburgers 70th birthday, pp. 2330–2337. ISSN: 0898-1221. DOI: https://doi.org/10.1016/j.camwa.2015.12.023. URL: http://www.sciencedirect.com/science/article/pii/S089812211500591X.

[SH74] I.E. Sutherland and G.W. Hodgman. “Reentrant Polygon Clipping”. In: *Commun. ACM* 17.1 (1974), pp. 32–42.

[SKM21] A. Shevchenko, V. Kungurtsev, and M. Mondelli. “Mean-field Analysis of Piecewise Linear Solutions for Wide ReLU Networks”. In: *CoRR* abs/2111.02278 (2021). arXiv: 2111.02278. URL: https://arxiv.org/abs/2111.02278.

[Smi86] G. D. Smith. “Numerical Solution of Partial Differential Equations: Finite Difference Methods”. In: Oxford University Press, U.S.A., 1986. ISBN: 978-0198596509. URL: https://wp.kntu.ac.ir/ghoreishif/smith.pdf.

[SS18] J. Sirignano and K. Spiliopoulos. “DGM: A deep learning algorithm for solving partial differential equations”. In: *Journal of Computational Physics* 375 (2018), pp. 1339–1364. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.08.029. URL: https://www.sciencedirect.com/science/article/pii/S0021999118305527.

[Str08] G. Strang. *An analysis of the finite element method*. eng. 2nd edition. Wellesley, MA: Wellesley-Cambridge, 2008. ISBN: 0980232708.

[SW05] S. Sun and M. Wheeler. “Discontinuous Galerkin methods for coupled flow and reactive transport problems”. In: *Applied Numerical Mathematics* 52 (Feb. 2005), pp. 273–298. DOI: 10.1016/j.apnum.2004.08.035.

[Tan05] T. Tang. “Moving mesh methods for computational fluid dynamics”. In: *Contemp. Math.* 383 (Jan. 2005). DOI: 10.1090/conm/383/07162.

[Ver94] R. Verfürth. “A posteriori error estimation and adaptive mesh-refinement techniques”. In: *Journal of Computational and Applied Mathematics* 50.1 (1994), pp. 67–83. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(94)90290-9. URL: https://www.sciencedirect.com/science/article/pii/0377042794902909.

[WB18] E. Weinan and Y. Bing. “The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems”. English (US). In: *Communications in Mathematics and Statistics* 6 (2018), pp. 1–12. ISSN: 2194-6701. DOI: 10.1007/s40304-018-0127-z.

[Wel+16] H. Weller et al. “Mesh adaptation on the sphere using optimal transport and the numerical solution of a Monge-Ampère type equation”. In: *Journal of Computational Physics* 308 (2016), pp. 102–123.

[Wih03] T. Wihler. “Discontinuous Galerkin FEM for elliptic problems in polygonal domains”. PhD thesis. Apr. 2003.

[Win66] A.M. Winslow. “Numerical solution of the quasilinear poisson equation in a nonuniform triangle mesh”. In: *Journal of Computational Physics* 1.2 (1966), pp. 149–172.

[WTP21] S. Wang, Y. Teng, and P. Perdikaris. “Understanding and mitigating gradient pathologies in physics-informed neural networks”. In: *SIAM J. Sci. Comput.* 43 (2021), A3055–A3081.

[Xu+10] X. Xu et al. “Convergence of de Boor’s algorithm for the generation of equidistributing meshes”. In: *IMA Journal of Numerical Analysis* 31.2 (Mar. 2010), pp. 580–596. ISSN: 0272-4979. DOI: 10.1093/imanum/drp052. eprint: https://academic.oup.com/imajna/article-pdf/31/2/580/1930467/drp052.pdf. URL: https://doi.org/10.1093/imanum/drp052.

[Yar17] D. Yarotsky. “Error bounds for approximations with deep ReLU networks”. In: *Neural Networks* 94 (2017), pp. 103–114. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2017.07.002. URL: https://www.sciencedirect.com/science/article/pii/S0893608017301545.

[Yua06] Y.X. Yuan. “A new step-size for the Steepest Descent Method”. In: *Journal of Computational Mathematics* 24.2 (2006), pp. 149–156. ISSN: 02549409, 19917139. URL: http://www.jstor.org/stable/43694074 (visited on 06/19/2022).

[Zha+93] H. Zhang et al. “Discrete form of the GCL for moving meshes and its implementation in CFD schemes”. In: *Computers & Fluids* 22.1 (1993), pp. 9–23. ISSN: 0045-7930. DOI: https://doi.org/10.1016/0045-7930(93)90003-R. URL: https://www.sciencedirect.com/science/article/pii/004579309390003R.

[Zha06] Z. Zhang. “Moving mesh method with conservative interpolation based on L^2 projection”. In: *Communications in Computational Physics* 1 (2006).

[ZHS09] J. Zou, Y. Han, and S.S. So. “Overview of Artificial Neural Networks”. In: *Methods in molecular biology (Clifton, N.J.)* 458 (Jan. 2009), pp. 14–22. DOI: 10.1007/978-1-60327-101-1_2.

[ZS02] C. Zhiqiang and K. Seokchan. “A Finite Element Method Using Singular Functions for the Poisson Equation: Corner Singularities”. In: *SIAM Journal on Numerical Analysis* 39.1 (2002), pp. 286–299. ISSN: 00361429. URL: http://www.jstor.org/stable/3062090.

[ZSG02] C. Zhiqiang, K. Seokchan, and W. Gyungsoo. “A finite element method using singular functions for the Poisson equation: crack singularities”. In: *Numerical Linear Algebra with Applications* 9.67 (2002), pp. 445–455. DOI: https://doi.org/10.1002/nla.303. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nla.303. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.303.